# EARLY FAULT DETECTION AND ROOT CAUSE ANALYSIS FOR MANUFACTURING SYSTEMS WITH A HYBRID MACHINE LEARNING FRAMEWORK

Kaustubh Indane [1, *, †]

1. Northwestern University

* kindane/early-fault-detection-and-RCA

† Kaustubh.indane@gmail.com

# Abstract

Manufacturing systems are highly vulnerable to processing faults that can cause unplanned downtime, yield loss, and quality degradation. Traditional monitoring methods often depend on static thresholds and post-failure diagnostics, limiting their ability to provide early warnings or actionable insights. This study introduces a hybrid machine learning framework for proactive fault detection and root cause analysis using the Tennessee Eastman Process dataset. The framework integrates a supervised XGBoost classifier for identifying known faults with interpretable SHAP analysis, and an unsupervised autoencoder trained on fault-free data to capture previously unseen failure modes.The XGBoost model achieved 93% accuracy with balanced precision and recall, detecting most faults within 36 minutes of onset. SHAP interpretability highlighted key process variables such as cooling water flow, stripper conditions, and feed compositions, providing actionable guidance for engineering diagnosis. The autoencoder detected 75% of unseen faults with high faulty precision (0.998) and strong fault-free recall (0.964), though with higher computational cost and an average detection time of 69 minutes from fault onset. Partial residual analysis further revealed critical variables driving deviations, enabling interpretability even in unlabeled fault conditions. Together, these models form a scalable and interpretable fault monitoring system that balances speed, accuracy, and adaptability, advancing proactive fault management in industrial environments.

# Table of Contents

Abstract ..... i
Introduction and Research Objectives ..... 1
Literature Review ..... 2
Data Introduction and Preparation ..... 4
XGBoost Data Preparation ..... 5
Autoencoder Data Preparation ..... 6
Feature Engineering ..... 7
XGBoost Features ..... 7
Autoencoder Features ..... 8
Methods ..... 8
XGBoost ..... 8
Autoencoder ..... 9
Results and Analysis ..... 10
XGBoost ..... 10
XGBoost Root Cause Analysis ..... 14
Autoencoder ..... 16
Autoencoder Root Cause Analysis ..... 18
Conclusion ..... 19
Directions for Future Work ..... 20
Acknowledgements ..... 21
Data Availability ..... 21
Code Availability ..... 21
References ..... 22
Appendix A ..... 24
Appendix B ..... 25
Appendix C ..... 26

# Introduction and Research Objectives

In high-volume manufacturing environments, such as chemical processing, minor process deviations can result in significant financial loss through reduced yield and extended downtime. Traditionally, fault detection and diagnosis rely on threshold-based alarms, manual root cause investigations, and post-failure analysis. These conventional systems tend to identify problems after product quality has already been compromised, limiting their effectiveness for proactive decision-making. According to Deloitte (2021), manufacturers experience an average of 800 hours of equipment downtime per year, with an estimated cost of $260,000 per hour in lost productivity, materials, and labor. By proactively identifying the onset of faults and understanding their root causes in real-time, organizations can reduce downtime, protect product quality, and improve the speed of engineering decision-making.

This study explores a deep/machine learning approach to early fault detection and root cause analysis using the Tennessee Eastman Process (TEP), a widely used simulation benchmark for complex industrial systems. This study will demonstrate how modern data techniques can move fault detection and management from a reactive to a proactive task. The system developed will enable faster identification of known and unknown failure modes while also identifying key process variables that indicate various failures, providing interpretable insights to support engineering diagnostics.

The following research questions will be addressed:

1. **Can process faults be detected early before manifesting into system failures and product impact?**
2. **Which process variables contribute most to various fault types?**

3. **Can unseen failure modes or faults be detected?**

To answer these questions, the modelling approach will cover:

1. XGBoost classifier for early fault detection and classification with interpretable SHAP variable insights
2. Unsupervised autoencoder to detect unknown/unseen faults

Accuracy, F1-score, and evaluation time will be key metrics to measure performance of the supervised XGBoost model as real-time analysis is key in manufacturing. For root cause analysis, SHAP values and feature attribution heatmaps will provide interpretability of variable importance. The autoencoder will be evaluated based on reconstruction error, false positive rates, and how early it flags faults relative to their onset.

Keywords:

- Faults: Abnormalities in process behavior that impact tool processing, leading to poor product quality.
- Root Cause Analysis (RCA): Systematic approach to identifying the underlying factors that contribute to observed faults, helping engineers explain why a fault occurred or which fault occurred.

# Literature Review

Early fault detection and root cause analysis in industrial systems is an active area of research. Several studies have laid the groundwork for the work in this study.

Ricker (1996) evaluated univariate control chart techniques on TEP and concluded that they failed to detect more than half of the faults, especially those with gradual drift or multivariate interactions.

Chiang et al. (2001) applied Principal Component Analysis and Partial Least Squares to the TEP dataset, finding that PCA-based fault detection could identify 13 out of 21 faults with over 90% accuracy, though interpretability remained limited.

Ge et al. (2013) reviewed over 100 studies in process fault diagnosis and concluded that hybrid approaches, combining statistical models with domain knowledge, improved detection rates and fault classification precision, in some cases by 20–30% over traditional methods.

Krishnan et al. (2020) employed LSTM networks on multivariate industrial time-series data and reported a 14% improvement in early detection time and a 12% gain in fault classification accuracy compared to feedforward networks.

In unsupervised learning, Zhao et al. (2018) used autoencoders trained only on fault-free data to detect unknown anomalies in process variables and achieved an AUC of 0.93 for multiple fault types, while maintaining a false positive rate below 5%.

This study builds upon these prior works to create a hybrid fault detection and RCA system that is accurate, interpretable, and scalable. Unlike many prior studies, this study emphasizes early fault detection and root cause diagnosis.

# Data Introduction and Preparation

| Column | Description | Type |
|---|---|---|
| faultNumber | Classification of fault in simulation run (0 = fault-free, 1-20 = fault) | int32 |
| simulationRun | Simulation run number (1-500) | int32 |
| sample | Sample number per simulation (one sample collected every 3 minutes over 24/48-hour periods) | int32 |
| xmeas(1-41) | Measured process variables | float64 |
| Xmv(1-11) | Manipulated process variables | float64 |

**Table 1. Data Dictionary – Each *simulationRun* contains only one *faultNumber*. If faults exist in a simulation, they are introduced at hour 1 (sample = 20). *Xmeas* includes 41 dependent process variable data collected and *Xmv* includes 11 manipulated independent process variables, decoded in Appendix B**

The dataset is from the Tennessee Eastman Process, a widely used chemical production simulation benchmark developed by Eastman Chemical Company. It was curated and made public by Kaggle user Averkij, containing multivariate time-series data, representing both fault-free and fault-injected simulations (Table 1). 500 simulations of fault free data and 500 simulations of faulty data are used.

**Appendix A** displays the simulation process diagram with detailed information about components and measurements. Table 2 details the nature of each fault condition.

**Table 2. Description of faultNumber – 0 represents fault-free data. Faults 1-7 identify a step shift in process condition, and 8-12 introduce random variation. Fault 13 defines drifts in reaction kinetics, and faults 14-15 are sticking valves. Faults 16-20 are undefined faults introduced in the data. Source: *Russell, Chiang, and Braatz 2000, 105***

| faultNumber | Fault Description | Fault Type |
|---|---|---|
| 0 | None | None |
| 1 | A/C feed ratio, B composition constant | Step |
| 2 | B composition, A/C ratio constant | Step |
| 3 | D feed temperature | Step |
| 4 | Reactor cooling water inlet temperature | Step |
| 5 | Condenser cooling water inlet temperature | Step |
| 6 | A feed loss | Step |
| 7 | C header pressure loss-reduced availability | Step |
| 8 | A, B, C feed composition | Random variation |
| 9 | D feed temperature | Random variation |
| 10 | C feed temperature | Random variation |
| 11 | Reactor cooling water inlet temperature | Random variation |
| 12 | Condenser cooling water inlet temperature | Random variation |
| 13 | Reaction kinetics | Slow drift |
| 14 | Reactor cooling water valve | Sticking |
| 15 | Condenser cooling water valve | Sticking |
| 16 | Unknown | Unknown |
| 17 | Unknown | Unknown |
| 18 | Unknown | Unknown |
| 19 | Unknown | Unknown |
| 20 | Unknown | Unknown |

## XGBoost Data Preparation

XGBoost modelling will only consider a subset of faults. Faults 16-20 will be excluded and preserved for unsupervised autoencoder detection of unknown/unseen faults. Faults 3, 9, and 15 are excluded due to their historically poor detectability in literature. These faults exhibit subtle and extremely complex patterns often indistinguishable between noise and normal conditions of the system. However, recent work by Zhang et al. (2024) introduced AE-FEnet, a deep feature ensemble model designed to detect *only* faults 3, 9, and 15, achieving over 96% accuracy. Nonetheless, detecting these faults is out of the scope of this study.

To focus on early detection power, only a subset of samples will be considered for each simulationRun (Figure 1). Faults are introduced at hour 1 of the simulation at sample 20.

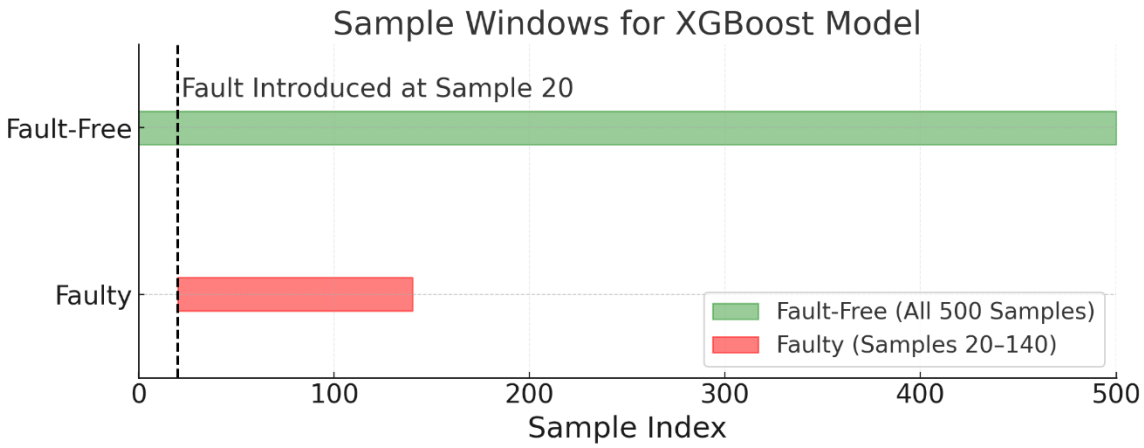



**Figure 1. Sample Windows for XGBoost Model – All faulty-free samples will be used to distinguish normal behavior. For faulty data, samples 20-140 will be used, hour 1 to hour 7, to focus on early fault detection.**

## Autoencoder Data Preparation

For autoencoder modeling, a simple feedforward network will be used. All features are standardized. A sliding window technique transformed sequential data into fixed-size samples suitable for input, illustrated in Figure 2. In the case of faulty data, only samples after the known fault injection point (sample 20) were used to generate windows. For fault-free data, all 500 samples were used.

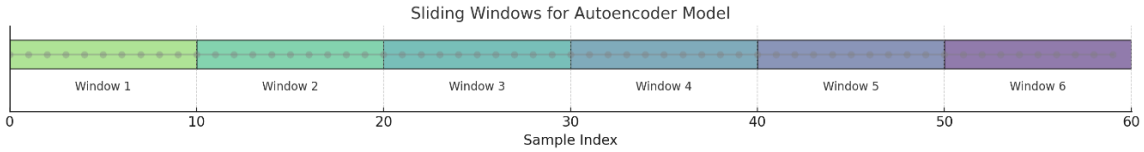



**Figure 2. Sliding Windows for Autoencoder Model – Each simulationRun is divided into windows of 10 samples. Each window will be an independent input into the model.**

# Feature Engineering

## XGBoost Features

**Appendix C** illustrates each predictor by sample in the first fault-free simulation, showing noisy data across all variables. To increase signal-to-noise ratio, windowed moving averages are calculated for each predictor, Xmeas and Xmv, at every sample with a window of 3 samples. Additionally, each sample will be independently trained and classified for faultNumber. Therefore, to incorporate temporal information, lag variables are added. This lag window, defined as x, incorporates the fixed value of the previous x samples.

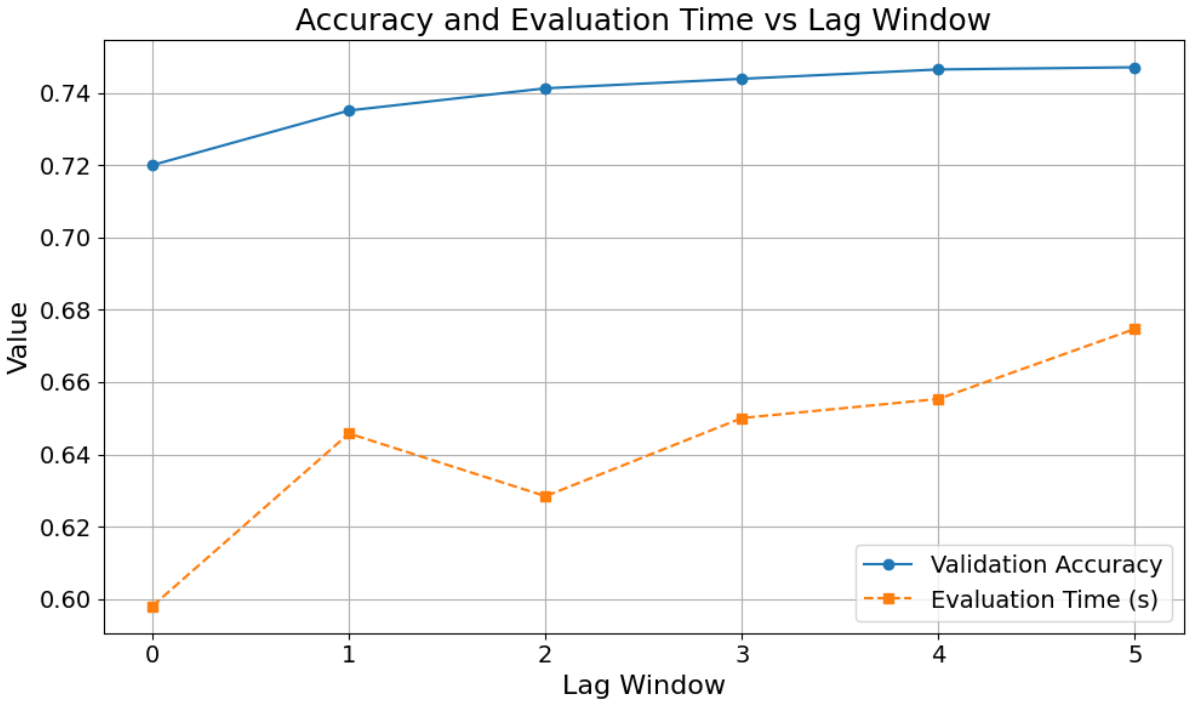



**Figure 3. Accuracy and Evaluation Time vs Lag Window – Initial validation accuracy increases with exponential loss at larger lag windows, sacrificing evaluation time and storage. Due to real-time high-accuracy requirements, a lag window of '2' was determined as the best option. This choice enables the model to run with limited storage on edge devices while incorporating temporal information.**

## Autoencoder Features

Unlike XGBoost, features to incorporate temporal information were not used in autoencoder modeling. Temporal information was indirectly accommodated by feeding the model a window of 10 samples for each input, as discussed previously.

# Methods

## XGBoost

| **Hyperparameters** |
| --- |
| { subsample=0.6,<br>n_estimators=200,<br>max_depth=8,<br>learning_rate=0.1,<br>gamma=1,<br>colsample_bytree=0.8 } |

**Table 3. XGBoost Hyperparameters – The model uses 200 trees with 8 maximum depth to balance complexity and overfitting. A subsample rate of 0.6 and column sampling of 0.8 improve generalization, while a learning rate of 0.1 ensures stable yet fast convergence. Gamma of 1 penalizes overly complex splits. Multiclass softmax provides the final classification. All hyperparameters were tuned for best overall accuracy.**

The XGBoost model will provide a fault classification at each time-step (sample). To identify the start of a fault, 3 consecutive samples with predicted labels of the same fault will determine the output. Once 3 consecutive samples conclude a fault has occurred, all following samples will be predicted with the same fault, as the process does not recover from faults without human intervention. A stratified 60% train, 20% validation, and 20% test set are used.

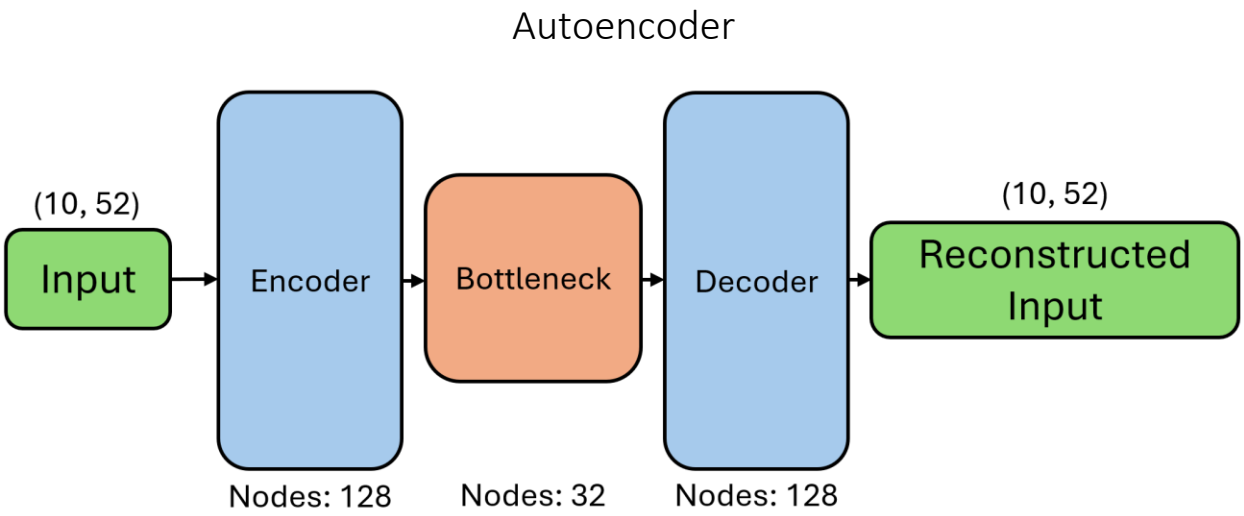



**Figure 4. Autoencoder Model – The input consists of 10-sample windows with 52 features, fed into a 128-unit encoder layer. The bottleneck encoded representation of the input has 32 units. A 128-unit decoder layer reconstructs the input data as the output. The model is trained with MSE with Adam optimizer (learning rate = 0.001), with early stopping on validation loss.**

The unsupervised autoencoder model is exclusively trained on fault-free data to reconstruct normal process behavior. A 60% training, 20% validation, and 20% test set of fault-free data are created. Then, all simulations from faults 16-20 are appended to the test set (Note: The test set is imbalanced with 100 fault-free simulations and 125 faulty simulations).

To distinguish between normal and faulty process, a reconstruction error threshold will be determined from a distribution of error from each class (fault-free or faulty). This will enable us to balance sensitivity of faults vs normal system variability. To determine this threshold and the effectiveness of the model, evaluation will consist of label accuracy, precision, and recall for fault and fault-free test data. Additionally, evaluation time will be considered for speed and computation cost. Average detection delay, defined as time between fault onset and first detection per simulationRun, will be calculated to evaluate early detection power.

# Results and Analysis

## XGBoost

| Model | Validation Accuracy | Test Accuracy | Weighted Precision | Weighted Recall | Weighted F1-Score | Evaluation Time (s) |
|---|---|---|---|---|---|---|
| XGBoost | 93% | 93% | 94% | 93% | 93% | 1.15 |

**Table 4. XGBoost Modeling Results – Fault classification results for XGBoost model. Validation accuracy and test accuracy, precision, recall, F1-Score, and Evaluation Time are key metrics to measure performance with powerful classification and real-time capability.**

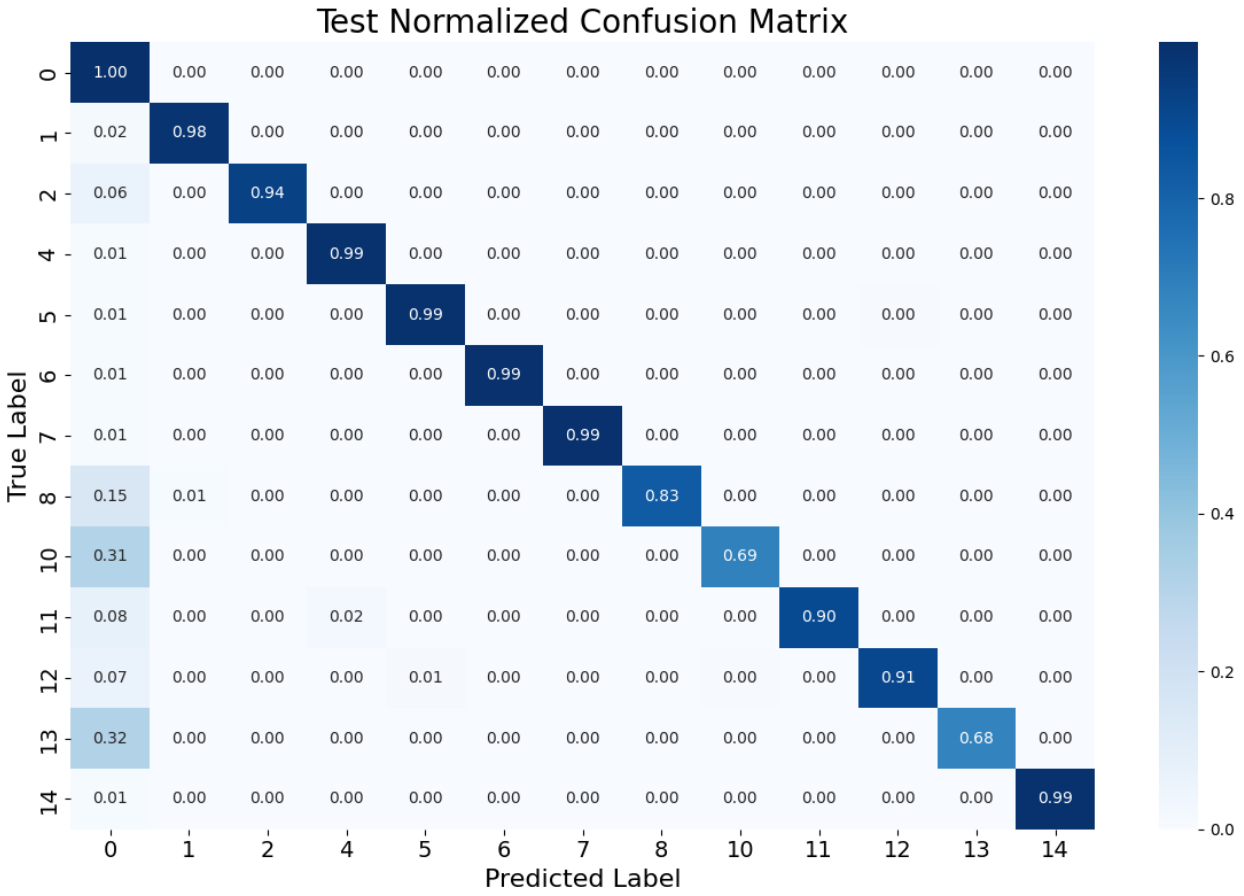



**Figure 5. Test Set Normalized Confusion Matrix – XGBoost model provides very strong detection in most faults. Faults 8, 10, and 13 are the most difficult to classify.**

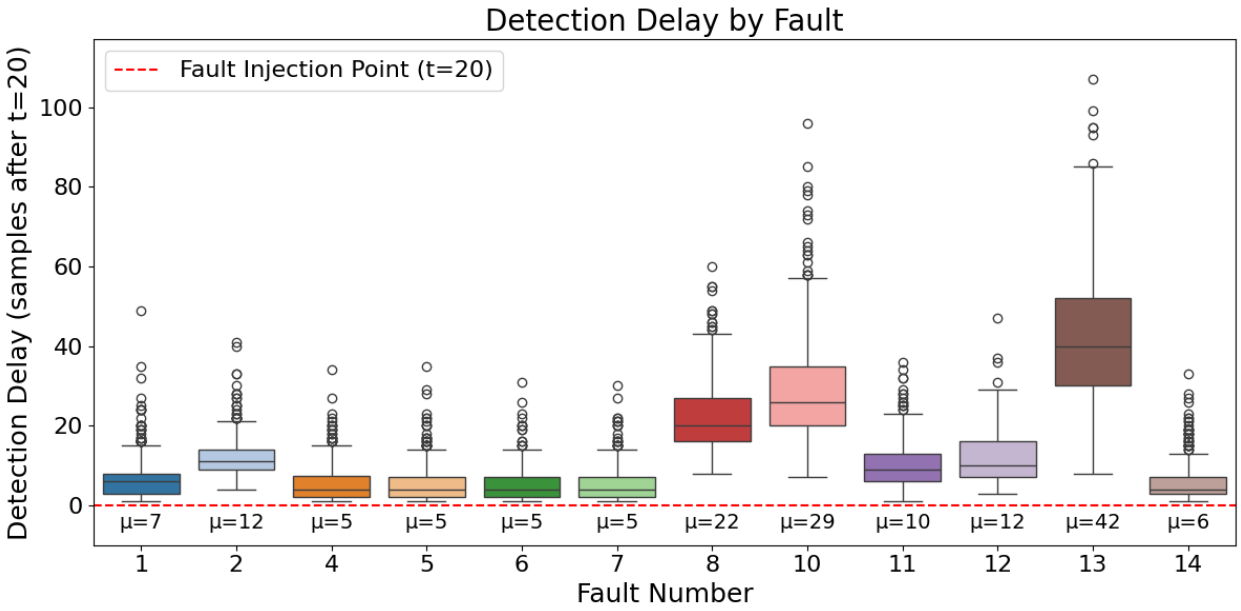



**Figure 6. Detection Delay by Fault – The number of samples after fault insertion, at sample 20, required to determine when a fault has occurred. Each point represents one simulationRun. Most faults are detectable early within a mean detection delay of 12 samples (36 minutes). Faults 8, 10, and 13 take longer to manifest and require an average of up to 42 samples (~2 hours) to detect.**

Table 4 shows strong performance and generalizability with validation and test accuracy at 93%. A weighted F1-score of 93% indicates balanced performance across most fault classes, with minimal precision-recall tradeoffs. This model can minimize both false positives and negatives. Additionally, the model evaluated 1000 simulation runs with a 1.15 second evaluation time, validating its suitability for real-time fault detection on manufacturing edge devices. This was achieved despite incorporating time-lag features to capture short-term temporal dependencies, a tradeoff that preserved detection power without incurring significant computational overhead.

The confusion matrix (Figure 5) reveals most fault times were classified with high accuracy, especially Fault 1, 4, 5, 6, 7, and 14 with near perfect classification. Performance degraded for Faults 8, 10, 13. These faults were often misclassified as fault-free. Figure 6

illustrates that these three faults had the highest detection delays, with 22, 29, and 42 samples respectively. As XGBoost modelling considered each sample point independently, a late detection delay explains the low accuracy of these three faults. Due to the nature of these faults, they take longer to manifest in process variables, contributing to lower accuracy measures and longer detection delay. For example, Fault 13 involves show drift in reaction kinetics, which can resemble normal system variability over short windows.

Fault onset occurs at sample 20 (1 hour), and the model flags most correct faults within 12 samples on average (36 minutes from onset). This provides valuable early detection of faults for operators to respond before large product quality impact. For faults 8, 10, and 13, the average delay extends up to 42 samples (~2 hours) but still provides a meaningful early warning.

Overall, these results demonstrate that the XGBoost detector offers a powerful balance of high detection accuracy and early fault recognition, within the computational constraint of real-time industrial environments.

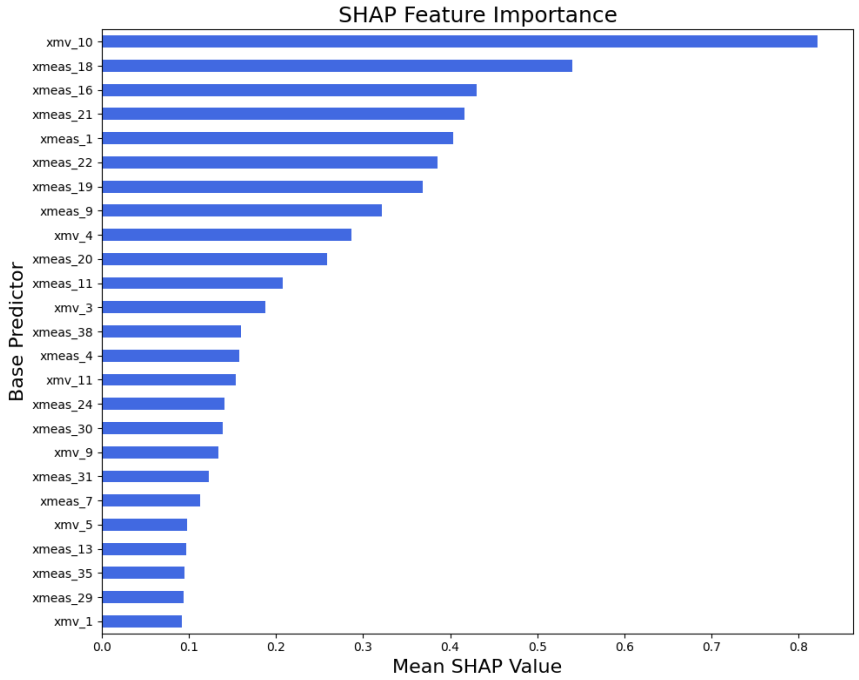



**Figure 7. SHAP Feature Importance of base variables, accommodating both lag and non-lag variations. SHAP helps identify most meaningful process variables for fault detection, highlighting xmv_10, xmeas_18, xmeas_16, xmeas_21, and xmeas_1 as most important predictors.**

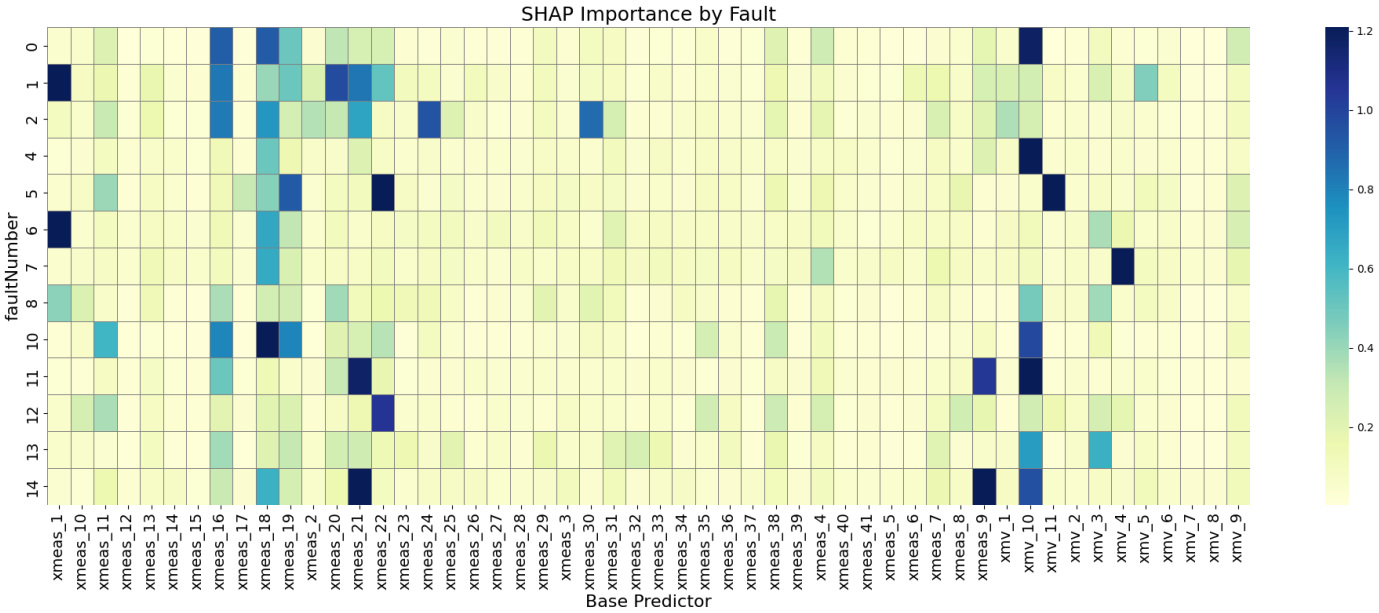



**Figure 8. SHAP Importance by Fault – The most important variables to detect each fault are illustrated. Similar to Figure 4, xmv_10, xmeas_18, xmeas_16, xmeas_21, and xmeas_1 provide the best detection power. Engineers can use this feature map to investigate key drivers and understand failure signatures of each fault.**

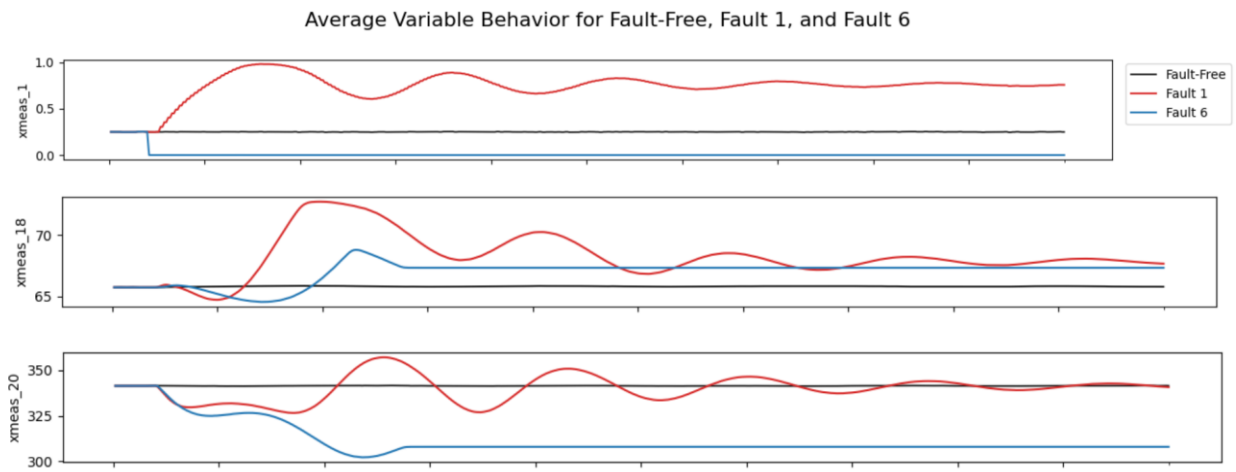



**Figure 9. Average Variable Behavior by Sample across all simulationRuns – Fault-Free, Fault 6, and Fault 1 data are illustrated for key importance features to classify these faults.**

## XGBoost Root Cause Analysis

While classification accuracy and early detection are critical, understanding why/how a fault is detected is equally important for engineering diagnosis. SHAP was applied to the XGBoost model to attribute each prediction to specific process variables. This enables root cause analysis by highlighting the most influential features for each detected fault.

Figure 7 presents the global SHAP summary across all fault types, identifying xmv_10 (reactor cooling water flow valve), xmeas_18 (stripper temperature), xmeas _16 (stripper pressure), xmeas _21 (reactor cooling water outlet temperature), and xmeas _1 (A feed stream) as the top contributors to model predictions. Identifying these process variables provides meaningful insight into where to implement process control methods.

Figure 8 breaks down feature importance per fault. This facilitates engineering diagnostics, allowing identification of variable importance for a given failure mode and improved understanding of the system. For example, Fault 1 (A/C feed ratio shift) and Fault 6 (A feed loss) both highlight xmeas_1 (A feed stream) as the most important predictor. However,

exploring figure 9 can explain the nuanced decision-making to distinguish these faults using xmeas_1, 18, and 20. Xmeas_1, which measures A feed stream, clearly illustrates A feed loss (Fault 6) in blue, as the feed stream takes a step down. For Fault 1 (A/C feed ratio step change), xmeas_1 also captures the shift but in the positive direction when compared to Fault-free data. Looking at xmeas_18 (stripper temperature), this value is a determining factor for fault 6 but not fault 1. Even though both faults exhibit similar behavior, a step change up with the same steady-state value at end of simulation, the model makes a distinction for importance of detection as xmeas_18 is more important to detect fault 6 than fault 1. This shows the root cause analysis value of SHAP importance, as engineers viewing xmeas_18 may incorrectly classify fault 6 as fault 1 or vice versa. Looking at xmeas_20 in figure 7, we notice Fault 1 introduces variability into the compressor work but reaches the same steady-state value as fault-free data. For fault 6, the compressor work drops, as there is less feed to move (a valuable insight for engineers). However, SHAP importance classifies xmeas_20 as important to predict Fault 1 rather than Fault 6. This illustrates how seemingly normal sinusoidal behavior can still be an important indication of a fault.

Overall, SHAP provides more than feature rankings as it helps disambiguate similar fault patterns and assist in root cause analysis and process understanding. Interpretability ensures engineers can derive rational, traceable insights from modeling, empowering faster fault response and targeted process improvements/control.

## Autoencoder

| Average Detection Delay (min) | Fault-Free Precision | Faulty Precision | Fault-Free Recall | Faulty Recall | Evaluation Time (s) |
|---|---|---|---|---|---|
| 65 | 0.137 | 0.998 | 0.964 | 0.748 | 34.4 |

**Table 5. Autoencoder Modeling Results – Average detection delay, precision and recall per class, and evaluation time are key metrics to determine an appropriate residual error threshold for detection and measure computation speed.**

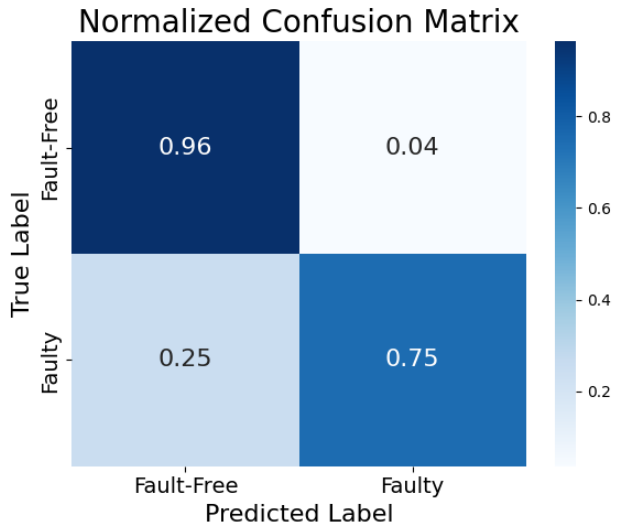



**Figure 10. Autoencoder Normalized Confusion Matrix – The chosen threshold identifies 75% of faults in the test data. It also correctly disregards 96% of Fault-free test data. However, 25% of faulty data is undetected.**

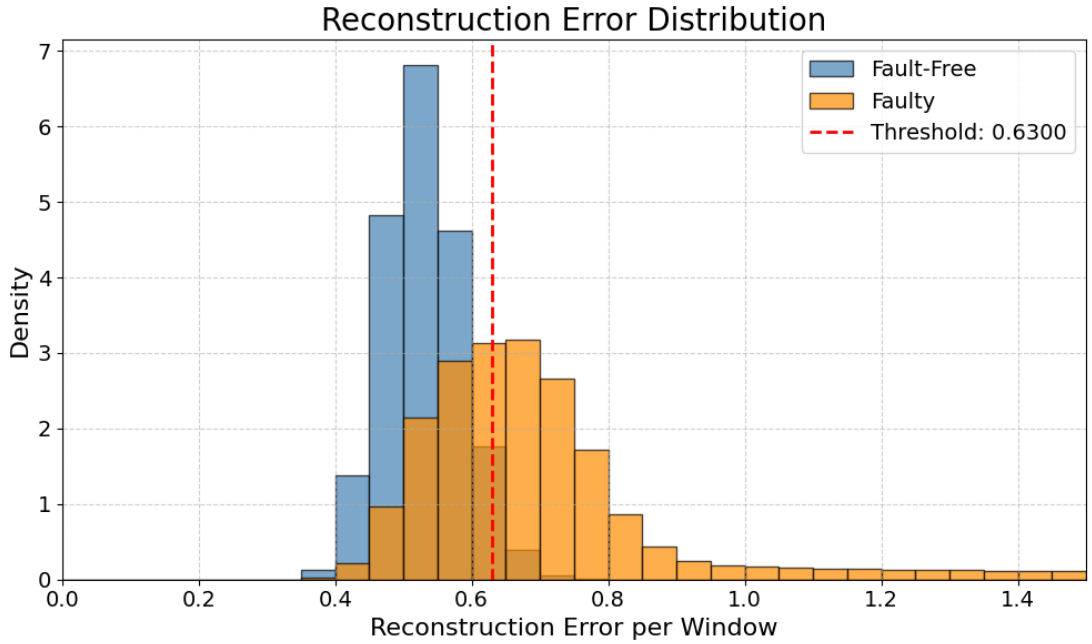



**Figure 11. Reconstruction Error Distribution – The reconstruction error distribution for both Fault-free and faulty test data. A 95% percentile value from the Fault-free distribution is chosen as the threshold value.**

Performance is determined by the reconstruction error threshold. Figure 11 shows the error distribution of fault-free and faulty data. As expected, faulty data has higher error compared to the fault-free distribution, with significant positive skew. The model obtains higher error reconstructing faulty data, as it is beyond normal operating behavior. However, there is overlap in this distribution, meaning reconstruction error alone cannot completely distinguish fault state.

In industrial settings, a false alarm is expensive. A false positive occurs when the model reports normal behavior as faulty. It is more acceptable to miss detecting faults (as there will be other traditional detection methods used simultaneously), rather than reporting a fault that is not there. It is also crucial to have confidence that when the system reports a fault, it is true. Therefore, a high faulty precision and high fault-free recall are most important, reported in Table 5 at 0.998 and 0.964 respectively. The chosen threshold is at the $95^{th}$ percentile of fault-free data (Figure 11). Figure 10 illustrates this threshold will result in 4% false positives of non-faulty runs with 75% of faults correctly detected. In higher stake processes, the threshold may be increased to $99^{th}$ percentile, resulting in less than 1% false positives at the sacrifice of less than 75% fault detection.

With the autoencoder model, an average detection delay of 69 minutes from fault onset was calculated. This model is strong at detecting faults early, although not as strong as the XGBoost model. Additionally, evaluation time of the autoencoder jumps to 34 seconds compared to XGBoost's 1.15 seconds. XGBoost is preferred for faster detection delay time and faster inference time when known fault data is available. However, it relies on supervised learning, and the autoencoder approach enables detection of unseen faults at the sacrifice of

computation and detection delay. This tradeoff highlights the utility of this hybrid framework for early fault detection pipelines.

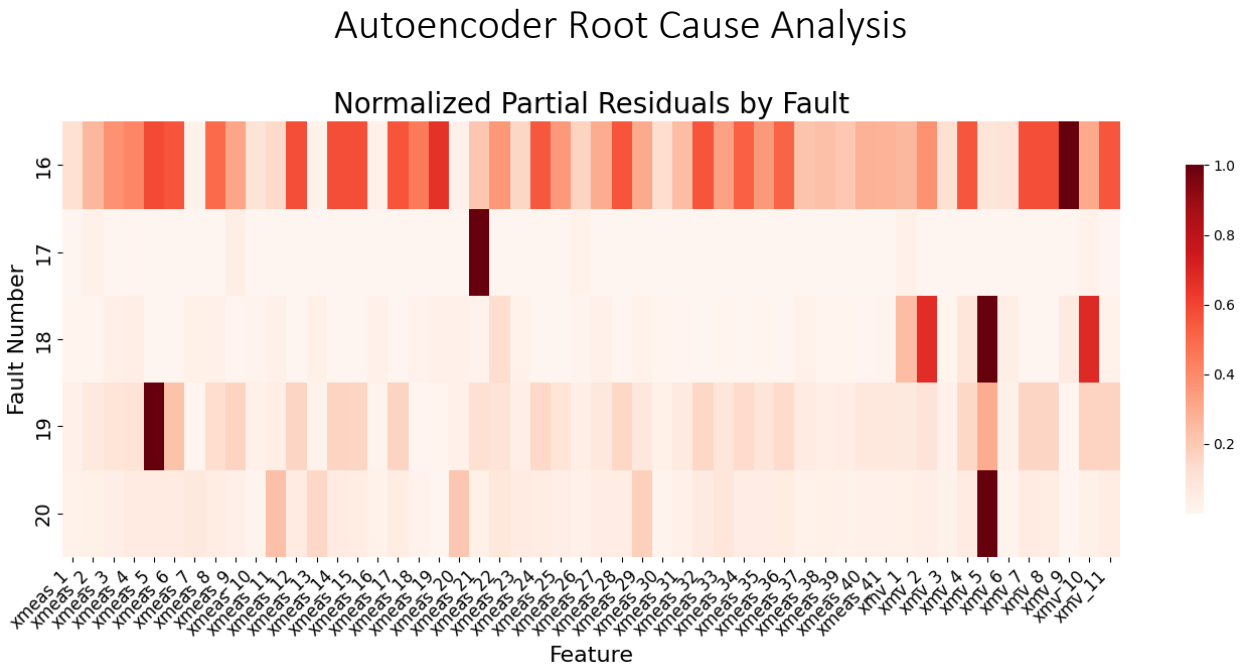



**Figure 12. Normalized Partial Residuals by Fault – The autoencoder partial residuals per feature are illustrated with true Fault labels. This enables interpretation of key driving variables for fault detection.**

To gain interpretability from the unsupervised autoencoder, partial residual analysis approximated feature-level contributions for each detected fault (Figure 12). The squared residual errors were reshaped and averaged over the input window to isolate high-error features. These residuals were then grouped by true fault number (unseen to the model). In real settings, similar calculations can be done on a single fault detection run to determine key driving features, enabling root cause analysis.

Fault 16 illustrates many system variables deviate from normal operating behavior. Therefore, this fault may represent a system-wide technology failure or a significant upstream issue. Fault 17 is dominated by xmeas_21 (stripper underflow), which may reflect sensor

calibration issues or a localized flow disruption within the stripper system. Fault 18 shows high reconstruction error for xmv_2, xmv_5, and xmv_10, all of which are tied to reflux and condenser operation. This indicates potential actuator misbehavior or improper control action affecting condenser flow rates. Similarly, fault 19 is most affected by reactor pressure, xmeas_5, and Fault 20 implicates xmv_5, condenser cooling water flow. These interpretations can guide process engineers to narrow down investigations without labeled fault details.

# Conclusion

This study proposed a hybrid machine learning framework for early fault detection and root cause analysis in complex manufacturing systems using the Tennessee Eastman Process. By combining a supervised XGBoost classifier with SHAP-based interpretability and an unsupervised autoencoder for detecting unknown failure modes, the framework offers both predictive accuracy and actionable insights for industrial decision-making.

The XGBoost model achieved high classification performance, with 93% test accuracy and balanced precision-recall across most fault types. Importantly, it provided early fault recognition, detecting most known faults within 12 samples (36 minutes) from onset, enabling proactive intervention before quality degradation or system failure. SHAP analysis further enriched model transparency, revealing key contributing process variables for each fault and facilitating engineering-driven root cause analysis.

The autoencoder extended the framework by modeling fault-free behavior and flagging deviations through reconstruction error. While detection delay averaged 69 minutes and inference cost was higher than XGBoost, the autoencoder demonstrated strong ability to

capture unseen failure modes with high faulty precision (0.998) and strong fault-free recall (0.964). Partial residual analysis additionally provided a mechanism for root cause insights, highlighting the process variables most responsible for deviations even in unlabeled fault conditions.

Together, the hybrid framework demonstrates the potential of integrating interpretable machine learning and anomaly detection to support intelligent fault monitoring and engineering diagnostics in manufacturing environments.

## Directions for Future Work

While this study demonstrates the effectiveness of a hybrid machine learning framework for early fault detection and root cause analysis, several opportunities remain to expand and refine this framework:

1. Designing an end-to-end pipeline that runs continuously along the process to actively monitor and highlight faults with high confidence and their contributing factors.
2. Exploring feature importance of each fault vs fault-free runs independently to highlight key variables deviating from the baseline.
3. Extension to multi-fault or compound fault scenarios as real-world systems may encounter overlapping or sequential faults.
4. Temporal autoencoders based on convolutions or LSTMs could further enhance fault detection.

# Acknowledgements

The author would like to thank Professor Aliana J. Maren from Northwestern University Master of Data Science Program for her guidance on academic report structure and feedback throughout this study.


# Data Availability

The data originates from a simulation of the Tennessee Eastman Process, a widely used chemical production benchmark developed by Eastman Chemical Company. It was curated and made public by Kaggle user Averkij, containing multivariate time-series data representing both fault-free and fault-injected simulations. The dataset is accessible through Kaggle at: https://www.kaggle.com/datasets/averkij/tennessee-eastman-process-simulation-dataset

# Code Availability

The code supporting the findings of this study is available at the following GitHub repository: kindane/early-fault-detection-and-RCA. EDA, feature creations, and modelling code are all documented and shared.

# References


Averkij. 2022. *Tennessee Eastman Process Simulation Dataset*. Kaggle. https://www.kaggle.com/datasets/averkij/tennessee-eastman-process-simulation-dataset

Chacharkar, Shreyash. 2022. *TEP Fault Diagnosis*. Kaggle. https://www.kaggle.com/code/shreyashchacharkar/tep-fault-diagnosis

Chiang, Leo H., Evan L. Russell, and Richard D. Braatz. 2001. *Fault Detection and Diagnosis in Industrial Systems*. London: Springer.

Deloitte. 2021. *Predictive Maintenance: Driving Improved Asset Management with Advanced Analytics*. Deloitte Development LLC. https://www2.deloitte.com/us/en/pages/operations/articles/predictive-maintenance.html

Russell, E. L., L. H. Chiang, and R. D. Braatz. 2000. *Data-Driven Methods for Fault Detection and Diagnosis in Chemical Processes*. London: Springer.

Ge, Zhengxin, Zhihuan Song, and Fuli Wang. 2013. "Review of Recent Research on Data-Based Process Monitoring." *Industrial & Engineering Chemistry Research* 52(10): 3543–3562. https://doi.org/10.1021/ie301167s

Krishnan, Shankar, Santosh Ansumali, and Harsha Vardhan. 2020. "Early Fault Detection in Industrial Processes Using LSTM Networks." *Procedia Computer Science* 170: 512–519. https://doi.org/10.1016/j.procs.2020.03.111

Ricker, Norman L. 1996. "Decentralized Control of the Tennessee Eastman Challenge Process." *Journal of Process Control* 6(4): 205–221. https://doi.org/10.1016/0959-1524(96)00016-8

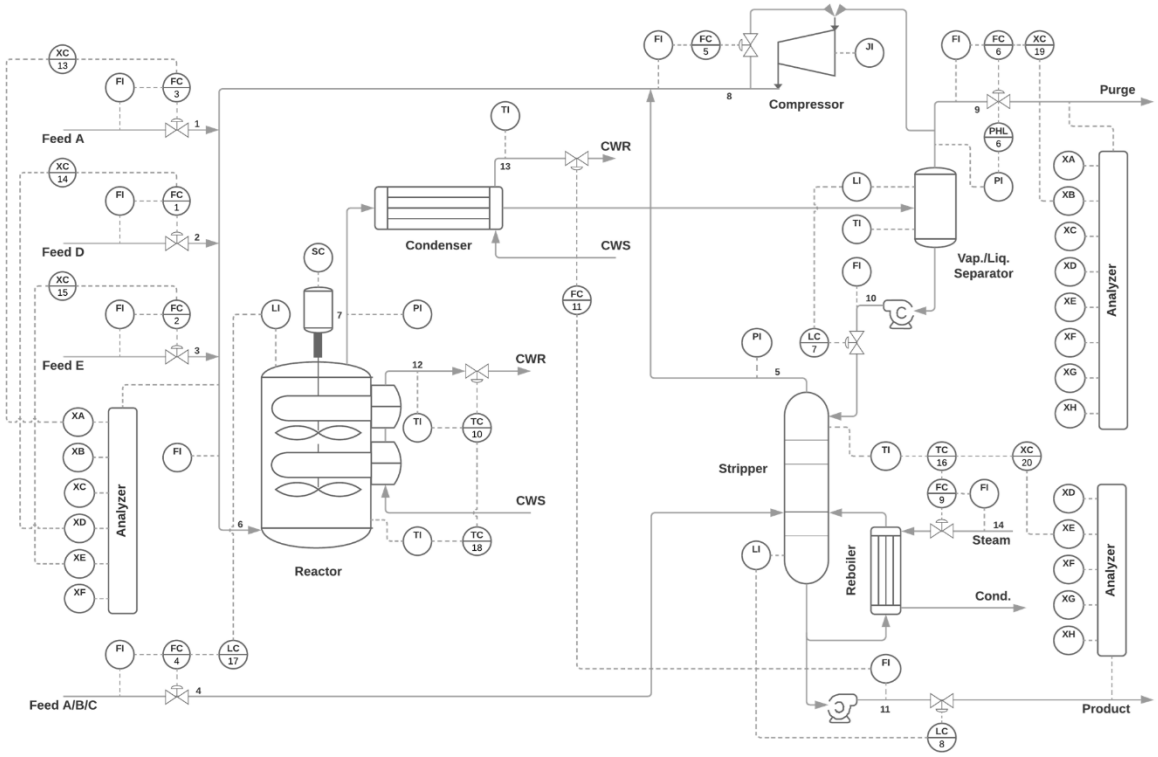
Rieth, Cory A., Margaret L. Taylor, Christina K. E. Schunn, and Michael J. Schoelles. 2017. "Issues and Advances in Anomaly Detection Evaluation for Joint Human-Automated Systems." *Cognitive Systems Research* 44: 297–311. https://doi.org/10.1016/j.cogsys.2017.05.002

Zhao, Ruilin, Ruqiang Yan, Zhenyu Chen, Kezhi Mao, Peng Wang, and Robert X. Gao. 2019. "Deep Learning and Its Applications to Machine Health Monitoring." *Mechanical Systems and Signal Processing* 115: 213–237. https://doi.org/10.1016/j.ymssp.2018.05.050

Zhang, Xinyao, Qiuyue Lu, Yue Jin, and Fei Gao. 2024. "AE-FENet: A Feature Ensemble Net Based on Autoencoder for Process Monitoring." *arXiv preprint* arXiv:2404.13941. https://arxiv.org/abs/2404.13941.

Github Copilot for code completions

# Appendix A

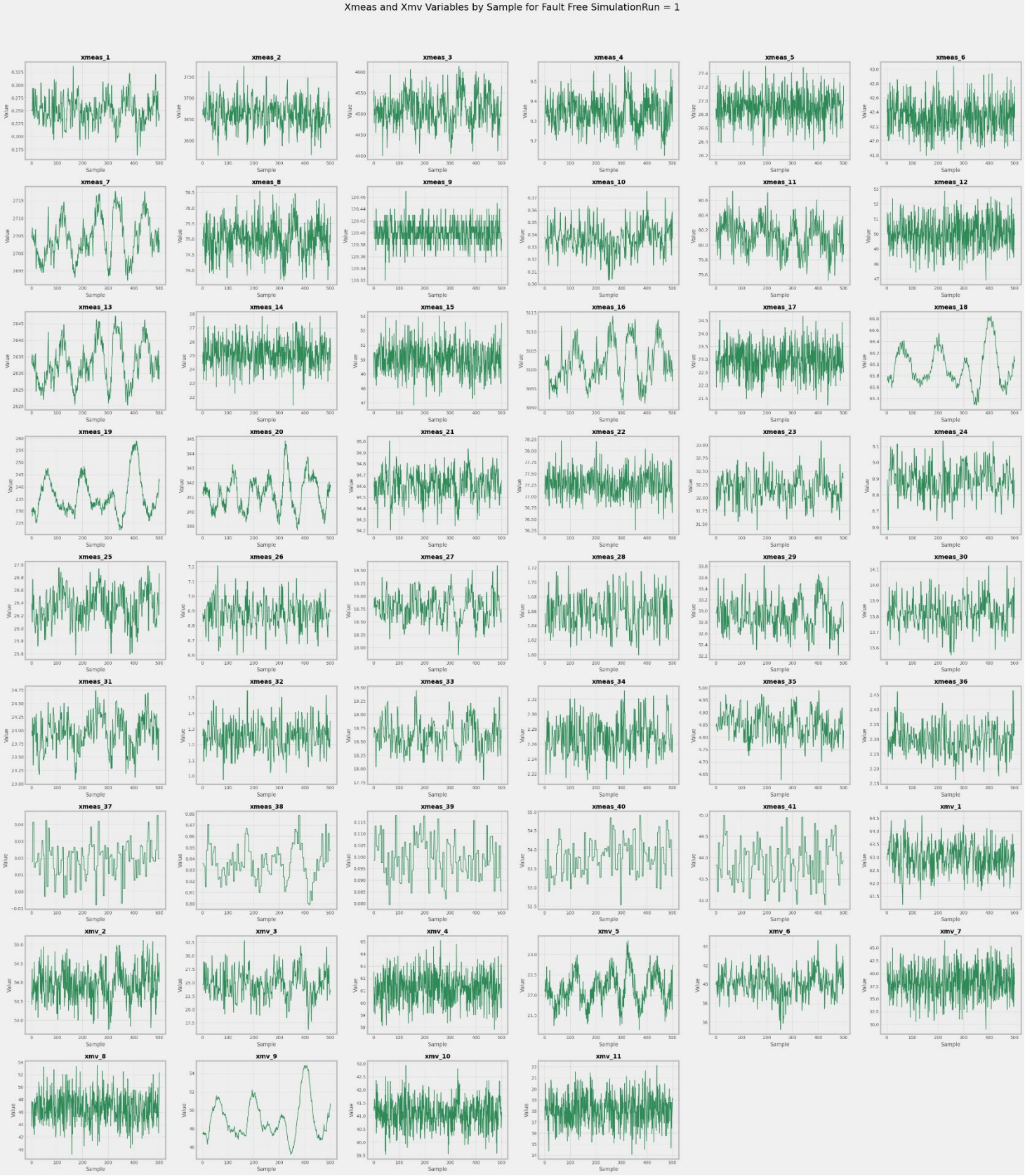



Source: *Russell, Chiang, and Braatz 2000, 105*

# Appendix B

| Variable Name | Description |
|---|---|
| XMEAS_1 | A feed stream |
| XMEAS_2 | D feed stream |
| XMEAS_3 | E feed stream |
| XMEAS_4 | Total fresh feed stripper |
| XMEAS_5 | Recycle flow into rxtr |
| XMEAS_6 | Reactor feed rate |
| XMEAS_7 | Reactor pressure |
| XMEAS_8 | Reactor level |
| XMEAS_9 | Reactor temp |
| XMEAS_10 | Purge rate |
| XMEAS_11 | Separator temp |
| XMEAS_12 | Separator level |
| XMEAS_13 | Separator pressure |
| XMEAS_14 | Separator underflow |
| XMEAS_15 | Stripper level |
| XMEAS_16 | Stripper pressure |
| XMEAS_17 | Stripper underflow |
| XMEAS_18 | Stripper temperature |
| XMEAS_19 | Stripper steam flow |
| XMEAS_20 | Compressor work |
| XMEAS_21 | Reactor cooling water outlet temp |
| XMEAS_22 | Condenser cooling water outlet temp |
| XMEAS_23 | Composition of A rxtr feed |
| XMEAS_24 | Composition of B rxtr feed |
| XMEAS_25 | Composition of C rxtr feed |
| XMEAS_26 | Composition of D rxtr feed |
| XMEAS_27 | Composition of E rxtr feed |
| XMEAS_28 | Composition of F rxtr feed |
| XMEAS_29 | Composition of A purge |
| XMEAS_30 | Composition of B purge |
| XMEAS_31 | Composition of C purge |
| XMEAS_32 | Composition of D purge |
| XMEAS_33 | Composition of E purge |
| XMEAS_34 | Composition of F purge |
| XMEAS_35 | Composition of G purge |
| XMEAS_36 | Composition of H purge |
| XMEAS_37 | Composition of D product |
| XMEAS_38 | Composition of E product |
| XMEAS_39 | Composition of F product |
| XMEAS_40 | Composition of G product |
| XMEAS_41 | Composition of H product |
| XMV_1 | D feed flow valve |
| XMV_2 | E feed flow valve |
| XMV_3 | A feed flow valve |
| XMV_4 | Total feed flow stripper valve |
| XMV_5 | Compressor recycle valve |
| XMV_6 | Purge valve |
| XMV_7 | Separator pot liquid flow valve |
| XMV_8 | Stripper liquid product flow valve |
| XMV_9 | Stripper steam valve |
| XMV_10 | Reactor cooling water flow valve |
| XMV_11 | Condenser cooling water flow valve |
| XMV_12 | Agitator speed |

# Appendix C


Xmeas and Xmv Variables by Sample for Fault Free SimulationRun = 1
xmeas_1
xmeas_2
xmeas_3
xmeas_4
xmeas_5
xmeas_6
xmeas_7
xmeas_8
xmeas_9
xmeas_10
xmeas_11
xmeas_12
xmeas_13
xmeas_14
xmeas_15
xmeas_16
xmeas_17
xmeas_18
xmeas_19
xmeas_20
xmeas_21
xmeas_22
xmeas_23
xmeas_24
xmeas_25
xmeas_26
xmeas_27
xmeas_28
xmeas_29
xmeas_30
xmeas_31
xmeas_32
xmeas_33
xmeas_34
xmeas_35
xmeas_36
xmeas_37
xmeas_38
xmeas_39
xmeas_40
xmeas_41
xmv_1
xmv_2
xmv_3
xmv_4
xmv_5
xmv_6
xmv_7
xmv_8
xmv_9
xmv_10
xmv_11
Value
Sample