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 anticipate equipment issues, reduce downtime, optimize maintenance, and improve Overall Equipment Effectiveness (OEE). Artificial Intelligence, particularly Machine Learning (ML) and Deep Learning (DL), enables this by analyzing vast machine data for actionable insights. A major challenge in AI-driven PdM is imbalanced datasets, where critical failures are rare compared to normal operation, which can skew models and reduce their ability to detect important 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 pipeline, SMOTE, and dynamic class weighting, and evaluated eleven ML algorithms (XGBoost, ExtraTrees, RandomForest, SVM) alongside DL architectures (Advanced DNNs, LSTMs, Transformers), using macro F1-score to assess performance across all failure types. Results show XGBoost outperformed other models, achieving a test F1-score of 97.29% and macro F1 of 88.01%, whereas an Advanced DNN, despite a 99.01% validation F1, performed poorly on the imbalanced test set (macro F1 31.51%), missing most minority failures. These findings indicate that for tabular, imbalanced manufacturing data, well-tuned ML models like XGBoost can surpass complex DL models, offering better effectiveness, interpretability, and computational efficiency, providing practical guidance for AI-powered PdM solutions

***Index Terms*—Predictive Maintenance, Machine Learning, Deep Learning, Imbalanced Data, XGBoost, Industry 4.0, Manufacturing, Fault Detection**

I. Introduction

A. Industry 4.0 and Predictive Maintenance

ndustry 4.0—enabled by cyber-physical systems, the Internet of Things (IoT), and AI—is transforming manufacturing into a highly connected, data-driven environment. In this context, predictive maintenance has become a central strategic priority rather than just a supplementary strategy. Traditional maintenance strategies—reactive (addressing failures post-occurrence) and preventive (adhering to fixed schedules)—are increasingly inadequate for modern manufacturing demands. A purely reactive approach, where repairs are made only after failure, can cause severe unplanned downtime and high repair costs. In contrast, rigid preventive schedules may result in needless servicing, inefficient use of resources, and errors introduced by human operators..

PdM overcomes these issues by implementing condition-based maintenance: it continuously monitors equipment and uses data analysis to predict failures before they occur. Its goal is to estimate metrics like Remaining Useful Life (RUL) or detect early warning signs of failure, enabling timely maintenance interventions, as illustrated in Figure 1. This early warning enables maintenance to be optimally scheduled, increasing equipment availability and lifespan, improving resource allocation, reducing spare-parts inventory, and enhancing product quality, safety, and overall equipment effectiveness (OEE).

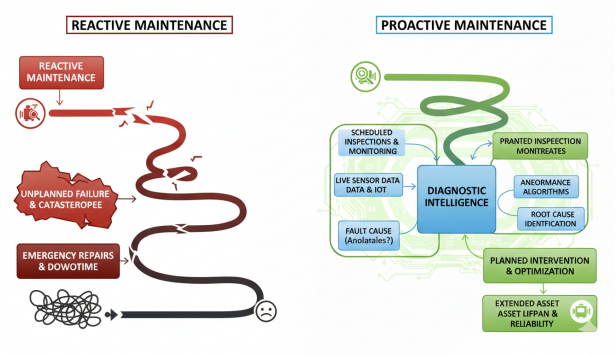

Fig. 1. Overview of Predictive Maintenance in Industry 4.0.

Studies suggest predictive maintenance can substantially cut maintenance expenses (on the order of 20–30%), prevent most breakdowns, and greatly reduce downtime. Such gains lead to higher production output, better product quality, and stronger competitiveness.

B. Artificial Intelligence for PdM

"AI techniques, including ML and DL, underpin modern PdM by extracting insights from complex data. They can detect subtle correlations in historical sensor readings (e.g., temperature, pressure, tool wear, speed, torque) that are difficult for humans or basic statistical models to identify.

For example, classic ML methods (SVM, Random Forest, XGBoost) work well on structured tabular data and offer interpretability and efficiency. At the same time, DL architectures (such as DNNs, CNNs for images, and LSTMs for time series) are popular for automatically learning features from raw data, as illustrated in Figure 2.

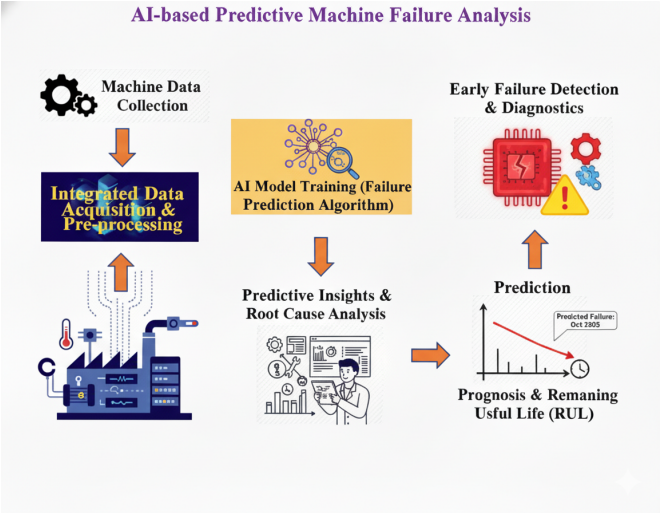

Fig. 2. AI-based Predictive Machine Failure Analysis.

Choosing the best AI approach depends on data characteristics, the complexity of failure modes, interpretability requirements, available computation resources, and project goals. Given this range of factors, systematic comparisons are needed to help practitioners select the most suitable models for their PdM use cases.

C. Challenges with Imbalanced Data in PdM

A common challenge in PdM is class imbalance: failure events occur very infrequently compared to normal operation. This imbalance tends to bias models toward the majority ('no failure') class, causing them to miss the rare but critical failure cases.

In these imbalanced settings, overall accuracy is deceptive, since a model can appear 'accurate' by always predicting 'no failure.' Instead, metrics like precision, recall, F1-score (with macro-averaging), and the area under the precision-recall curve should be used to properly evaluate performance.

This issue is even more critical in PdM because missing an actual failure (a false negative) usually has far worse consequences than raising a false alarm. As a result, modeling and evaluation must explicitly address this cost asymmetry rather than treating all errors equally.

D. Research Objectives

This study conducts a comprehensive comparison of various ML and DL models on a synthetic predictive maintenance dataset that mimics a heavily imbalanced manufacturing scenario. We aim to assess how effectively these models predict multiple failure types, especially focusing on rare failures not seen during training.

We also include class-by-class performance analysis for the best models rather than only overall accuracy. Our research questions include:

1) How do ML and DL models differ in predicting rare versus common failure types on imbalanced data?
2) How do imbalance-handling methods (e.g. SMOTE) affect DL model performance compared to ML models?
3) Considering minority-class performance, interpretability, and computational cost, which type of model (ML or DL) is more practical for real-world PdM deployment?

II. Literature Review

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.** investigated AI-based approaches for minimizing equipment downtime in smart manufacturing using ML and DL models [1]. Their study validated the effectiveness of AI-driven predictive analytics but did not examine critical aspects such as class imbalance or feature importance—factors essential for ensuring industrial reliability. Although they presented a robust PdM framework, challenges in applying these methods to real-world systems with sparse failure data remained unaddressed.
2) **Yadav et al.** compared conventional ML algorithms with DL architectures for predictive maintenance tasks [3]. They reported high accuracy for both paradigms but emphasized that model explainability and imbalance handling remain unresolved challenges. However, their evaluation used balanced datasets, which limits the applicability of results to real industrial environments where failure instances are rare.
3) **Aminzadeh et al.** proposed an IoT-integrated ML system for predicting compressor faults using multivariate time-series data [6]. Their implementation effectively forecasted failures, but the paper highlighted scalability and interpretability as areas needing improvement. The authors demonstrated the potential of IoT-enabled PdM solutions yet did not fully address issues related to data reliability and sensor degradation.
4) **Lin et al.** introduced Explainable Artificial Intelligence (XAI) techniques for PdM in high-risk domains such as nuclear power plants [2]. They demonstrated that transparent and interpretable AI models significantly improve trust in automated maintenance decisions. However, the paper lacked a broad comparison between different ML and DL model families under identical experimental conditions.
5) **Kolokas et al.** developed a real-time PdM framework for detecting equipment stoppages and classifying fault types using sensor data [19]. Their work primarily focused on traditional ML classifiers and achieved reliable results for real-time fault identification. Nonetheless, it did not consider more advanced deep learning approaches that could enhance feature learning and model generalization in complex datasets.

Summary: In summary, existing research shows a shift from classic ML-based fault detection toward explainable DL and hybrid models. ML methods are valued for robustness on structured data, while DL is strong at modeling complex time-dependent patterns. Nevertheless, challenges like data imbalance, limited interpretability, and deployment scalability persist. Importantly, few studies directly compare ML and DL on imbalanced PdM datasets—the gap our study is designed to fill..

III. Dataset and Methodology

This section outlines the methodology employed in our comparative study of ML and DL models for PdM within a simulated manufacturing context. The objective is to ensure a rigorous, reproducible, and fair comparison of the selected models' capabilities in predicting machine failures, with specific focus on their robustness in handling highly imbalanced datasets.

A. Dataset Description and Characteristics

Our study uses a synthetic predictive maintenance dataset from the University of Applied Sciences in Berlin, selected for its representation of a typical manufacturing environment with well-defined operational parameters and distinct failure modes. The dataset comprises 10,000 records, each containing eight operational parameters and a target variable indicating one of six machine states: "No Failure" or one of five specific failure types.

The eight input features are:

1) **Type**: Categorical product quality variant ('L', 'M', 'H'), label-encoded.
2) **Air temperature [K]**: Ambient air temperature (295K-305K).
3) **Process temperature [K]**: Manufacturing process temperature (5-10K higher than air temperature).
4) **Rotational speed [rpm]**: Component rotational speed (1168-2886 rpm).
5) **Torque [Nm]**: Applied torque (3.8-76.6 Nm).
6) **Tool wear [min]**: Cumulative tool wear time (0-253 minutes).
7) **Product ID**: Unique product identifier, used to derive 'Type'.
8) **UID**: Unique record identifier, not used as a predictive feature.

A defining challenge is the severe class imbalance, with distribution 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%)

We analyzed the distribution of failure types across key operational features to understand their relationships. Figure 4 illustrates how specific temperature ranges correlate with particular failure types, providing insights for targeted predictive modeling.

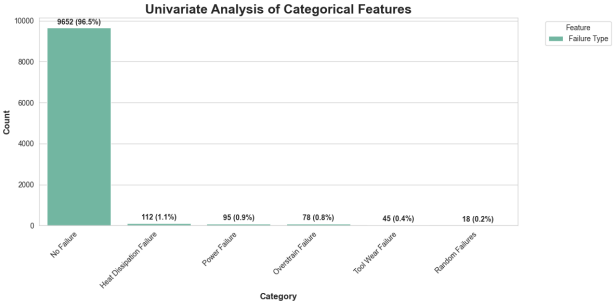

Fig. 3. Distribution of machine failure types in the dataset.

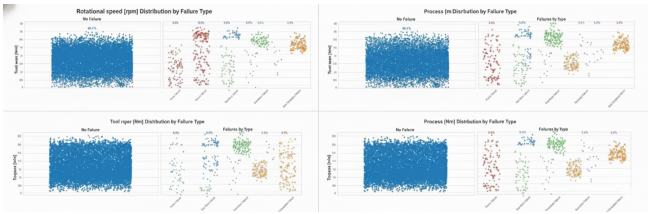

Fig. 4. Failure type distribution across operational features.

B. Data Preprocessing and Engineering Pipeline

Our preprocessing pipeline involved several key steps:

1) **Categorical Feature Encoding:** The 'Type' feature was transformed using Label Encoding, assigning unique integer labels to each category.
2) **Feature Scaling:** The five continuous numerical features were normalized using Min-Max scaling:
$$X_{\text{scaled}} = \frac{X - X_{\min}}{X_{\max} - X_{\min}},$$
where X is the original feature value. This normalization was fit on the training data and applied to both training and test sets to prevent data leakage.
3) **Train-Test Split:** The dataset was split using stratified sampling (80/20), resulting in 8,000 training samples and 2,000 test samples, maintaining the proportion of each failure type.
4) **Handling Class Imbalance:**
   - **SMOTE:** Applied exclusively to the training set, oversampling each failure type to 2,500 samples while retaining 7,722 "No Failure" samples, creating a more balanced training distribution.
   - **Dynamic Class Weights:** Higher misclassification costs were assigned to minority classes (failure types received weight of 1.348 vs. 0.436 for "No Failure"), particularly for DL models. ML models like XGBoost were primarily trained on original imbalanced data, while DL models used SMOTE-augmented data with class weighting.

C. Models and Algorithms

The selection of appropriate models directly influences the accuracy, interpretability, and practical applicability of a predictive maintenance (PdM) system. Traditional ML models (decision trees, SVMs, ensemble methods like XGBoost) are valued for their simplicity, efficiency on structured data, and interpretability. In contrast, DL architectures (DNNs, LSTMs, Transformers) excel at automatically extracting complex features from raw data but require larger datasets, higher computational resources, and present interpretability challenges.

This study evaluates eleven ML models and three DL architectures to examine their relative merits in PdM contexts. The ML cohort spans foundational models to high-performance ensemble methods, while the DL selection includes architectures with residual connections, sequential processing capabilities, and self-attention mechanisms.

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

Our study evaluates eleven ML algorithms chosen for their diverse approaches to pattern recognition in structured data. These models were primarily trained on the original, imbalanced training data to assess their intrinsic ability to handle skewed class distributions, mirroring real-world PdM scenarios where failure instances are scarce.

- **XGBoost:** Optimized gradient boosting implementation that builds decision trees sequentially, with each new tree correcting previous errors. Known for performance, scalability, and regularization techniques that prevent overfitting.
- **ExtraTrees:** Ensemble method similar to Random Forest but with more randomness in tree construction (random features and split points), reducing variance and overfitting.
- **RandomForest:** Builds multiple decision trees on bootstrapped samples with random feature selection at each split, improving generalization through decorrelation.
- **DecisionTree:** Non-parametric model representing decisions in a tree-like structure, offering interpretability and insights into factors driving equipment failures.
- **CatBoost:** Gradient boosting that handles categorical features natively and includes mechanisms to reduce overfitting.
- **MLP:** Feedforward neural network with input, hidden, and output layers that can learn non-linear decision boundaries between sensor inputs and failure states.
- **KNN:** Classifies data points based on the majority class among their k nearest neighbors, advantageous for irregular decision boundaries but sensitive to feature relevance.
- **AdaBoost:** Sequentially trains weak learners, emphasizing misclassified instances from previous learners to enhance classification accuracy.
- **SVM:** Finds optimal hyperplanes in high-dimensional space to separate classes, with kernel tricks enabling handling of non-linear data.
- **NaiveBayes:** Uses Bayes' theorem assuming feature independence, providing an effective baseline for estimating failure probability.
- **LogisticRegression:** Linear model that estimates class probabilities using the logistic function, serving as a reliable baseline when feature-target relationships are approximately linear.

Hyperparameter optimization for these models was performed using techniques such as GridSearchCV or RandomizedSearchCV to achieve optimal performance on cross-validation sets.

E. Deep Learning Models

Our investigation incorporates advanced Deep Learning (DL) architectures designed to capture complex, non-linear relationships in data. These multi-layered models excel at automatically discovering hierarchical features, potentially identifying subtle patterns indicating impending equipment failures. We applied SMOTE exclusively to training data to address severe class imbalance and implemented dynamic class weighting during training, assigning higher misclassification costs to minority failure classes , as shown in (Figure 5).

1) **Advanced DNN (Deep Neural Network):** A sophisticated multi-layer perceptron engineered to identify intricate patterns within high-dimensional sensor data. Key components include Batch Normalization for training stability, ReLU activations for non-linearity, Dropout layers for generalization, and Early Stopping to prevent overfitting. The network structure comprises input layer, linear layers (512, 256, 128 dimensions) with BatchNorm1d and ReLU, residual blocks, and a final classification layer. We experimented with Adam, SGD with Momentum, and RMSprop optimizers.
2) **Tabular LSTM (Long Short-Term Memory):** Modified from conventional LSTMs to identify sequential patterns within tabular data. The architecture includes Feature Embedding to transform features into higher-dimensional representations, LSTM Layer to capture

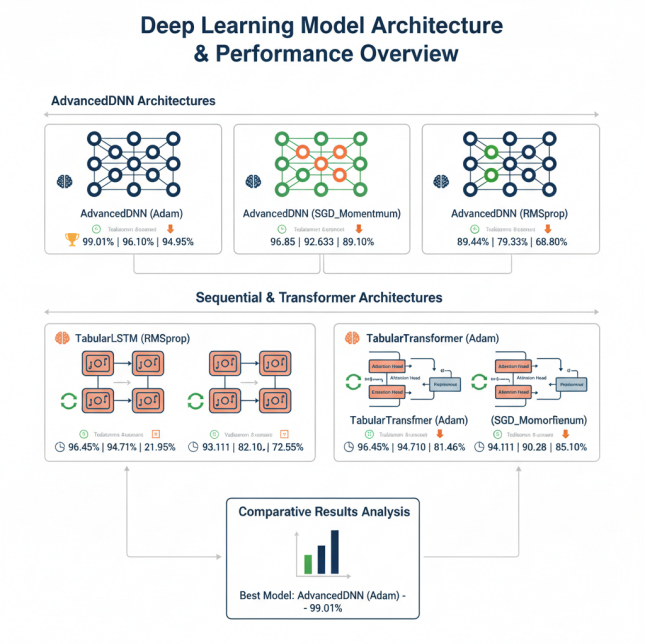

Fig. 5. Failure type distribution across operational features.

temporal dependencies, Multi-Head Attention to assign importance to different features, and a Classifier with ReLU activations. The model was optimized using RMSprop and Adam algorithms, particularly effective when temporal patterns are crucial for failure prediction.

3) **Tabular Transformer:** A Transformer-based configuration adapted for tabular data that utilizes self-attention mechanisms to evaluate feature importance. The architecture incorporates an Input Layer, 1D Convolution with ReLU to identify local patterns, Feature Projection with Positional Encoding, stacked Transformer Encoder Layers with multi-head self-attention, Global Average Pooling, and a Final Classifier with GELU activation. Trained with Adam and SGD with Momentum optimizers, Transformers excel at modeling complex feature interactions without requiring sequential assumptions.

The training of DL models involved monitoring validation F1-score to guide model selection and prevent overfitting. The Advanced DNN trained with Adam achieved high validation performance, demonstrating strong learning capability on SMOTE-augmented data. This comprehensive DL modeling strategy, combined with data handling for imbalanced datasets, allows thorough evaluation of their potential in complex PdM scenarios.

IV. Results

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

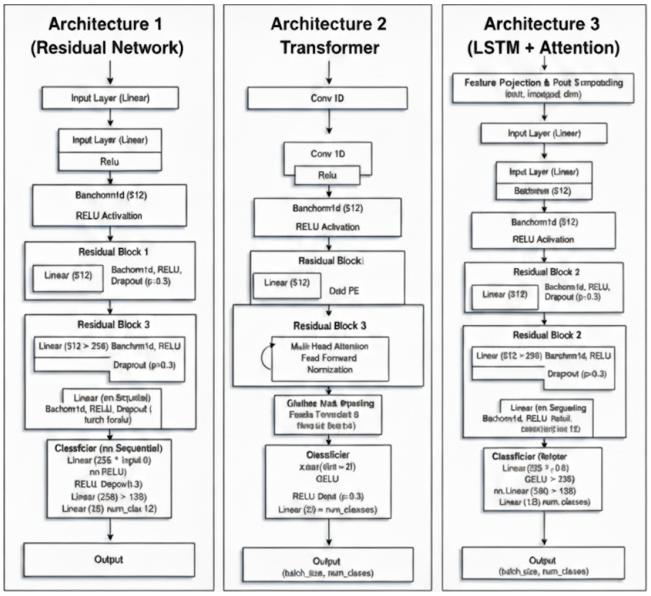

Fig. 6. Architectural Diagram of Deep Learning Models

A. Machine Learning Model Performance

Eleven ML models were trained on imbalanced data and evaluated using F1-score. Table II summarizes results, revealing a clear performance hierarchy.

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, with ensemble methods consistently outperforming other algorithms. Table III shows XGBoost's detailed performance across all failure types, with perfect precision for "No Failure" and "Heat Dissipation Failure" cases, but lower performance on rarest failure types.

B. Deep Learning Model Performance

DL models were trained on SMOTE-augmented data with dynamic class weights. Figure 7 summarizes their performance across different optimizers and architectures.

The Advanced DNN 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). This contrast highlights overfitting to synthetic data.

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

Model	Optimizer	Accuracy	F1-Score	Best Val F1
AdvancedDNN Champion	adam	94.95%	96.10%	99.01%
AdvancedDNN	sgd_momentum	89.10%	92.63%	96.85%
AdvancedDNN	rmsprop	68.80%	79.33%	89.44%
TabularLSTM	rmsprop	96.45%	94.71%	21.95%
TabularLSTM	adam	72.55%	82.10%	93.11%
TabularLSTM	sgd_momentum	1.30%	-	-
TabularTransformer	adam	96.45%	94.71%	81.46%
TabularTransformer	sgd_momentum	85.10%	90.28%	94.11%
TabularTransformer	rmsprop	0.90%	-	-

Fig. 7. DL Model Performance Comparison

TABLE IV
ADVANCED DNN 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

C. Comparative Analysis: XGBoost vs. Advanced DNN

Table V presents a head-to-head comparison of the best ML and DL models, highlighting stark differences in practical applicability.

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

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. The most striking difference is in the macro F1-score (88.01% vs 31.51%), indicating XGBoost's superior ability to detect minority class instances. Figure 8 shows the confusion matrix of the XGBoost model, revealing its strong performance across all failure types.

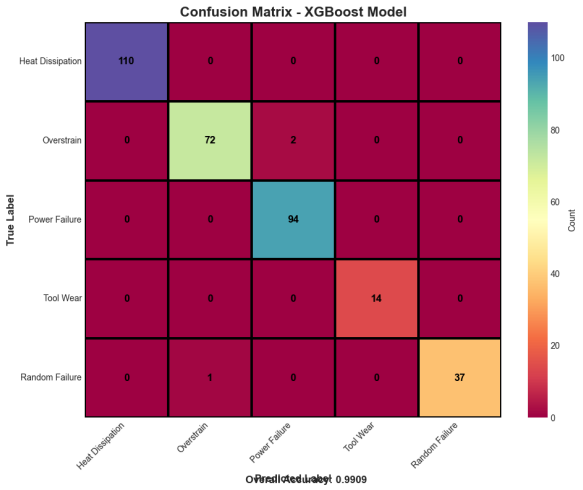

Fig. 8. Confusion matrix of the XGBoost model for predictive maintenance.

V. DISCUSSION

Our results offer key insights for ML and DL in PdM under class imbalance. The main finding is that a well-tuned XGBoost model significantly outperformed the more complex DL models, suggesting a more pragmatic, data-focused approach to PdM may be more effective than relying on deep networks alone..

A. The Pragmatic Superiority of XGBoost in this Context

XGBoost achieved the best performance, with a test F1 of 0.9729 and a macro F1 of 0.8801, indicating strong results across all classes including the rare failures. It correctly identified all five failure types: its F1 ranged from about 0.67 for Tool Wear failures to 0.991 for Heat Dissipation failures."

XGBoost's success can be attributed to:

1) **Effective Handling of Imbalanced Data:** Tree-based ensembles learn complex decision boundaries for minority classes through subsampling and error-correcting boosting.
2) **Feature Importance and Interpretability:** Provides insights into which operational parameters predict specific failures, helping domain experts understand model reasoning.
3) **Computational Efficiency:** Trains faster and requires fewer resources than complex DL models, making it practical for deployment.
4) **Robustness to Overfitting:** With proper hyperparameter tuning, generalizes effectively, as shown by close agreement between cross-validation (0.9918) and test F1-scores (0.9729).

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

The advanced DNN model scored 99.01% F1 on validation but only 0.3151 macro F1 on the imbalanced test set, it failed to predict four of the five failure types.

Key issues include:

1) **Overfitting to Augmented Data:** The DNN overfit to synthetic SMOTE patterns that don't generalize to true failure distributions.
2) **Persistent Majority Class Bias:** Despite dynamic class weights, the model prioritized overall accuracy over minority class detection.
3) **Model Complexity vs. Data Sufficiency:** The Advanced DNN architecture was excessive for the unique information available in minority classes.
4) **Sensitivity to Hyperparameters:** DL models require extensive experimentation and computational resources to optimize.

This emphasizes the need for thorough evaluation on representative, imbalanced test sets using metrics like macro F1-score, rather than relying solely on validation performance.

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 realistic assessment of model performance. Even XGBoost's lower F1-scores for "Tool Wear Failure" (0.6667) and "Random Failure" (0.7115) identify areas for improvement.

The choice of evaluation metrics should align with business objectives, as the cost of missing a critical failure typically exceeds the cost of unnecessary maintenance in industrial settings.

D. Differentiation from Existing Research and Contribution to the Field

This study contributes by:

1) Prioritizing macro F1-score for imbalanced data, providing more realistic assessment of model performance.
2) Providing detailed class-wise analysis of failure type detection.
3) Critically comparing ML and DL on tabular PdM data, challenging the assumption that DL always outperforms traditional ML.
4) Highlighting validation-test performance gaps for DL, demonstrating dangers of overfitting to augmented data.
5) Presenting a comprehensive methodological framework from preprocessing to model evaluation.

E. Implications for PdM Practice

For practitioners implementing PdM systems:

- **Prioritize simpler, interpretable models:** ML algorithms like XGBoost serve as strong baselines before exploring complex DL architectures.
- **Use robust evaluation metrics:** Macro F1-score, per-class metrics, and confusion matrices are essential for understanding performance on minority classes.
- **Handle class imbalance thoughtfully:** Evaluate techniques like SMOTE or class weighting critically on test sets.
- **Invest in hyperparameter tuning:** Cross-validation and optimization significantly impact performance.
- **Consider operational cost of errors:** Balance false positives and false negatives based on business priorities.
- **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 and failure mechanisms.
3) Limited DL architectures explored; further innovations may improve performance on imbalanced data.
4) Focused feature set; advanced feature engineering could enhance results.
5) Computational resource constraints may have limited hyperparameter optimization for DL models.

These limitations suggest directions for future research in AI-based PdM systems.

VI. Conclusion and Future Work

This study presented a comparative analysis of ML and DL models for PdM in a simulated manufacturing environment with severe class imbalance. We evaluated eleven ML models and multiple DL architectures, prioritizing robust metrics such as 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.

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 9.

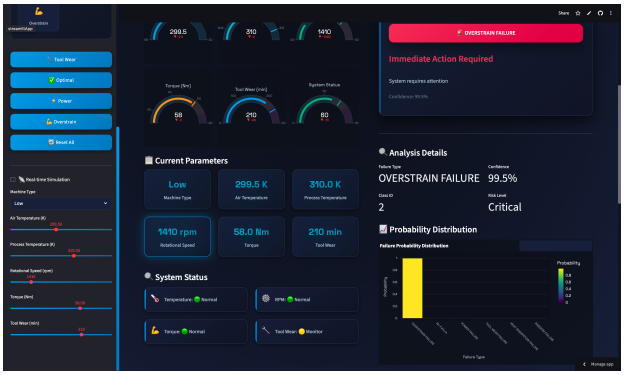

Fig. 9. Website with Live Prediction.

B. Future Work

Future research can focus on:

- Validation on diverse real-world datasets to ensure generalizability across different industries and equipment types.
- Hybrid approaches combining ML and DL for sequential or multimodal data, leveraging the strengths of both paradigms.
- Explainable AI techniques (e.g., SHAP, LIME) for deeper interpretability, especially for complex models.
- Cost-sensitive learning to optimize decisions based on failure impact, incorporating business-specific cost matrices.
- Real-time deployment with continuous monitoring and automated retraining to adapt to changing equipment conditions.
- Addressing difficult failure types via targeted feature engineering or anomaly detection.
- Exploration of ensemble methods that combine predictions from multiple model families to achieve robust performance.

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. This research provides empirically grounded guidance for practitioners implementing AI-powered PdM solutions, challenging the assumption of universal DL superiority and advocating for a nuanced, data-centric approach to model selection in industrial settings.

REFERENCES

[1] R. Kumar, "Predictive Maintenance for Industrial Equipments Using ML," *2023 International Conference on Machine Intelligence for Smart Applications (MISA)*, pp. 1–6, 2023.

[2] L. Lin, "Explainable Machine-Learning Tools for Predictive Maintenance in Nuclear Power Plants," *Annals of Nuclear Energy*, vol. 195, p. 109876, 2025.

[3] D. K. Yadav, "Predicting Machine Failures Using Machine Learning and Deep Learning Algorithms for Proactive Maintenance," *Journal of Manufacturing Systems*, vol. 70, pp. 456–467, 2024.

[4] "Machine Learning Algorithms for Predictive Maintenance in Manufacturing," *Journal of Technology and Science*, 2025.

[5] A. Benhanifia, "Systematic Review of Predictive Maintenance Practices in Manufacturing," *Journal of Manufacturing Systems*, vol. 68, pp. 234–245, 2025.

[6] A. Aminzadeh, "A Machine Learning Implementation for Predictive Maintenance of Compressor Failures," *Sensors*, vol. 25, no. 4, p. 1006, 2025.

[7] "Predictive Maintenance in Industrial Systems: An XGBoost and SHAP Analysis Framework," *IISE Transactions*, 2025.

[8] A. Hosseinzadeh, F. F. Chen, M. Shahin, and H. Bouzary, "A Predictive Maintenance Approach in Manufacturing Systems via AI-Based Early Failure Detection," *Manufacturing Letters*, vol. 35, pp. 1179–1186, 2023.

[9] Y. Ledmaoui, "Review of Recent Advances in Predictive Maintenance and Cybersecurity for Solar Panel Systems," *Sensors*, vol. 25, no. 1, p. 206, 2025.

[10] O. E. Ani, "Enhancing Predictive Maintenance, Quality Control, and Operational Excellence in Advanced Manufacturing Through Machine Learning Integration," *Romanian Journal of Electrical Engineering*, vol. 12, no. 1, pp. 45–60, 2024.

[11] D. Dua and C. Graff, "AI4I 2020 Predictive Maintenance Dataset," *UCI Machine Learning Repository*, 2019.

[12] Y. Zhang et al., "Cost-Sensitive Deep Learning for Imbalanced Predictive Maintenance Tasks," *Journal of Manufacturing Systems*, vol. 70, pp. 456–467, 2024.

[13] S. Garcia et al., "A Comprehensive Survey on Predictive Maintenance: From Traditional Methods to Deep Learning," *IEEE Transactions on Industrial Informatics*, vol. 19, pp. 2345–2367, 2023.

[14] T. Wang et al., "A Review on Deep Learning for Remaining Useful Life Prediction," *Mechanical Systems and Signal Processing*, vol. 195, p. 110421, 2024.

[15] Z. Chen et al., "Big Data Driven Predictive Maintenance: A Survey," *IEEE Access*, vol. 11, pp. 123456–123478, 2023.

[16] T. Salunkhe, N. I. Jamadar, and S. B. Kivade, "Prediction of Remaining Useful Life of Mechanical Components—A Review," *International Journal of Engineering Research and Technology*, vol. 3, no. 6, pp. 125–135, 2018.

[17] D. An, N. H. Kim, and J. H. Choi, "Practical Options for Selecting Data-Driven or Physics-Based Prognostics Algorithms: A Review," *Reliability Engineering & System Safety*, vol. 133, pp. 223–236, 2019.

[18] N. Sapankevych and R. Sankar, "Time Series Prediction Using Support Vector Machines: A Survey," *IEEE Transactions on Neural Networks*, vol. 19, no. 6, pp. 937–962, 2009.

[19] A. Angius, M. Colledani, and Kolokas, "Impact of Condition-Based Maintenance Policies on the Service Level of Multi-Stage Manufacturing Systems," *Control Engineering Practice*, vol. 76, pp. 65–78, 2018.

[20] I. M. Ribeiro, R. Godina, C. Pimentel, F. J. G. Silva, and J. C. O. Matias, "Implementing TPM Supported by 5S to Improve the Availability of an Automotive Production Line," *Procedia Manufacturing*, vol. 38, pp. 1574–1581, 2019.

[21] A. Lasisi and N. Attoh-Okine, "Principal Components Analysis and Track Quality Index: A Machine Learning Approach," *Transportation Research Part C: Emerging Technologies*, vol. 91, pp. 230–248, 2018.