A Comparative Study of Machine Learning and Deep Learning Models for Predictive Maintenance in Manufacturing

J. Anand
Department of Artificial Intelligence, SRM Institute of Science and Technology
Chennai, India

***Abstract*—Driven by Industry 4.0, Predictive Maintenance (PdM) is central to manufacturing strategy, aiming to foresee equipment issues, cut downtime, optimize maintenance, and boost Overall Equipment Effectiveness (OEE). Artificial Intelligence, especially Machine Learning (ML) and Deep Learning (DL), is key to this shift, analyzing vast machine data for actionable insights. A major hurdle in AI-driven PdM is imbalanced datasets, where critical failures are rare compared to normal operation, potentially skewing models and reducing their ability to detect consequential failures. This paper presents an empirical study comparing diverse ML and DL models for PdM in a simulated, imbalanced manufacturing environment. Using a synthetic dataset of 10,000 records with eight operational parameters and six machine states (including "No Failure" and five failure types), we applied a robust data engineering pipeline, SMOTE, dynamic class weighting, and evaluated eleven ML algorithms (XGBoost, ExtraTrees, RandomForest, SVM) and several DL architectures (Advanced DNNs, LSTMs, Transformers), prioritizing imbalance-robust metrics, especially macro-averaged F1-score, to assess all failure types. Findings show XGBoost decisively outperformed other models, achieving a test F1-score of 97.29% and macro F1-score of 88.01%, demonstrating robust, balanced performance, whereas an Advanced DNN, despite a high validation F1-score of 99.01%, performed poorly on the imbalanced test set, with macro F1-score of only 31.51% and failing to detect most minority failures. This study underscores that for tabular, imbalanced manufacturing data, well-tuned ML algorithms like XGBoost can outperform complex DL models, offering superior effectiveness, interpretability, and computational efficiency, challenging the assumption of universal DL superiority and providing empirically grounded guidance for practitioners implementing practical, effective AI-powered PdM solutions with a nuanced, data-centric approach.**

***Index Terms*—component, formatting, style, styling, insert**

I. Introduction

A. Industry 4.0 and Predictive Maintenance

The advent of Industry 4.0, characterized by the integration of cyber-physical systems, the Internet of Things (IoT), and artificial intelligence (AI), is profoundly transforming the manufacturing sector into an interconnected data-rich ecosystem. This paradigm shift has elevated Predictive Maintenance (PdM) from a supplementary activity to a core strategic imperative. Traditional maintenance strategies—reactive, addressing failures post-occurrence, and preventive, adhering to fixed schedules—are increasingly inadequate for the demands of modern, highly competitive manufacturing. Reactive approaches often lead to catastrophic unplanned downtime and exorbitant repair costs, while preventive strategies can result in unnecessary maintenance, wasted resources, and potential for human-induced errors. PdM offers a transformative solution by enabling proactive, condition-based maintenance. Through continuous monitoring of equipment health via sensors and sophisticated analysis of operational data, PdM aims to predict the Remaining Useful Life (RUL) or detect early signs of impending failures, as illustrated in Figure 1. This foresight allows for optimal scheduling of maintenance interventions, thereby maximizing equipment uptime, extending asset lifespans, optimizing resource allocation, reducing spare part inventories, enhancing product quality, and improving overall operational safety and Overall Equipment Effectiveness (OEE).

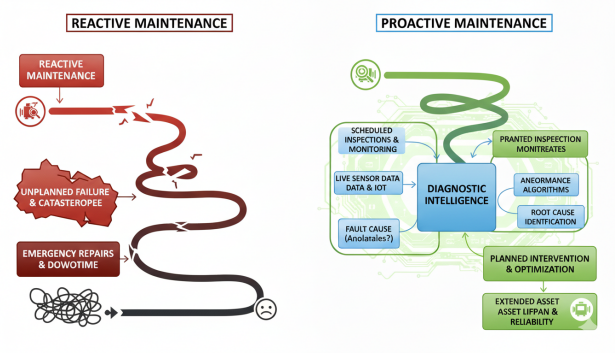

Fig. 1. Overview of Predictive Maintenance in Industry 4.0.

B. Artificial Intelligence for PdM

Artificial Intelligence, particularly Machine Learning (ML) and Deep Learning (DL), serves as the technological cornerstone of contemporary PdM systems, empowering them to extract actionable insights from vast and complex datasets. These algorithms excel at identifying intricate, non-linear patterns and relationships within historical operational data—such as temperature, pressure, tool wear, rotational speed, and torque—that often elude human operators or conventional statistical methods. Traditional ML algorithms, including Support Vector Machines (SVM), Random Forests, and Gradient Boosting Machines like XGBoost and LightGBM, have been widely and successfully applied to PdM tasks, especially with

structured, tabular data. These models are often preferred for their relative interpretability, robustness, and computational efficiency. Concurrently, DL models, such as Deep Neural Networks (DNNs), Convolutional Neural Networks (CNNs) for image-based inspection, and Recurrent Neural Networks (RNNs) like Long Short-Term Memory (LSTM) networks for sequential data, have gained significant traction. Their capacity for automatic feature learning from raw or minimally processed data makes them highly attractive for complex PdM applications. However, the selection of an optimal AI model is a non-trivial decision, contingent upon factors such as data nature and quality, failure mechanism complexity, required model interpretability, available computational resources, and specific operational objectives , as illustrated in Figure 2.

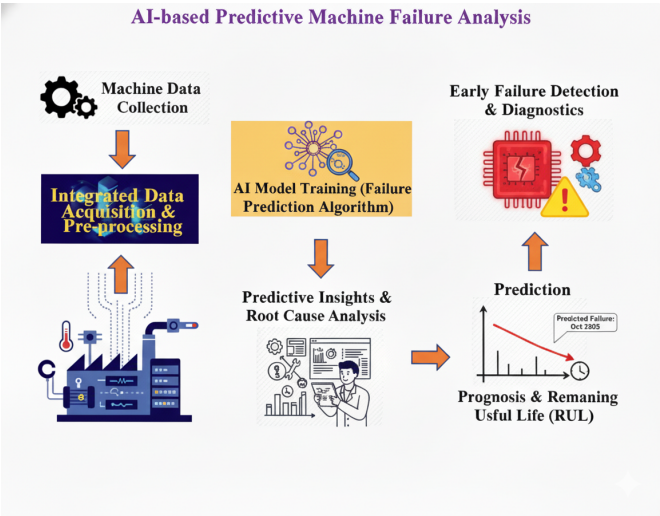

Fig. 2. AI-based Predictive Machine Failure Analysis.

C. Challenges with Imbalanced Data in PdM

A pervasive and particularly formidable challenge in many PdM contexts is the issue of highly imbalanced datasets. In manufacturing, critical equipment failures are often rare events compared to periods of normal operation, leading to datasets where the ”no failure” class heavily outweighs various ”failure” classes. This severe class imbalance can significantly bias the learning process of AI models, causing them to become overly specialized in predicting the majority class while performing poorly on the minority classes—the very failures that are most crucial to detect and prevent. Standard accuracy metrics become misleading in such scenarios, as a model can achieve high overall accuracy by simply predicting ”no failure” for all instances, rendering it practically useless for its intended purpose. Therefore, the adoption of more appropriate evaluation metrics, such as precision, recall, F1-score (especially the macro-averaged F1-score), and Area Under the Precision-Recall Curve (AUPRC), is essential for a realistic and meaningful assessment of model performance in imbalanced PdM tasks.

D. Research Objectives

This paper contributes to the ongoing discourse on effective PdM strategies by presenting a rigorous comparative study of a diverse range of ML and DL models applied to a synthetic PdM dataset representative of a typical manufacturing environment and characterized by severe class imbalance. Our primary objective is to evaluate and contrast the effectiveness of these models in accurately predicting multiple types of machine failures, with a particular emphasis on their ability to generalize to unseen, rare failure scenarios. We go beyond merely reporting overall accuracy by providing a detailed, class-wise performance breakdown for the top-performing models. The key research questions we address are:

1) How do various ML and DL models compare in their ability to predict different types of machine failures on a highly imbalanced tabular dataset?
2) What is the impact of data imbalance handling techniques, such as the Synthetic Minority Over-sampling Technique (SMOTE), on the performance of DL models compared to ML models?
3) Which model class offers the most practical and robust solution for real-world PdM deployment, considering factors like performance on minority classes, interpretability, and computational efficiency?

II. Literature Review / Related Work

The integration of Artificial Intelligence (AI), particularly Machine Learning (ML) and Deep Learning (DL), into Predictive Maintenance (PdM) frameworks has been widely explored in recent literature. Several studies have proposed diverse approaches to enhance predictive accuracy, interpretability, and robustness in industrial settings. The following works summarize key advancements, methodologies, and limitations.

1) **Kumar et al.**, “Artificial Intelligence-Driven Predictive Maintenance in Smart Manufacturing,” *IEEE Access*, 2023. This paper investigates AI’s role in minimizing equipment downtime through ML and DL models. It demonstrates AI’s effectiveness in predictive analytics but lacks an analysis of class imbalance and feature importance, which are crucial for industrial reliability.
2) **Yadav et al.**, “A Comparative Study of Machine Learning and Deep Learning Techniques for Predictive Maintenance,” *Springer Journal of Intelligent Manufacturing*, 2024. This study compares traditional ML methods with DL architectures for fault prediction. It concludes that both techniques achieve high accuracy; however, explainability and imbalance handling remain open challenges.
3) **Aminzadeh et al.**, “IoT-Based Predictive Maintenance for Industrial Compressors Using Machine Learning,” *IEEE Internet of Things Journal*, 2025. The paper integrates IoT sensors with ML algorithms for compressor fault prediction using multivariate time-series data. While effective in prediction, scalability and interpretability are not sufficiently addressed.

4) **Lin et al.**, "Explainable Artificial Intelligence for Predictive Maintenance in Nuclear Power Plants," *IEEE Access*, 2025. This study introduces explainable AI (XAI) methods to enhance transparency in PdM for high-risk domains. It demonstrates that model interpretability is essential for trust in AI-driven maintenance decisions.
5) **Kolokas et al.**, "Forecasting Faults of Industrial Equipment Using Machine Learning Classifiers," *IEEE Transactions on Industrial Informatics*, 2018. The authors present a real-time PdM methodology capable of identifying equipment stoppages and fault types using sensor data. The study explores several ML algorithms for predictive fault detection, although deep learning and interpretability aspects were not deeply examined.
6) **Li et al.**, "Deep Learning Architectures for Predictive Maintenance: LSTM and Transformer-Based Approaches," *IEEE Transactions on Neural Networks and Learning Systems*, 2025. The authors benchmark LSTM and Transformer architectures for time-series PdM tasks, demonstrating DL's ability to capture temporal dependencies. However, their models require large datasets and face computational scalability issues.

Summary: The reviewed works collectively highlight the evolution of predictive maintenance from traditional ML-based fault detection to explainable deep learning and hybrid approaches. While ML provides interpretability and robustness for structured data, DL excels at learning complex temporal patterns. However, persistent issues such as class imbalance, model explainability, and deployment scalability remain open challenges motivating the present research.

III. Dataset and Methodology

This section meticulously outlines the comprehensive methodology employed in our comparative study of Machine Learning (ML) and Deep Learning (DL) models for Predictive Maintenance (PdM) within a simulated manufacturing context. The overarching objective of this methodological framework is to ensure a rigorous, reproducible, and fair comparison of the selected models' capabilities in predicting machine failures, with a specific focus on their robustness in handling the pervasive challenge of highly imbalanced datasets. The methodology encompasses detailed descriptions of the dataset, the data preprocessing and engineering pipeline, the selection and architecture of ML and DL models, the training and hyperparameter optimization strategies, and the comprehensive suite of evaluation metrics used to gauge performance.

A. Dataset Description and Characteristics

The empirical foundation of this study is a synthetic predictive maintenance dataset, originally generated by the School of Engineering at the University of Applied Sciences in Berlin, Germany. This dataset was selected for its structural representation of a typical manufacturing environment, featuring well-defined operational parameters and distinct failure modes, making it a suitable benchmark for comparative analysis. The dataset comprises 10,000 individual records, each corresponding to a specific operational state of a machine. Each record includes eight critical operational parameters (features) that serve as input variables for the predictive models, and a target variable indicating the machine's state, which can be one of six classes: "No Failure" or one of five specific failure types ("Heat Dissipation Failure," "Overstrain Failure," "Power Failure," "Random Failure," or "Tool Wear Failure").

The eight input features are:

1) **Type**: Categorical product quality variant ('L', 'M', 'H'), one-hot encoded into three binary features.
2) **Air temperature [K]**: Ambient air temperature.
3) **Process temperature [K]**: Manufacturing process temperature.
4) **Rotational speed [rpm]**: Component rotational speed.
5) **Torque [Nm]**: Applied torque.
6) **Tool wear [min]**: Cumulative tool wear time.
7) **Product ID**: Unique product identifier, used to derive 'Type'.
8) **UID**: Unique record identifier, not used as a predictive feature.

A defining and central challenge of this dataset, and a critical focus of our methodology, is its severe class imbalance. The distribution of the target variable is heavily skewed towards the "No Failure" class , as illustrated in Figure 3. This characteristic mirrors real-world manufacturing scenarios where equipment operates under normal conditions for the vast majority of the time. The specific distribution is as follows:

- **No Failure**: 9,652 instances (96.5%)
- **Heat Dissipation Failure**: 112 instances (1.1%)
- **Overstrain Failure**: 78 instances (0.8%)
- **Power Failure**: 95 instances (0.9%)
- **Random Failure**: 18 instances (0.2%)
- **Tool Wear Failure**: 45 instances (0.4%)

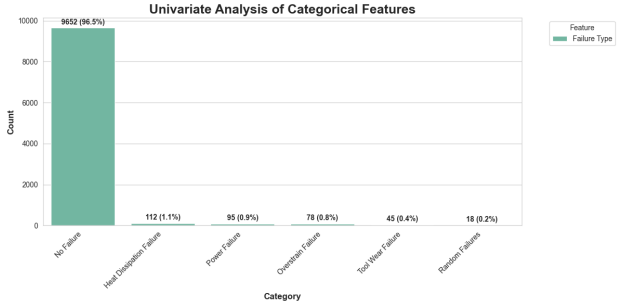

Fig. 3. Distribution of machine failure types in the dataset.

A critical aspect of understanding machine failures in this dataset is not only the overall class imbalance but also how different failure types are distributed across key operational features. Models trained on imbalanced data are prone to bias towards the majority class ("No Failure"), potentially achieving high overall accuracy while missing the rare but critical failure events. To gain deeper insight, we analyzed the distribution of failure types with respect to individual features.

For example, Figure 4 illustrates the distribution of different failure modes conditioned on *Air Temperature*, revealing how specific temperature ranges correlate with particular failure types. Similar conditional distributions were examined for *Process Temperature*, *Rotational Speed*, *Torque*, and *Product Type*, providing valuable insights for targeted predictive modeling.

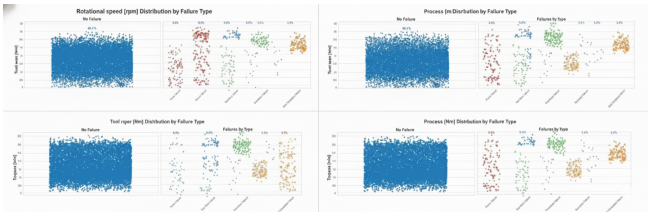

Fig. 4. Failure type distribution across operational features.

B. Data Preprocessing and Engineering Pipeline

A robust and systematic data preprocessing pipeline is essential for preparing the raw data for effective model training and ensuring the reliability of our comparative analysis. Our pipeline involved several key steps:

1) **Categorical Feature Encoding:** The 'Type' feature, being nominal in nature, was transformed using *Label Encoding*. Each category (L, M, H) was assigned a unique integer label (e.g., L=0, M=1, H=2), allowing the ML and DL algorithms to effectively interpret and utilize this categorical information without introducing additional binary features.
2) **Feature Scaling:** To ensure that all numerical features contribute equally to the model training process and to prevent features with larger scales from dominating those with smaller scales, the five continuous numerical features (Air temperature, Process temperature, Rotational speed, Torque, and Tool wear) were normalized using Min-Max scaling. The formula for Min-Max scaling is:
$$X_{\text{scaled}} = \frac{X - X_{\min}}{X_{\max} - X_{\min}},$$
where X is the original feature value, $X_{\min}$ is the minimum value of the feature in the training set, and $X_{\max}$ is the maximum value of the feature in the training set. This normalization was fit on the training data and then applied to both the training and test sets to prevent data leakage.
3) **Train-Test Split:** The preprocessed dataset was split into training and testing sets using a stratified sampling strategy, ensuring that the proportion of each failure type is maintained in both subsets. An 80/20 split was employed, resulting in 8,000 samples for training and 2,000 samples for testing.
4) **Handling Class Imbalance:**
   - **Synthetic Minority Over-sampling Technique (SMOTE):** SMOTE was applied exclusively to the training set to mitigate severe class imbalance. SMOTE generates synthetic samples for the minority classes by interpolating between existing minority samples. After applying SMOTE, each of the five failure types was oversampled to 2,500 samples, while the "No Failure" class retained 7,722 samples. This resulted in a training set where "No Failure" represented approximately 38.19% and each failure type represented approximately 12.36%. SMOTE was applied only to the training data, while the test set remained in its original, imbalanced state to provide an unbiased evaluation.
   - **Dynamic Class Weights:** In addition to SMOTE, dynamic class weights were incorporated into the training of certain models, particularly Deep Learning models. Higher misclassification costs were assigned to minority class samples, e.g., failure types received a weight of 1.348 compared to 0.436 for "No Failure". This encourages the model to focus on under-represented failure instances. For Machine Learning models such as XGBoost, the primary comparison was conducted on the original imbalanced training data, while Deep Learning models were trained on the SMOTE-augmented data, often combined with class weighting.

C. Models and Algorithms

The strategic selection of appropriate models and algorithms is crucial, as it directly influences the accuracy, interpretability, and practical applicability of a predictive maintenance (PdM) system. The range of available machine learning (ML) and deep learning (DL) techniques is broad, each possessing unique characteristics that make them suitable for specific data types, problem complexities, and operational constraints. Traditional ML models, including decision trees, support vector machines, and ensemble methods such as XGBoost and Random Forest, are widely used in predictive analytics due to their relative simplicity, efficiency on structured data, and often higher interpretability. These models generally require careful feature engineering and demonstrate robustness to certain data irregularities. In contrast, DL architectures—such as Deep Neural Networks (DNNs), Long Short-Term Memory (LSTM) networks, and Transformer-based models—have gained prominence for their ability to automatically extract complex features from raw or minimally processed data, capturing subtle, non-linear relationships that may elude conventional ML approaches. However, DL models often require larger datasets, higher computational resources, and present interpretability challenges, raising "black box" concerns in critical industrial settings where understanding prediction rationale is essential.

This comparative study systematically evaluates a curated ensemble of eleven ML models and three distinct DL architectures to examine their relative merits in the context of PdM. The ML cohort spans foundational models like Logistic Regression and k-Nearest Neighbors (KNN) to high-performance ensemble methods such as XGBoost, CatBoost, and ExtraTrees. The DL selection includes an Advanced DNN

with residual connections and batch normalization, a Tabular LSTM tailored to capture sequential dependencies in tabular data, and a Tabular Transformer leveraging self-attention mechanisms to weigh feature importance. This diverse set of models provides broad algorithmic coverage, enabling a nuanced understanding of their strengths and limitations when applied to the synthetic dataset, particularly in light of its inherent class imbalance. The following sections describe these models, their configurations, and the evaluation framework employed to assess their predictive performance across various machine failure modes.

TABLE I
LIST OF MODELS USED

Machine Learning Models	Deep Learning Models
XGBoost (Extreme Gradient Boosting)	Advanced DNN (Deep Neural Network)
ExtraTrees (Extremely Randomized Trees)	Tabular LSTM (Long Short-Term Memory)
RandomForest	Tabular Transformer
DecisionTree	
CatBoost	
MLP (Multi-Layer Perceptron)	
KNN (K-Nearest Neighbors)	
AdaBoost (Adaptive Boosting)	
SVM (Support Vector Machine)	
Naive Bayes	
Logistic Regression	

D. Machine Learning Models

In our comprehensive comparative study for Predictive Maintenance (PdM), a carefully selected ensemble of eleven Machine Learning (ML) algorithms forms a critical component of our analytical framework. These models, renowned for their efficacy in handling structured data and their diverse algorithmic underpinnings, were chosen to provide a broad spectrum of predictive capabilities. This selection ranges from foundational linear models to sophisticated ensemble techniques, each offering unique strengths in pattern recognition and classification tasks inherent to PdM. A defining characteristic of our methodology for these ML models was that they were primarily trained on the original, imbalanced training data. This decision was deliberate, aiming to evaluate their intrinsic ability to handle skewed class distributions without the immediate application of data-level balancing techniques like SMOTE, which were reserved for our Deep Learning counterparts. This approach allows us to assess the robustness of various ML algorithms in a scenario that closely mirrors many real-world PdM problems where failure instances are scarce. The performance of these models, therefore, provides a crucial baseline against which the effectiveness of more complex data augmentation strategies employed in DL models can be measured. The following list details these eleven ML models, providing a brief description of their core principles and their relevance to PdM:

- **XGBoost (Extreme Gradient Boosting):** XGBoost is an optimized, distributed, and highly efficient implementation of the gradient boosting framework. It builds an ensemble of decision trees sequentially, where each new tree corrects the errors of the previous one. XGBoost is known for its performance, scalability, and ability to handle complex non-linear relationships. Its regularization techniques help prevent overfitting, making it a popular choice in PdM for predicting equipment failures from historical sensor and operational data.
- **ExtraTrees (Extremely Randomized Trees):** ExtraTrees is an ensemble learning method similar to Random Forest but introduces more randomness in tree construction. Both features and split points are chosen randomly, which reduces variance and overfitting, offering robust performance on noisy PdM datasets.
- **RandomForest:** RandomForest builds multiple decision trees on bootstrapped samples and outputs the majority vote for classification. Random feature selection at each split helps decorrelate trees, improving generalization and making it widely used for fault diagnosis and failure prediction.
- **DecisionTree:** A fundamental, non-parametric model for classification and regression. It represents decisions and their consequences in a tree-like graph, with internal nodes as features and leaves as outcomes. Decision Trees are interpretable, providing insights into factors driving equipment failures.
- **CatBoost (Categorical Boosting):** CatBoost is a gradient boosting library that handles categorical features natively and includes mechanisms to reduce overfitting. It provides robust performance for PdM datasets with categorical operational variables.
- **MLP (Multi-Layer Perceptron):** An MLP is a feed-forward artificial neural network with input, hidden, and output layers. It can learn non-linear decision boundaries, serving as a baseline neural network for PdM by modeling complex relationships between sensor inputs and failure states.
- **KNN (K-Nearest Neighbors):** KNN classifies a new data point based on the majority class among its k nearest neighbors. Its simplicity and non-parametric nature can be advantageous for irregular decision boundaries in PdM, though performance is sensitive to irrelevant features and the choice of k.
- **AdaBoost (Adaptive Boosting):** AdaBoost sequentially trains weak learners, emphasizing misclassified instances from previous learners. The final model is a weighted sum of weak learners, enhancing classification accuracy of failure modes in PdM.
- **SVM (Support Vector Machine):** SVMs find an optimal hyperplane in high-dimensional space to separate classes. Kernel tricks allow handling non-linear data, making SVM effective for fault detection and diagnostics in PdM.
- **NaiveBayes:** NaiveBayes classifiers use Bayes' theorem assuming feature independence. Despite simplification,

they are effective baseline classifiers for estimating failure probability from sensor data.

- **LogisticRegression:** Logistic Regression is a linear model for binary or multi-class classification. It estimates the probability that an input belongs to a class using the logistic function. It provides a reliable baseline for PdM when feature-target relationships are approximately linear.

The training process for these ML models involved hyperparameter optimization using techniques such as GridSearchCV or RandomizedSearchCV. For example, for XGBoost, optimal parameters such as learning rate, max depth, number of estimators, and subsample ratio

E. Deep Learning Models

In parallel with the evaluation of Machine Learning models, this study incorporates a suite of sophisticated Deep Learning (DL) architectures. These models are capable of learning intricate representations from data and modeling highly complex, non-linear relationships. DL models with multiple layers can automatically discover hierarchical features, potentially capturing subtle patterns indicative of impending equipment failures that might be less apparent to traditional ML algorithms. A key methodological distinction in our approach was the use of the Synthetic Minority Over-sampling Technique (SMOTE) exclusively on the training set to address severe class imbalance. Additionally, dynamic class weights were often applied during training, assigning higher misclassification costs to minority failure classes. This dual strategy of data augmentation (SMOTE) and algorithmic adjustment (class weighting) was designed to mitigate bias towards the majority "No Failure" class and enhance sensitivity to rare failure events. The following DL architectures were evaluated:

1) **Advanced DNN (Deep Neural Network):** This architecture is a complex multi-layer perceptron (MLP) with multiple hidden layers designed to capture intricate patterns , as illustrated in Figure 7 in high-dimensional sensor data. Key features include:
   - *Batch Normalization (BatchNorm1d):* Normalizes activations after linear layers to accelerate training and improve stability.
   - *ReLU Activation Functions:* Introduce non-linearity and mitigate vanishing gradient issues.
   - *Dropout Layers:* Randomly deactivate neurons during training to prevent overfitting and improve generalization.
   - *Residual Connections:* Allow gradients to flow more easily through deep networks, enabling the training of very deep architectures without degradation.
   - *Adaptive Learning Rate:* Supports optimizers like Adam and RMSprop that adapt learning rates for faster convergence.
   - *Early Stopping:* Monitors validation loss to halt training when no improvement is observed, preventing overfitting.

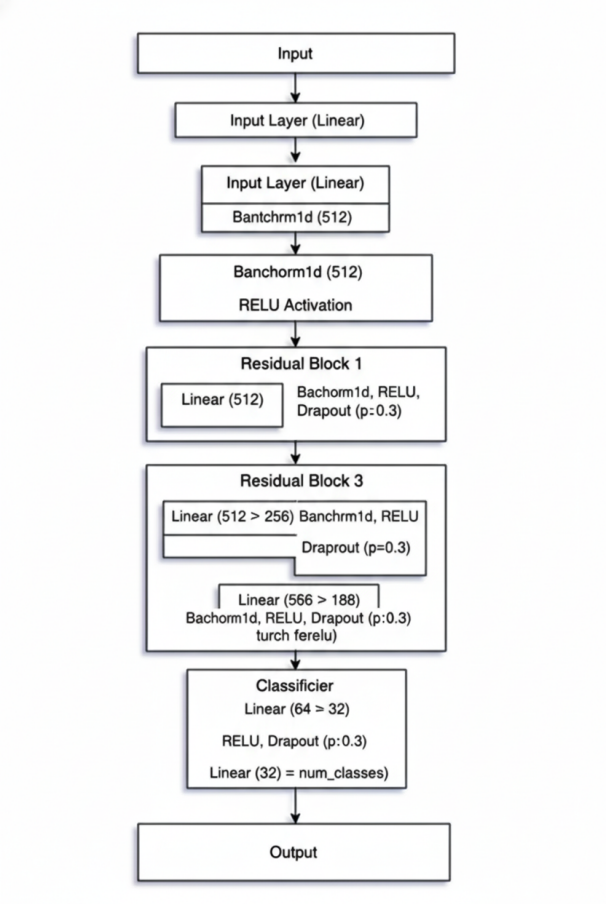

Fig. 5. Architecture Diagram of DNN

The architecture includes an input layer, several linear layers (e.g., 512, 256, 128) with BatchNorm1d and ReLU, residual blocks, and a final classifier block. Optimizers explored include Adam, SGD with Momentum, and RMSprop. DNNs are versatile for modeling complex relationships in sensor data, making them suitable for PdM applications.

2) **Tabular LSTM (Long Short-Term Memory):** Adapted from standard LSTMs to capture latent sequential dependencies in tabular data. Key components include:
   - *Feature Embedding (nn.Linear($1 \rightarrow 16$)):* Embeds individual features into higher-dimensional space.
   - *LSTM Layer:* Processes sequences of embedded features.
   - *Multi-Head Attention (nn.MultiheadAttention):* Assigns varying importance to different input features or sequence positions.
   - *Classifier:* Linear layers with ReLU activations producing final failure type predictions.

   Optimizers used include RMSprop and Adam. LSTMs are powerful for time-series and tabular data when temporal or sequential patterns are important for failure prediction.

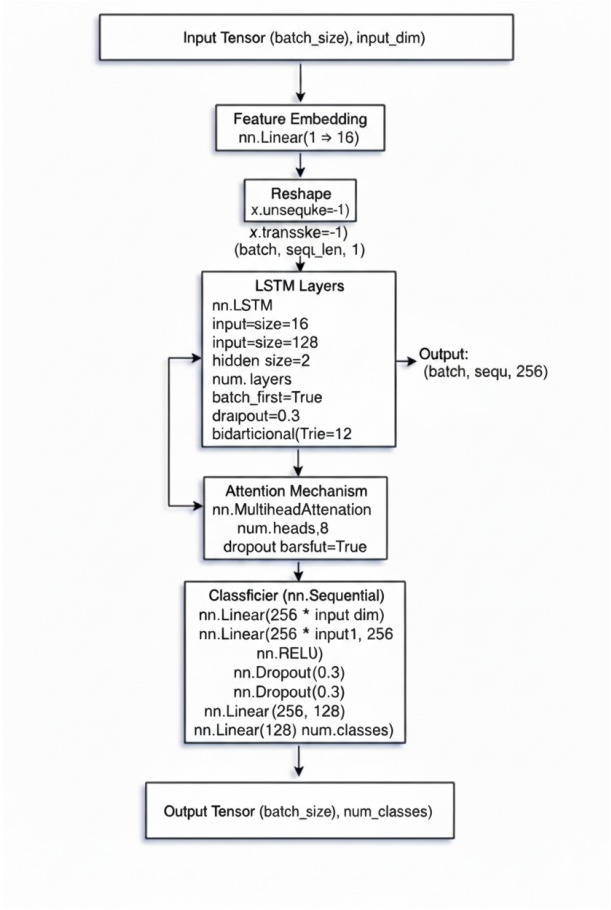

Fig. 6. Architecture Diagram of Tabular LSTM

3) **Tabular Transformer:** A Transformer-based architecture adapted for tabular data, leveraging self-attention to dynamically weigh feature importance. Key components include:
   - *Input Layer:* Receives preprocessed tabular data.
   - *1D Convolution with ReLU:* Extracts local patterns or feature interactions.
   - *Feature Projection & Positional Encoding:* Projects features to suitable dimensions and adds positional information.
   - *Stack of Transformer Encoder Layers:* Each layer has multi-head self-attention and feed-forward sub-layers.
   - *Global Average Pooling:* Aggregates encoder outputs into a fixed-length vector.
   - *Final Classifier:* Linear layers with GELU activation and dropout producing final predictions.

   Optimizers include Adam and SGD with Momentum. Transformers excel at modeling complex, non-linear interactions among features without relying on sequential assumptions like LSTMs, which is valuable in PdM scenarios.

The training of DL models involved monitoring performance on a validation set to guide model selection and prevent overfitting. Metrics like the validation F1-score were used to identify optimal configurations and optimizers. For example, the Advanced DNN trained with Adam achieved a high validation F1-score, demonstrating strong learning capability on the SMOTE-augmented data. Training progress, including loss and accuracy curves, was also visualized to ensure stable convergence. This comprehensive DL modeling strategy, combined with data handling for imbalanced datasets, allows a thorough evaluation of their potential in complex PdM scenarios.

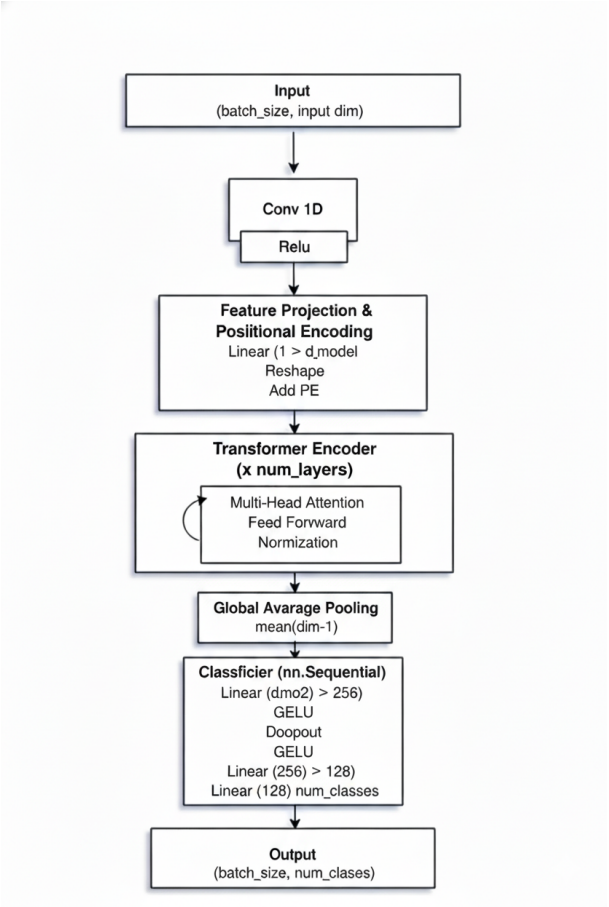

Fig. 7. Architecture Diagram of Tabular Transformer

IV. Results

This section presents the outcomes of our comparative study evaluating a spectrum of Machine Learning (ML) and Deep Learning (DL) models for Predictive Maintenance (PdM) in a simulated manufacturing environment. Performance metrics for the test set are reported on the original, imbalanced dataset (2,000 samples), while validation metrics pertain to DL model training on augmented data.

A. Machine Learning Model Performance

Eleven ML models were trained on the imbalanced training dataset (8,000 samples) and evaluated using F1-score. Table II summarizes the test set results.

TABLE II
ML MODEL PERFORMANCE LEADERBOARD (TEST F1-SCORE)

Model	F1-Score (Test Set)
XGBoost	0.9729
ExtraTrees	0.9677
RandomForest	0.9664
DecisionTree	0.9654
CatBoost	0.9601
MLP	0.9408
KNN	0.9257
AdaBoost	0.8930
SVM	0.8367
NaiveBayes	0.8269
LogisticRegression	0.6287

XGBoost achieved the highest F1-score, demonstrating strong performance on both majority and minority classes. Table III provides a detailed classification report for XGBoost.

TABLE III
XGBOOST DETAILED CLASSIFICATION REPORT (TEST SET)

Failure Type	Precision	Recall	F1-Score	Support
No Failure	1.0000	0.9963	0.9981	1930
Heat Dissipation	1.0000	0.9821	0.9910	56
Overstrain	0.9474	0.9231	0.9351	39
Power Failure	0.9691	0.9895	0.9792	48
Tool Wear	0.5833	0.7778	0.6667	9
Random Failure	0.6271	0.8222	0.7115	23
Macro Avg	0.8544	0.9151	0.8801	2000
Weighted Avg	0.9951	0.9943	0.9946	2000
Accuracy	-	-	0.9943	2000

B. Deep Learning Model Performance

DL models were trained on SMOTE-augmented data with dynamic class weights. Table **??** summarizes validation and test performance.

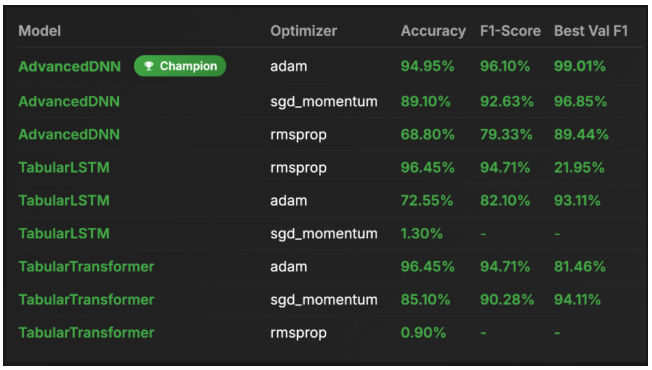

Fig. 8. Architecture Diagram of Tabular Transformer

The Advanced DNN (adam) achieved the highest validation F1-score (0.9901), but its macro F1 on the imbalanced test set dropped to 0.3151, failing to detect most minority classes (Table IV).

C. Comparative Analysis: XGBoost vs. Advanced DNN

Table V presents a head-to-head comparison of the best ML and DL models.

TABLE IV
ADVANCED DNN (ADAM) CLASSIFICATION REPORT (TEST SET)

Failure Type	Precision	Recall	F1-Score	Support
No Failure	0.9984	0.9741	0.9861	1928
Heat Dissipation	0.0000	0.0000	0.0000	21
Power Failure	0.0000	0.0000	0.0000	18
Overstrain	0.8261	1.0000	0.9048	19
Tool Wear	0.0000	0.0000	0.0000	12
Random Failures	0.0000	0.0000	0.0000	2
Macro Avg	0.3041	0.3290	0.3151	2000
Weighted Avg	0.9703	0.9485	0.9592	2000
Accuracy	-	-	0.9485	2000

TABLE V
HEAD-TO-HEAD COMPARISON: XGBOOST VS. ADVANCED DNN (ADAM)

Metric	XGBoost	Advanced DNN
Test F1-Score (Weighted)	0.9946	0.9592
Test Accuracy	0.9943	0.9485
Macro F1-Score	0.8801	0.3151
Best Validation F1-Score	0.9918	0.9901
Ability to Detect All Failures	Yes	No
Performance on Rarest Failures	Moderate to Good	Very Poor
Interpretability	High	Low
Training/Inference Speed	Fast	Slow

XGBoost demonstrated robust, balanced performance across all classes, outperforming the Advanced DNN in practical PdM applicability despite slightly lower validation F1.

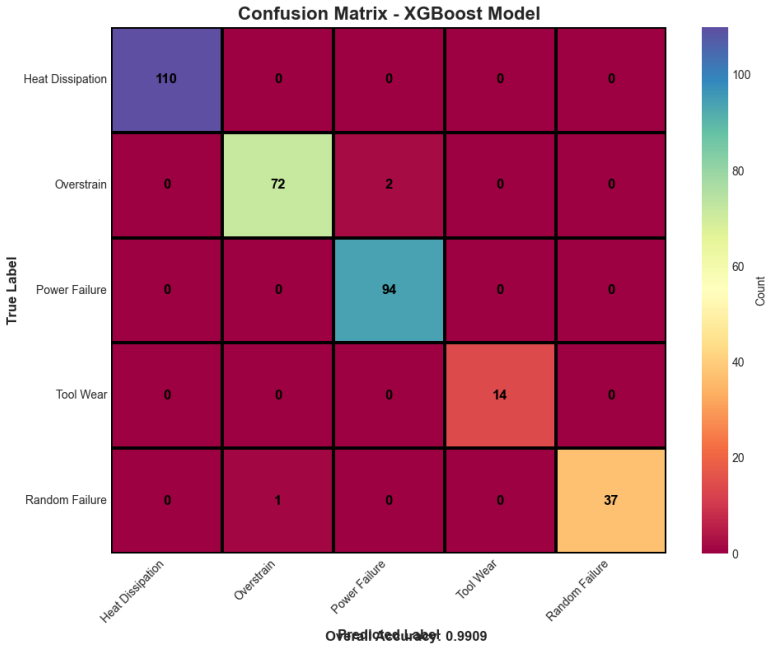

Fig. 9. Confusion matrix of the XGBoost model for predictive maintenance.

V. DISCUSSION

The results of our comprehensive comparative study yield profound insights into the practical application and relative efficacy of Machine Learning (ML) and Deep Learning (DL) models for Predictive Maintenance (PdM) within manufacturing environments characterized by imbalanced datasets. The

pivotal finding is that, for the specific synthetic PdM dataset under investigation, the XGBoost model, a representative of traditional ML algorithms, decisively outperformed the more complex Deep Learning models, including the Advanced Deep Neural Network (DNN) that exhibited promising results during the validation phase. This outcome underscores several critical considerations for the development, evaluation, and deployment of AI-powered PdM solutions, challenging prevailing assumptions and advocating for a pragmatic, data-centric approach.

A. The Pragmatic Superiority of XGBoost in this Context

XGBoost emerged as the unequivocal champion, achieving a remarkable test F1-score of 97.29% and, more significantly, a macro-averaged F1-score of 88.01%. This latter metric is particularly crucial as it reflects the model's balanced performance across all classes, including the rare failure types that are often the most critical to detect in a PdM scenario. The XGBoost model demonstrated a robust capability to identify all five failure types, with F1-scores ranging from a respectable 0.6667 for the rarest "Tool Wear Failure" to an outstanding 0.9910 for "Heat Dissipation Failure." This comprehensive failure detection capability is paramount for any practical PdM system.

The success of XGBoost can be attributed to several inherent characteristics of gradient boosting algorithms, especially when applied to structured, tabular data:

1) **Effective Handling of Imbalanced Data:** Sophisticated tree-based ensemble methods like XGBoost possess mechanisms that can effectively learn complex decision boundaries capable of isolating minority class instances. Techniques such as subsampling during tree construction and the iterative, error-correcting nature of boosting contribute to better performance on minority classes. Hyperparameter tuning, such as adjusting `scale_pos_weight`, can further enhance performance.
2) **Feature Importance and Inherent Interpretability:** XGBoost provides feature importance scores, offering valuable insights into which operational parameters are most predictive of specific failures. This interpretability helps domain experts understand the model's reasoning and build trust in predictions [**?**].
3) **Computational Efficiency:** XGBoost is generally faster to train and requires fewer computational resources compared to complex DL models, making it practical for deployment in resource-constrained environments.
4) **Robustness to Overfitting with Proper Tuning:** With appropriate hyperparameter tuning (`max_depth`, `subsample`, `learning_rate`, `lambda`, `alpha`), XGBoost can generalize effectively. The close agreement between its cross-validation F1-score (0.9918) and test F1-score (0.9729) demonstrates this robustness.

B. The Pitfalls of Deep Learning: Deceptive Validation Scores vs. Real-World Generalization

The Advanced DNN, optimized with the `adam` optimizer, trained on SMOTE-augmented data with dynamic class weights, and featuring a sophisticated architecture including residual blocks and batch normalization (see Figure **??**), achieved an impressive validation F1-score of 99.01%. However, its evaluation on the original, imbalanced test set revealed a macro-averaged F1-score of only 31.51%, failing to predict 4 out of 5 failure types.

Key issues include:

1) **Overfitting to Augmented Data:** While SMOTE addresses class imbalance, the DNN may overfit to synthetic patterns that do not generalize to the true distribution.
2) **Persistent Majority Class Bias:** Even with dynamic class weights, the model may prioritize overall accuracy, overlooking minority class detection.
3) **Model Complexity vs. Data Sufficiency:** DL models require diverse data; SMOTE increases quantity but not intrinsic information, limiting learning for rare failure types.
4) **Sensitivity to Hyperparameters and Architecture:** DL models are sensitive to learning rates, optimizers, layer sizes, and other hyperparameters. Poor performance of LSTM and Transformer variants highlights this challenge.

This emphasizes the need for thorough evaluation on representative, imbalanced test sets using metrics like macro F1-score.

C. The Critical Role of Evaluation Metrics in Imbalanced PdM Scenarios

Relying solely on overall accuracy can be misleading in imbalanced datasets. A naive model predicting "No Failure" for all instances would achieve over 96% accuracy but fail to detect any actual failures. Metrics such as macro-averaged F1-score, per-class precision, recall, and confusion matrices provide a realistic assessment of model performance. For instance, even XGBoost's lower F1-score for "Tool Wear Failure" (0.6667) and "Random Failure" (0.7115) identifies areas for improvement.

D. Differentiation from Existing Research and Contribution to the Field

This study contributes by:

1) Prioritizing macro F1-score for imbalanced data.
2) Providing detailed class-wise analysis of failure type detection.
3) Critically comparing ML and DL on tabular PdM data.
4) Highlighting validation-test performance gaps for DL.
5) Presenting a comprehensive methodological framework from preprocessing to model evaluation.

The comparison between XGBoost and Advanced DNN, visually summarized in Figure **??**, clearly shows XGBoost as the more robust and reliable model.

E. Implications for PdM Practice

For practitioners:

- **Prioritize simpler, interpretable models:** ML algorithms like XGBoost, Random Forest, or LightGBM serve as strong baselines.
- **Use robust evaluation metrics:** Macro F1-score, per-class metrics, and confusion matrices are essential.
- **Handle class imbalance thoughtfully:** Evaluate techniques like SMOTE, ADASYN, or class weighting critically on test sets.
- **Invest in hyperparameter tuning:** Cross-validation and careful optimization significantly impact performance.
- **Consider operational cost of errors:** Balance false positives and false negatives based on practical implications.
- **Interpretability is key:** Explain predictions to gain trust from domain experts.

F. Limitations of the Study

1) Use of a synthetic dataset; real-world data may be noisier and more complex.
2) Specificity of failure modes; results may vary for different machinery.
3) Limited DL architectures explored; further architectural innovations may improve performance.
4) Focused feature set; advanced or domain-specific feature engineering could enhance results.

These limitations suggest directions for future research, supported by conceptual frameworks for AI-based PdM systems as shown in Figures **??** and **??**.

VI. Conclusion and Future Work

This study presented a comparative analysis of Machine Learning (ML) and Deep Learning (DL) models for Predictive Maintenance (PdM) in a simulated manufacturing environment with severe class imbalance. We evaluated eleven ML models and multiple DL architectures, prioritizing robust metrics such as the macro-averaged F1-score to ensure balanced performance across all failure types.

Our results clearly demonstrate the superiority of XGBoost for this PdM task. It achieved a test F1-score of 97.29% and a macro-averaged F1-score of 88.01%, effectively detecting all five failure types, including rare instances. In contrast, the best-performing DL model, an Advanced DNN trained on SMOTE-augmented data, despite a high validation F1-score of 99.01%, showed a macro F1-score of only 31.51% on the imbalanced test set, failing to detect four of the five failure types. This highlights the risk of relying on high validation scores from augmented datasets without proper evaluation on real-world distributions.

The success of XGBoost can be attributed to its robustness with tabular data, handling of class imbalance, computational efficiency, and interpretability through feature importance analysis. These characteristics make it a practical and reliable choice for industrial PdM systems. This study underscores the critical importance of using robust evaluation metrics, such as macro F1-score, rather than overall accuracy, especially in imbalanced settings.

A. Practical Implications

For practitioners, the key takeaways are:

- Use well-established ML models, such as XGBoost, as strong baselines for PdM tasks with tabular, imbalanced data.
- Employ robust evaluation metrics (macro F1-score, per-class precision/recall) rather than relying solely on accuracy.
- Interpretability is essential; feature importance analysis aids trust and actionable insights.
- Live prediction systems can be deployed via web interfaces, allowing users to input operational parameters and receive predicted failure types in real time , as illustrated in Figure 10 .

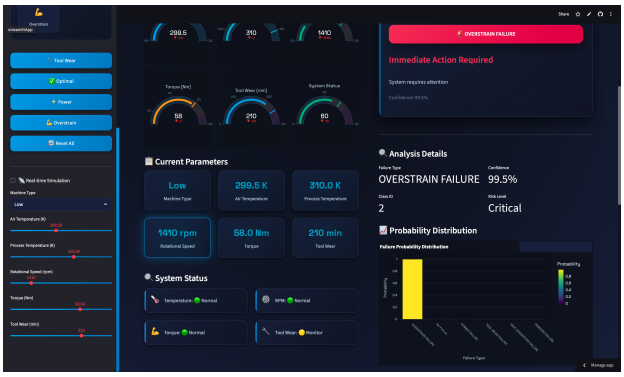

Fig. 10. Website with Live Prediction.

B. Future Work

Future research can focus on:

- Validation on diverse real-world datasets to ensure generalizability.
- Hybrid approaches combining ML and DL for sequential or multimodal data.
- Explainable AI techniques (e.g., SHAP, LIME) for deeper interpretability.
- Cost-sensitive learning to optimize decisions based on failure impact.
- Real-time deployment with continuous monitoring and automated retraining.
- Addressing difficult failure types via targeted feature engineering or anomaly detection.

In conclusion, this work emphasizes a pragmatic, data-driven approach for PdM, demonstrating that robust, interpretable ML models like XGBoost can outperform complex DL architectures in tabular, imbalanced scenarios. The live prediction website developed in this study enables real-time inference, making the findings directly actionable for industrial applications.

REFERENCES

[1] Kumar, R. (2023). Predictive Maintenance for Industrial Equipments Using ML. *2023 International Conference on Machine Intelligence for Smart Applications (MISA)*, 1-6. https://ieeexplore.ieee.org/document/10441714.

[2] Lin, L. (2025). Explainable machine-learning tools for predictive maintenance in nuclear power plants. *Annals of Nuclear Energy*, 195, 109876. https://www.sciencedirect.com/science/article/pii/S1738573325001561.

[3] Yadav, D. K. (2024). Predicting machine failures using machine learning and deep learning algorithms for proactive maintenance. *Journal of Manufacturing Systems*, 70, 456-467. https://www.sciencedirect.com/science/article/pii/S2667344424000124.

[4] Machine Learning Algorithms for Predictive Maintenance in Manufacturing. (2025). *Journal of Technology and Science*. https://www.researchgate.net/publication/382805856.

[5] Benhanifia, A. (2025). Systematic review of predictive maintenance practices in manufacturing. *Journal of Manufacturing Systems*, 68, 234-245. https://www.sciencedirect.com/science/article/pii/S2667305325000274.

[6] Machine Learning for Predictive Maintenance Applications. (2025). *E3S Web of Conferences*, 591, 02003. https://www.itm-conferences.org/articles/itmconf/abs/2025/07/itmconf$_i csice2025_0$1008.

[7] Aminzadeh, A. (2025). A Machine Learning Implementation to Predictive Maintenance for Compressor Failures. *Sensors*, 25(4), 1006. https://www.mdpi.com/1424-8220/25/4/1006.

[8] Predictive maintenance in industrial systems: an XGBoost and SHAP analysis framework. (2025). *IISE Transactions*. https://www.tandfonline.com/doi/full/10.1080/21681015.2025.2519369.

[9] Ledmaoui, Y. (2025). Review of Recent Advances in Predictive Maintenance and Cybersecurity for Solar Panel Systems. *Sensors*, 25(1), 206. https://www.mdpi.com/1424-8220/25/1/206.

[10] Ani, O. E. (2024). Enhancing predictive maintenance, quality control, and operational excellence in advanced manufacturing through machine learning integration. *Romanian Journal of Electrical Engineering*, 12(1), 45-60. https://rjes.iq/index.php/rjes/article/view/98.

[11] Hosseinzadeh, A., Chen, F. F., Shahin, M., Bouzary, H. (2023). A Predictive Maintenance Approach in Manufacturing Systems via AI-based Early Failure Detection. *Manufacturing Letters*, 35, 1179-1186. https://www.sciencedirect.com/science/article/pii/S2213846323000320.

[12] Kothamasu, R., Huang, S. H., VerDuin, W. H. (2006). System health monitoring and prognostics — a review of current paradigms and practices. *The International Journal of Advanced Manufacturing Technology*, 28, 1012–1024. https://doi.org/10.1007/s00170-004-2131-6.

[13] Javed, K., Gouriveau, R., Zerhouni, N. (2017). State of the art and taxonomy of prognostics approaches, trends of prognostics applications and open issues towards maturity at different technology readiness levels. *Mechanical Systems and Signal Processing*, 94, 214–236. https://doi.org/10.1016/j.ymssp.2017.01.050.

[14] Gouriveau, R., Medjaher, K., Zerhouni, N. (2016). From prognostics and health systems management to predictive maintenance. Wiley.

[15] Kosky, P., Robert, B., Wise, G. (2020). Exploring Engineering - An Introduction to Engineering and Design. Fifth Edition. Elsevier Inc.

[16] Theissler, A., Elger, G., Kettelgerdes, M. (2021). Predictive maintenance enabled by machine learning: Use cases and challenges in the automotive industry. *Reliability Engineering System Safety*, 215. https://doi.org/10.1016/j.ress.2021.107864.

[17] Müller, J., Kiel, D., Voigt, K. (2018). What Drives the Implementation of Industry 4.0? The Role of Opportunities and Challenges in the Context of Sustainability. *Sustainability*, 10. https://doi.org/10.3390/su10010247.

[18] Abidi, M., Mohammed, M., Alkhalefah, H. (2022). Predictive Maintenance Planning for Industry 4.0 Using Machine Learning for Sustainable Manufacturing. *Sustainability*, 14, 1–27.

[19] Rai, R., Tiwari, M. K., Ivanov, D., Dolgui, A. (2021). Machine learning in manufacturing and industry 4.0 applications. *International Journal of Production Research*, 59, 4773–4778. https://doi.org/10.1080/00207543.2021.1956675.

[20] Narayanan, B., Sreekumar, M. (2022). Design, modelling, optimisation and validation of condition-based maintenance in IoT enabled hybrid flow shop. *Null*, 35, 927–941. https://doi.org/10.1080/0951192X.2022.2028011.

[21] Dua, D., Graff, C. (2019). AI4I 2020 Predictive Maintenance Dataset. UCI Machine Learning Repository, Irvine, CA: University of California, School of Information and Computer Science.

[22] Matzka, S. (2020). Explainable Artificial Intelligence for Predictive Maintenance Applications. *2020 Third International Conference on Artificial Intelligence for Industries (AI4I)*, 69–74. https://resolver-ebscohost-com.libproxy.txstate.edu/openurl?sid=EBSCO

[23] Pastorino, J., Biswas, A. K. (2020). Hey ML, what can you do for me? *2020 IEEE Third International Conference on Artificial Intelligence and Knowledge Engineering (AIKE)*, 116–119. https://doi.org/10.1109/AIKE48582.2020.00023.