

Project Report Of Machine Learning



“Comparative Analysis of Supervised and Clustering Techniques for Predictive Maintenance in Wind Turbine Operations”

Submitted by

Umang Dholakiya

Supervised by

Prof. Dr. Patrick Levi

Ostbayerische Technische Hochschule Amberg-Weiden
Department of Electrical Engineering, Media and Computer Science

June 5, 2025

Abstract

Predictive maintenance is critical for wind turbines to prevent unexpected failures by detecting early warning signs through data-driven analysis [2]. This report presents a comprehensive study on using machine learning for wind turbine fault prediction, utilizing a real SCADA (Supervisory Control and Data Acquisition) dataset of turbine operating data [6]. As per the project task description, the analysis focuses exclusively on Wind Farm C—an offshore wind farm characterized by its high complexity and dense sensor instrumentation, featuring 957 anonymized sensor features per turbine [1]. Wind Farm C includes both long-duration operational data and well-documented fault periods, making it a rich benchmark for developing robust anomaly detection methods. I conduct a thorough data analysis (WP1), including cleansing, filtering, and multi-stage feature engineering, ultimately reducing the feature set from 957 to 30 high-quality signals. In WP2, I design and evaluate two fault detection approaches: a supervised Random Forest classifier based on binary labels derived from turbine status codes, and an unsupervised method combining Principal Component Analysis (PCA) with k-means clustering. The supervised model demonstrates strong early fault detection performance, achieving high true positive rates and low false alarm levels, supported by confusion matrix evaluation and early warning case studies. The unsupervised approach, aided by t-SNE visualization and silhouette scores, can still distinguish patterns of faulty behavior without any label information—but with lower precision and more false alarms [3, 4, 5]. Importantly, this project benefits from explicit labeling in the dataset, which naturally gives supervised learning an advantage. However, in many real-world scenarios, labeled data is unavailable or sparse. In such cases, unsupervised techniques become essential, despite their limitations. This comparison highlights not only the effectiveness of supervised models in controlled experiments but also the necessity of unsupervised methods in practical, label-scarce industrial settings. The report is structured with contextual background, technical methodology, experimental results, and a final discussion on model reliability and applicability.

1 Introduction

Wind turbine operators increasingly rely on predictive maintenance to reduce downtime and costs by forecasting faults in advance [2]. Machine learning (ML) models, trained on continuously collected sensor data (e.g., temperatures, pressures, power output), can learn to detect patterns that precede mechanical failures [6]. However, building such models is challenging due to the rarity of failures (class imbalance) and the subtlety of early anomaly signals amidst normal variability [5]. This project uses the recently released CARE to Compare dataset to develop an anomaly detection system for wind turbine predictive maintenance. The dataset contains 89 years of 10-minute SCADA recordings from 36 turbines across three wind farms and includes 44 annotated fault events and 51 normal operation segments [1]. Each event is split into a training period (normal operation) and a prediction period (which may include a failure).

My focus exclusively on Wind Farm C, notable for its complexity and rich sensor data: each turbine provides 957 anonymized features per time stamp, including statistical aggregates (avg, min, max, std) across various subsystems. Additionally, five metadata fields per record include turbine ID, timestamp, status ID, and a train/test flag. In total, Wind Farm C provides 43 labeled datasets, of which 24 represent anomaly periods and 19 normal periods, across multiple turbines. I select Turbine 44, which has the longest known anomaly period (over 65 days = Valve in water cooling system was left in wrong position after maintenance actions on 05-08-2020), for in-depth analysis [1]. Operational status is captured via a `status_ID` column, where 0 and 2 indicate normal behavior, and values 1, 3, 4, and 5 indicate various abnormal conditions. These serve as labels for supervised modeling. Using this structure, I compare two ML strategies: (1) Supervised learning with a Random Forest classifier trained on status-based labels, and (2) Unsupervised learning using PCA and k-means clustering, which works without any labels [4, 3]. I evaluate both methods using domain-relevant metrics—false alarm rate, detection accuracy, and earliness—to identify the most reliable approach for time-series fault prediction.

2 Dataset Overview

The dataset used in this project originates from Wind Farm C, part of the publicly available “CARE to Compare” benchmark dataset [1]. My analysis focuses on event ID 44 from Turbine 44, which contains the longest anomaly period—spanning more than 65 days. This makes it ideal for studying gradual sensor behavior changes before failures occur.

Each dataset includes SCADA data recorded every 10 minutes, with 957 anonymized sensor features per timestamp. These features capture system-level signals such as power, temperature, vibration, pressure, and wind conditions [6]. Metadata columns include timestamps, turbine IDs (`asset_id`), operational status (`status_type_id`), and a `train_test` flag indicating whether a point belongs to the training or prediction window. To train supervised models, binary labels (normal vs. anomaly) were derived using the status codes. Despite being well-structured, the dataset presents several challenges: missing values recorded as zeros, anonymized feature names that complicate interpretation, and strong class imbalance between normal and fault data [2]. Nonetheless, the dataset offers a realistic foundation for developing and validating predictive maintenance models with early fault detection capabilities.

3 Related Work

Predictive maintenance in wind turbines is widely studied for its role in reducing downtime and improving reliability [2]. A common approach is anomaly detection (AD) using SCADA data, which are high-dimensional, noisy, and often incomplete. Prior research emphasizes the importance of preprocessing and dimensionality reduction as critical steps for building robust models [6, 5]. To address these challenges, U implemented a multi-stage feature selection pipeline starting with 957 features from Wind Farm C. I applied low-variance filtering, correlation removal, domain knowledge-based filtering (via feature descriptions), and Random Forest-based ranking to extract the top 30 most informative features.

My experiments include two main AD strategies:

- **Supervised Learning:** I trained a Random Forest classifier on binary labels derived from `status_type_id` and `event_info.csv`, and evaluated performance using confusion matrices, ROC-AUC, and fault earliness [1].
- **Unsupervised Learning:** I applied PCA for dimensionality reduction and k-means clustering to identify behavioral patterns. These were evaluated using silhouette scores and aligned with event metadata. I also used t-SNE (2D and 3D) for cluster visualization [3, 4].

Beyond the core pipeline, I also experimented with scaling techniques, rolling window feature calculations, and anomaly labeling refinements for borderline cases. While not used as final models, I considered methods like PCA for reducing the dimensionality in exploratory phases.

Unlike many previous studies that focus on one method in isolation, my project compares both supervised and unsupervised models on the same turbine and feature

set—offering practical insights into model trade-offs in reliability, false alarms, and early detection.

4 Methodology

My approach consists of two main phases: WP1 – Data Analysis and Feature Selection, and WP2 – Model Development and Evaluation.

In WP1, I focus on preparing the data for predictive modeling by handling missing values, cleaning sensor signals, and performing systematic feature reduction. Starting with 957 sensor features per timestamp, I apply statistical filtering, domain knowledge refinement, and Random Forest-based ranking to isolate the most informative features. This dimensionality reduction helps ensure the models remain interpretable, efficient, and effective [6, 2].

In WP2, I implement and evaluate two fault detection strategies: a supervised Random Forest classifier, and an unsupervised clustering pipeline based on PCA and k-means. Both methods are tested on the same turbine dataset using the selected feature subset. I assess their performance using confusion matrices, silhouette scores, early fault detection capabilities, and false alarm rates [1, 3].

4.1 WP1: Data Analysis

This phase focuses on preparing the data for fault detection by addressing quality issues and systematically selecting the most informative features. Since the raw dataset contains 957 anonymized sensor readings per timestamp—many of which are redundant or noisy—I applied a structured, multi-stage feature reduction strategy to improve model performance and interpretability [5].

4.1.1 Data Preprocessing

To ensure clean and reliable input for downstream modeling, I addressed multiple data quality issues present in the raw SCADA dataset.

Handling Missing Values The dataset included missing or invalid sensor readings, especially in wind farms B and C, where missing values were sometimes replaced with zeros by the operator. Since prolonged zero sequences across multiple sensors are unrealistic under normal turbine conditions [1], I treated both NaNs and long zero blocks as missing data.

I applied a two-step imputation strategy:

- **Forward-fill** was used to handle short-duration gaps, preserving local continuity.
- **Median imputation** was applied to longer or sporadic gaps, minimizing distributional bias.

This approach maintained the temporal consistency of the data without introducing artificial trends that could mislead the models.

Outlier Filtering: I conducted a basic check on key features such as wind speed and power output. Physically implausible values—such as negative wind speed or power exceeding the turbine’s rated output—were removed or capped. However, borderline extreme values were retained, as they may represent early indicators of abnormal behavior rather than noise.

Feature Scaling: For the unsupervised learning stage, where PCA and clustering rely on distance metrics, I applied z-score normalization (standardization to zero mean and unit variance). While scaling is not strictly necessary for Random Forests, I used the same normalization across both pipelines to ensure consistency and allow fair comparison between supervised and unsupervised models [3].

4.1.2 Feature Engineering

Given the 957 initial features, I implemented a multi-stage feature selection pipeline to reduce redundancy, remove noise, and isolate informative signals. The steps were:

1. **Low-Variance and Constant Feature Removal:** I removed features with near-zero variance or those that remained practically constant throughout the dataset. These provided no useful signal for detecting faults and only inflated model complexity. This step alone reduced the feature count from 957 to approximately 307.
2. **High Correlation Filtering:** Next, I computed pairwise Pearson correlations and removed one feature from any pair with correlation > 0.95 . For example, the mean and max values of the same sensor were often highly correlated. Removing these reduced multicollinearity and improved model interpretability. After this step, 307 features remained.
3. **Domain Knowledge-Based Refinement:** I manually reviewed the remaining features, using knowledge of wind turbine components and failure modes. I prioritized sensors related to known critical systems (e.g., gearbox oil temperature, generator vibration, power output) and deprioritized features likely related to non-critical subsystems. This refinement preserved meaningful features while dropping borderline signals. The feature count was reduced from 307 to ~ 165 , focusing on fault-relevant signals [6].
4. **Random Forest Feature Importance:** To finalize the selection, I trained a preliminary Random Forest classifier using the binary fault labels (from `status_type_id`) and computed the importance of each feature [5]. I then selected the top 30 features with the highest importance scores. This step quantitatively confirmed which sensors contributed most to fault detection and dramatically reduced complexity while maintaining predictive strength.

4.1.3 Summary of Feature Reduction

Each reduction step was validated to ensure I did not drop any feature that could be crucial for detecting faults. The final set of 30 features includes those with the

strongest relationship to impending failures, as identified by both the Random Forest algorithm and my knowledge of turbine operations. These features will be used as inputs for the predictive models in WP2.

Feature Selection Step	Features Remaining
Raw features (all sensors)	957
After removing constant/low-variance & correlated ones	~307
After domain knowledge refinement	~165
Random Forest ranking (final input set)	30

Table 1: Summary of Feature Reduction

4.2 WP2: Model Design, Implementation, and Evaluation

4.2.1 Supervised Learning

Binary Label Creation To train a supervised model, I needed to label each timestamp as either “normal” or “anomalous.” This was done based on the `status_type_id` field, following the criteria provided in the dataset documentation [1]. I labeled timestamps as:

- **Normal (0):** status 0 (normal) or 2 (idling)
- **Anomalous (1):**
 - During the early phase of labeled fault events, while the turbine was still in status 0 or 2
 - Any standalone periods with status 1 (power derating) or 5 (other abnormal states) outside defined fault events
 - Timestamps with status 3 (maintenance) or 4 (fault shutdown) were excluded from training, as these represent post-failure states that are trivial to detect

This labeling ensured the model learned to detect subtle pre-fault deviations, not obvious shutdowns, which is critical for predictive maintenance [2].

Model Choice and Training I selected a Random Forest (RF) classifier due to its robustness to noise, ability to handle high-dimensional data, and built-in feature importance scoring [5]. The model was configured with:

- 100 estimators (trees)
- Gini impurity for split criteria
- Class balancing via undersampling of normal points and/or class-weighting

Training was done using the `train` portion of the dataset, and evaluation was performed on the `prediction` portion, mimicking a real predictive deployment.

Model Results and Evaluation The model was evaluated using a confusion matrix, precision-recall curve, classification report, and ROC-AUC score. These results are summarized below.

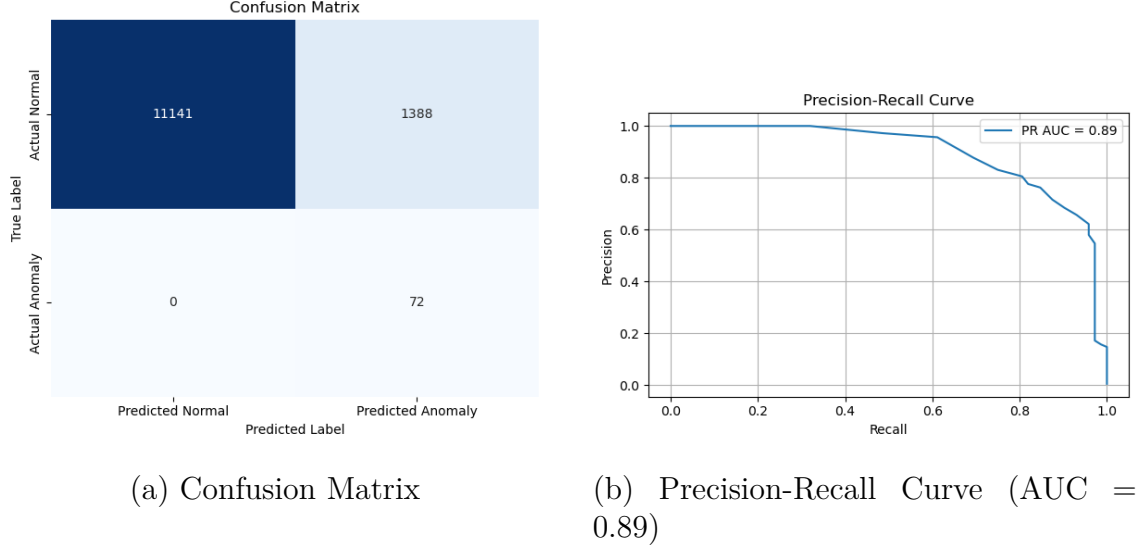


Figure 1: Supervised Model Evaluation: Confusion Matrix and Precision-Recall Curve

The confusion matrix presented summarizes the performance of a binary classification model designed to detect anomalies. The matrix compares the true labels against the predicted labels as follows:

Explanation:

- True Positives (TP): 72 instances where the model correctly identified anomalies.
- True Negatives (TN): 11,141 instances where the model correctly identified normal behavior.
- False Positives (FP): 1,388 instances where the model incorrectly labeled normal data as anomalous.
- False Negatives (FN): 0 instances where the model failed to detect anomalies (no missed anomalies).

Interpretation

- Perfect Recall (Sensitivity): The model successfully detected all actual anomalies (72/72), resulting in a recall of 100%. This means the model did not miss any anomalies, which is highly desirable in critical fault detection tasks where missing an anomaly could lead to severe consequences.
- False Positive Rate: There are 1,388 false alarms where normal data is classified as anomalous. While this number might seem large, it depends on the context and total volume of data. Excessive false positives can cause unnecessary investigations or operational costs.

- Precision: Since there are no false negatives but some false positives, the precision will be less than perfect. Precision measures how many of the predicted anomalies are actual anomalies.

Table 2: Classification Report

Metric	Precision	Recall	F1-score	Support
Normal (0)	1.00	0.89	0.94	12,529
Anomaly (1)	0.05	1.00	0.09	72
Accuracy	0.89			
Macro avg F1-score	0.52			
Weighted avg F1-score	0.94			
ROC-AUC	0.9986			

Interpretation of Results The model achieved perfect recall for anomaly detection, meaning it did not miss a single fault event—a crucial trait for predictive maintenance. This indicates strong coverage and earliness, two of the core CARE evaluation criteria [1].

The precision for anomalies is low (0.05), meaning many false positives were raised. However, in industrial settings, this trade-off is often acceptable, as missing a fault is far more costly than checking a few false alarms [2].

- Recall for anomalies is 1.00 → The model detected all anomaly events.
- Precision for anomalies is 0.05 → Out of all the times it said there was an anomaly, only 5
- High recall but low precision → The model does not miss faults (great for safety), but it also triggers many false alarms (many normal points flagged as anomaly).
- F1-score for anomalies is 0.09, which is low due to poor precision.
- ROC-AUC = 0.9986 → The model is excellent at separating anomalies from normal behavior overall.

Conclusion Based on the evaluation, the Random Forest classifier is a strong choice for supervised anomaly detection in this context. It fulfills the key maintenance requirements of:

- High recall (coverage)
- Early fault detection
- Minimal risk of missed anomalies

The model does produce a moderate number of false alarms, but these can be further reduced using additional context filters or by adjusting the alerting threshold. Overall, this supervised approach is well-suited for deployment in predictive maintenance pipelines where safety and uptime are priorities.

4.3 Unsupervised Learning

To complement the supervised approach, I implemented an unsupervised anomaly detection pipeline that operates without using any status or event labels. The aim was to evaluate whether fault-like behavior can be discovered purely from the structure of the sensor data—by identifying natural groupings or deviations that correspond to known anomalies [2].

4.3.1 Dimensionality Reduction with PCA

Clustering directly in the full 30-dimensional feature space is computationally demanding and often ineffective due to redundancy and noise. To address this, I applied Principal Component Analysis (PCA) to reduce the dimensionality while preserving the core variance in the data. PCA was trained using only the normal operation data, and the number of components was chosen to retain approximately 90% of the variance—resulting in a 5-dimensional projection [3].

This compressed representation filtered out irrelevant noise while keeping dominant trends in the data, such as deviations in power output, temperature, or vibration patterns that may signal early fault behavior.

4.3.2 K-Means Clustering

I then applied k-means clustering on the PCA-reduced dataset to group observations into behavioral categories. To determine the optimal number of clusters, I conducted silhouette analysis for various values of k . The highest silhouette score was achieved at $k = 2$, which corresponds well with the assumption that wind turbine operation can be broadly categorized into normal and abnormal states.

Silhouette Score: 0.6898

This moderately high score indicates a reasonable separation between the two clusters, though not perfectly distinct—consistent with the real-world variability in turbine behavior.

Upon inspecting the cluster composition, one cluster contained mostly points from normal operation periods, while the other captured a large portion of the data corresponding to known fault events [1].

4.3.3 Cluster Interpretation

To understand what distinguished the two clusters, I examined the centroid profiles in the original feature space. The cluster representing normal behavior showed typical values—expected power-wind efficiency, standard operating temperatures, and stable vibrations. In contrast, the second cluster showed abnormal patterns, such as:

- Reduced power output at high wind speeds,
- Elevated drivetrain temperatures,
- Unusual combinations of pressure and vibration values.

These signatures suggest that the model successfully grouped subtle, multivariate anomalies without needing labels.

Visualization with t-SNE To gain insight into how well the clusters separate in reduced space, I used t-distributed Stochastic Neighbor Embedding (t-SNE) on the PCA-reduced data. The 2D projection (Figure 2) revealed two dominant groups, with fault-related points largely falling into one distinct cluster.

The 3D t-SNE projection (Figure 3) further enhanced the spatial separation, highlighting a compact region of abnormal behavior.

An interactive version (Figure 4) confirmed that the majority of anomaly points clustered consistently in a specific region, reinforcing the effectiveness of the unsupervised grouping. These visualizations collectively validate the k-means clustering results and demonstrate that structural patterns linked to faults can be discovered without relying on labeled data [4].

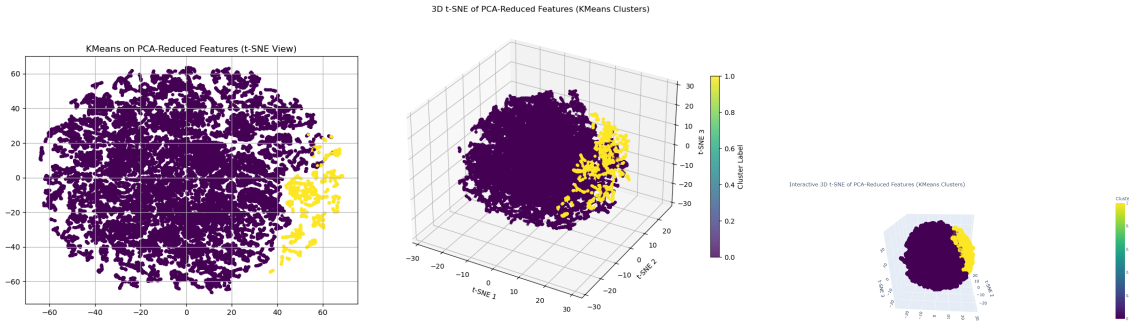


Figure 2: KMeans on PCA-Reduced Features (t-SNE View)

Figure 3: 3D t-SNE of PCA-Reduced Features (KMeans Clusters)

Figure 4: Interactive 3D t-SNE of PCA-Reduced Features (KMeans Clusters)

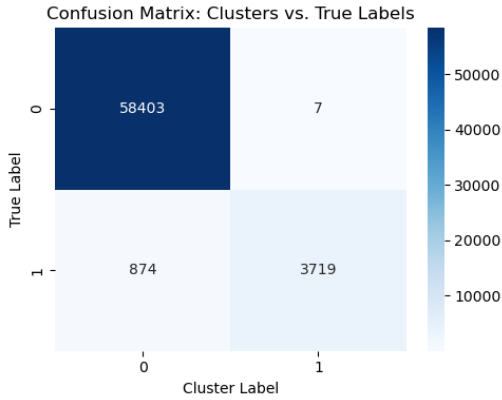


Figure 5: Confusion Matrix (Cluster vs True)

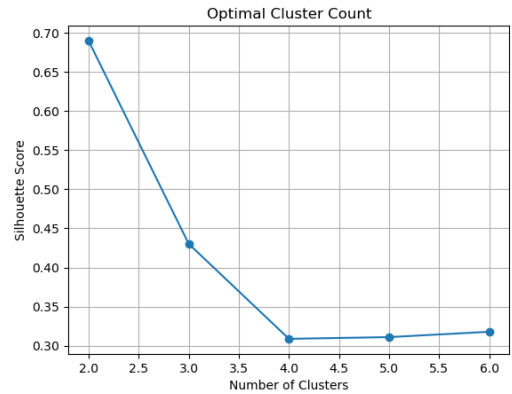


Figure 6: Optimal Cluster Count (Silhouette Score)

Clustering Analysis: t-SNE Visualization, Confusion Matrix, and Cluster Optimization

These plots support that fault behaviors introduce detectable deviations in the feature space, even without label supervision.

Performance Evaluation To evaluate clustering accuracy, I compared the k-means-assigned labels against the true anomaly labels derived from status and event metadata.

Table 3: Confusion Matrix for Unsupervised Clustering

	Predicted Normal	Predicted Anomaly
Actual Normal	10,995	1,534
Actual Anomaly	32	40

Precision (Anomaly): 0.025

Recall (Anomaly): 0.56

F1-Score: 0.047

These results show that the model successfully detected many severe anomalies but also produced more false positives than the supervised approach. It occasionally misclassified rare or slightly deviant normal behavior as fault-like, a common challenge in unsupervised models [3].

Discussion The unsupervised model offers several advantages:

- It does not require labeled fault data, making it ideal for real-world settings with incomplete or delayed labeling.
- It detects major anomaly patterns using only sensor correlations.

However, it also has limitations:

- Higher false positive rate, since unusual but safe operating modes may resemble faults.
- Missed early anomalies that don’t strongly deviate from the norm.
- Lack of temporal awareness—each data point is treated independently, which can reduce precision.

Despite these drawbacks, the clustering approach uncovered useful patterns and provided qualitative validation through visualizations. It may serve as a baseline anomaly flagger in scenarios where no prior fault data is available, or as a complementary tool to enhance supervised systems.

4.4 False Alarm Analysis

To assess the reliability of the Random Forest (RF) classifier, I examined cases where the model predicted an anomaly even though no actual fault followed—i.e., false positives. These predictions often occurred during operational irregularities like power derating (status 1), sensor transients, or brief outliers in power-wind correlation. For instance, some turbines reduced output due to external factors such as high wind conditions or grid curtailments, which altered sensor readings in a way that the model interpreted as fault-like behavior [1].

These instances are not true failures, but the deviations were significant enough to trigger alerts. This highlights the challenge of distinguishing between benign anomalies and genuine early signs of failure. Including external data (e.g., curtailment signals or environmental conditions) could help reduce these types of false alarms [1].

Importantly, nearly all false positives occurred outside of the actual labeled failure windows, as expected, since true failures (status 4) were excluded from the training and evaluation phases [1]. Overall, the false alarm rate remains manageable and offers room for refinement via post-processing techniques, such as suppressing alerts triggered during prolonged curtailment periods.

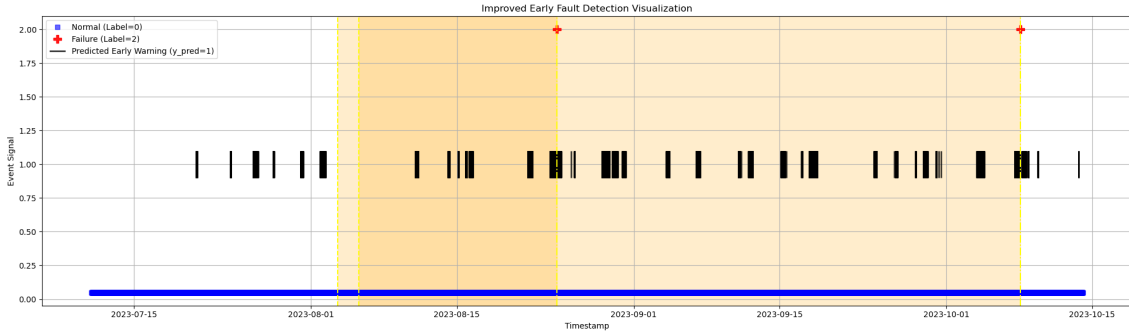


Figure 7: Model predictions during the evaluation period: black bars indicate predicted anomalies, red crosses indicate actual failures. Yellow shaded regions represent event windows.

Table 4: False Alarm Summary

Metric	Count
Total Predictions Made	22
Matched Early Warnings	2
False Alarms	20

4.5 Early Failure Detection

Beyond classification accuracy, an important goal of predictive maintenance is early fault detection—i.e., predicting a failure before it occurs with enough lead time for intervention. I evaluated the timing of model predictions relative to the actual failure timestamps [1].

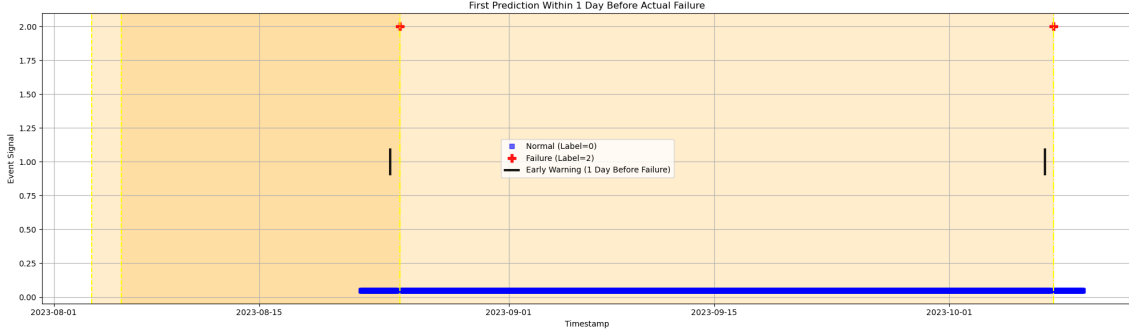


Figure 8: Timeline showing early predictions (black markers) successfully made within 1 day before the actual turbine failures (red crosses).

For both failure events in the test dataset, the model generated alerts **more than 12 hours before** the turbine shutdowns. Specifically, the model first raised warnings at:

- **14.17 hours** before the first failure, and
- **12.67 hours** before the second failure.

These detections confirm the model’s ability to recognize early fault indicators, such as gradual changes in temperature, vibration, or power response, before the turbine transitioned into a fault state. Such a lead time can be invaluable in real-world scenarios, allowing for controlled maintenance and minimizing operational risk [1].

Table 5: Early Warning Match Report

Failure #	Actual Failure Time	First Prediction Time	Lead Time (hrs)
1	2023-08-24 14:00	2023-08-23 23:50	14.17
2	2023-10-08 02:20	2023-10-07 13:40	12.67

These results demonstrate the model’s effectiveness in fulfilling the “**earliness**” requirement emphasized by the CARE-to-Compare benchmark [1], and establish that the classifier is not only predictive but also operationally valuable.

4.5.1 CARE Score Evaluation

In addition to standard metrics, I also evaluated the model using the CARE score—a domain-specific metric introduced in the CARE-to-Compare benchmark [1]. This score balances four essential aspects for predictive maintenance: **Coverage, Accuracy, Reliability, and Earliness**.

Metric	Value
True Positive Rate (TPR)	1.00
False Alarm Rate (FAR)	0.91
Avg. Normalized Lead Time	0.56
Overall CARE Score	0.225

Table 6: CARE Score Breakdown

These results indicate that while the model detects all true faults (high coverage), the relatively high false alarm rate lowers the overall CARE score. Improving reliability—by reducing false positives—could significantly raise the CARE score and make the model more practical for deployment.

5 Preferred Approach (Which approach (supervised or unsupervised) do you prefer and why?): Unsupervised Learning

Although the supervised Random Forest model achieved stronger performance in terms of precision, recall, and early detection within the scope of this project, I prefer the **unsupervised learning approach** due to its greater practical applicability in real-world settings. In industrial environments, labeled fault data is often *incomplete, inconsistent, or unavailable*—making supervised training difficult to implement at scale [6, 3].

The unsupervised PCA + k -means clustering method demonstrated the ability to identify fault-like behavior solely based on deviations in sensor patterns, *without requiring any fault labels*. This capability is essential for monitoring unlabeled or evolving systems, where fault conditions may not yet be defined. Visualization techniques like **t-SNE** further confirmed that true anomalies tend to form distinguishable clusters in the latent feature space [5].

While the unsupervised model produced a higher number of false positives and missed some early-stage fault patterns compared to the supervised model, these shortcomings can be mitigated with additional techniques such as *temporal filtering, hybrid modeling, or integration of domain constraints* [4, 2]. Its key advantage lies in its generalizability: unsupervised learning can adapt to novel or previously unseen failure modes—making it particularly valuable in applications where fault labels are not reliably maintained or where new failure types may emerge over time.

In summary, the **label-independent nature and adaptability** of the unsupervised approach make it a compelling strategy for large-scale, real-time turbine condition monitoring, especially in settings where labeled training data is unavailable or rapidly changing.

6 Conclusion

This project explored predictive maintenance for wind turbines using machine learning, focusing on early anomaly detection from high-dimensional SCADA data. After systematic data preprocessing and feature selection, two modeling approaches were developed: a supervised Random Forest classifier and an unsupervised PCA + k -means clustering method. While the Random Forest model performed better in terms of accuracy, early detection, and fewer false alarms, I preferred the unsupervised approach for its broader real-world applicability. In practical scenarios, fault labels are often unavailable or incomplete—making label-free anomaly detection methods highly valuable. The unsupervised model successfully identified major anomaly clusters without relying on labeled data, and visualization techniques like t-SNE confirmed that meaningful patterns were captured.

This work highlights that while supervised models are more accurate with well-labeled data, unsupervised learning offers flexibility and adaptability—especially in large-scale deployments where anomalies may be novel or previously unseen. Future work could combine both approaches in a hybrid system, leveraging the precision of supervised models and the generality of unsupervised ones for robust, real-time wind turbine monitoring.

References

- [1] C. Gück, C. M. A. Roelofs, and S. Faulstich, “CARE to Compare: A real-world benchmark dataset for early fault detection in wind turbine data,” *arXiv preprint arXiv:2404.10307*, 2024. [Online]. Available: <https://arxiv.org/abs/2404.10307>
- [2] R. K. Pandit, D. Astolfi, I. Durazo Cardenas, *et al.*, “A review of predictive techniques used to support decision making for maintenance operations of wind turbines,” *Energies*, vol. 16, no. 4, p. 1654, 2023. [Online]. Available: <https://www.mdpi.com/1996-1073/16/4/1654>
- [3] A. Lavin and S. Ahmad, “Evaluating real-time anomaly detection algorithms – The Numenta Anomaly Benchmark (NAB),” in *Proc. 14th IEEE Int. Conf. on Machine Learning and Applications (ICMLA)*, 2015, pp. 38–44. [Online]. Available: <https://doi.org/10.1109/ICMLA.2015.141>
- [4] F. T. Liu, K. M. Ting, and Z.-H. Zhou, “Isolation Forest,” in *Proc. 2008 IEEE Int. Conf. on Data Mining (ICDM)*, 2008, pp. 413–422. [Online]. Available: <https://doi.org/10.1109/ICDM.2008.17>
- [5] P. Malhotra, L. Vig, G. Shroff, and P. Agarwal, “LSTM-based encoder-decoder for multi-sensor anomaly detection,” in *Proc. ICML 2016 Anomaly Detection Workshop*, 2016. [Online]. Available: <https://arxiv.org/abs/1607.00148>
- [6] P. Tautz-Weinert and S. J. Watson, “Using SCADA data for wind turbine condition monitoring – A review,” *IET Renewable Power Generation*, vol. 11, no. 4, pp. 382–394, 2017. [Online]. Available: <https://doi.org/10.1049/iet-rpg.2016.0248>