



Research Article

Sampling Strategies and Metric Sensitivity in Imbalanced Predictive Maintenance: A Comparative Study with Ensemble Methods

Berk Mercan ^{a,*} , Eren Ağar ^a , Mert Çimentepe ^a , Meltem Apaydın Üstün ^a 

^a Tekirdağ Namık Kemal University, Çorlu Faculty of Engineering, Electrical and Electronics Engineering Department, Çorlu, Tekirdağ, Türkiye

ARTICLE INFO

Article history:

Received 29 June 2026

Accepted 14 September 2026

Keywords:

Class imbalance,
Feature engineering,
Predictive maintenance,
Random Forest,
SHAP,
XGBoost

ABSTRACT

This study presents a systematic comparative analysis of class balancing strategies for predictive maintenance of industrial equipment using the AI4I 2020 Predictive Maintenance Dataset, which comprises 10,000 samples with a severe class imbalance of 3.4% failure instances. Two tree-based ensemble classifiers, Random Forest (RF) and XGBoost, are evaluated under nine balancing configurations: Baseline, class-weight balancing, SMOTE, ADASYN, Random Undersampling (RUS), SMOTE-Tomek, SMOTEENN, SMOTE+RUS, and SVMSMOTE. Four domain-informed engineered features, Delta_T, Power Proxy, Temp_Ratio, and Torque_per_Wear, are derived from physical process knowledge and integrated into the modeling pipeline. Models are evaluated using Precision, Recall, F1-score, and PR-AUC, with particular emphasis on PR-AUC due to the severe class imbalance. The results show that RF is robust to class imbalance without complex resampling. Tuned Baseline RF achieves the best RF performance with an F1-score of 86.4% and a PR-AUC of 0.886, while Tuned Balanced RF provides a comparable F1-score of 85.7% and a PR-AUC of 0.875. In contrast, XGBoost benefits more from sampling strategy and hyperparameter optimization: Tuned SVMSMOTE + XGB achieves the strongest tuned XGBoost performance with an F1-score of 83.0% and the highest PR-AUC of 0.914. These findings demonstrate that sampling methods do not universally improve classifier performance; rather, their effectiveness depends on the learning algorithm. In addition, SHAP-based interpretability analysis shows that Rotational Speed, Torque, Tool Wear, and engineered load-related features contribute substantially to failure prediction. Overall, PR-AUC is recommended as a primary evaluation metric for imbalanced industrial predictive maintenance datasets.

This is an open access article under the CC BY-SA 4.0 license.
(<https://creativecommons.org/licenses/by-sa/4.0/>)

1. INTRODUCTION

Unplanned equipment failures in modern manufacturing environments impose significant economic burdens through production downtime, emergency repair costs, and safety risks. Traditional maintenance strategies, namely reactive (fix-on-failure) and preventive (schedule-based), have proven inadequate: the former results in costly unplanned stoppages, while the latter leads to unnecessary maintenance actions on healthy equipment. Predictive Maintenance (PdM), enabled by the convergence of Industry 4.0 technologies—Internet of Things (IoT) sensor networks, big data platforms, and machine learning—represents a paradigm shift toward condition-based, data-

driven maintenance decision-making that addresses both failure detection latency and unnecessary preventive interventions [1].

Machine learning-based PdM systems learn failure patterns from multivariate sensor streams (temperature, vibration, torque, rotational speed, tool wear) and generate probabilistic failure forecasts, enabling maintenance teams to intervene precisely when needed. However, a fundamental methodological challenge undermines the reliability of these systems: class imbalance. In real industrial environments, failure events are inherently rare, and predictive maintenance datasets are therefore typically characterized by severe class imbalance [1,2]. Standard

* Corresponding author. E-mail address: 1230657013@nku.edu.tr
DOI: 10.58190/ijamec.2026.179

classification algorithms trained on imbalanced data tend to be biased toward the majority (healthy) class, achieving deceptively high accuracy while failing to detect faults—the very events the system is intended to identify. Addressing this imbalance is therefore not merely a preprocessing step but a core design consideration that directly affects the operational utility of a PdM system.

The literature offers a wide range of techniques to mitigate class imbalance, broadly categorized as data-level methods (oversampling, undersampling, and hybrid methods), algorithm-level methods (cost-sensitive learning, class weighting), and ensemble-level methods. Despite the availability of these approaches, a critical gap persists: the interaction between the choice of balancing strategy and the choice of learning algorithm remains insufficiently characterized in the PdM domain. Most existing comparative studies either evaluate a limited set of strategies, focus on a single classifier, or lack interpretability analysis to explain model decisions to domain practitioners.

This paper addresses these gaps through a structured experimental framework designed to evaluate the interaction between class balancing strategies and learning algorithms in imbalanced predictive maintenance. Random Forest and XGBoost, two widely used tree-based ensemble classifiers in industrial machine learning applications, are benchmarked against nine class balancing configurations on the publicly available AI4I 2020 Predictive Maintenance Dataset [3]. These configurations include the baseline model without balancing, class-weight adjustment, synthetic oversampling methods, undersampling, and hybrid sampling strategies. In addition to model development, this study incorporates domain-informed feature engineering to improve both predictive performance and practical relevance by embedding basic physical relationships among temperature, torque, rotational speed, and tool wear variables. Furthermore, SHAP-based interpretability analysis is included to examine the contribution of individual features to model predictions and to provide more transparent explanations for domain practitioners [4].

The main contributions of this study are as follows: (i) a full factorial comparison of algorithm–sampling strategy interactions across 18 model configurations using Random Forest and XGBoost; (ii) the incorporation of physical domain knowledge through engineered features; (iii) a metric-oriented evaluation framework emphasizing F1-score, Recall, and PR-AUC for imbalanced industrial classification tasks; (iv) a comparative analysis of the relative effects of sampling strategies and hyperparameter optimization for RF and XGBoost; and (v) a SHAP-based interpretability analysis to support the explanation of model decisions.

The scope of this study is deliberately limited to tree-based ensemble methods. This choice is motivated by the

structure of the AI4I 2020 dataset, which consists of tabular sensor-derived features rather than raw sequential time-series or unstructured signal data. Although deep learning architectures have shown strong performance in time-series and signal-based predictive maintenance applications, they generally require larger datasets, higher computational resources, and often provide limited interpretability. In contrast, tree-based ensemble methods offer a practical balance between predictive performance, computational efficiency, and interpretability for structured industrial datasets. Therefore, deep learning architectures are not included in the experimental comparison and are left as a direction for future work.

1.1. Related Work

Research on PdM can be broadly categorized into three groups based on the signal modalities and modeling paradigms employed: classical threshold-based and signal processing approaches, classical machine learning approaches, and deep learning approaches.

Classical and Signal Processing-Based Approaches. Early PdM systems relied on physics-based thresholds and hand-crafted signal features derived from current, vibration, and temperature sensors. These methods exploit known failure physics: for instance, stator current monitoring in induction motors can identify rotor bar breakage through sideband frequency analysis in the current spectrum [5]. While interpretable and computationally inexpensive, these approaches require detailed prior knowledge of failure mechanisms and do not generalize well across equipment types or operating conditions. Their sensitivity is also limited in detecting incipient faults that manifest as subtle statistical deviations below deterministic alarm thresholds [6].

Classical Machine Learning Approaches. The availability of labeled sensor datasets has enabled the application of supervised classifiers to PdM. Several studies have compared traditional classifiers—Support Vector Machines (SVM), k-Nearest Neighbors (kNN), Decision Trees, and ensemble methods—on fault detection tasks using static feature vectors extracted from sensor signals. Kalafatelis et al. evaluated multiple classifiers combined with the Synthetic Minority Oversampling Technique (SMOTE), Adaptive Synthetic Sampling (ADASYN), and Random Undersampling (RUS) on an offset printing machine dataset, demonstrating that the interaction between sampling method and classifier architecture is a primary determinant of minority class recall rather than the sampling method alone [7]. Similarly, Hakami demonstrated that GAN-generated synthetic data, when used to augment training sets prior to RF and XGBoost training, can alleviate the scarcity of failure samples, achieving ANN accuracy of 88.98% and RF accuracy of 74.15% on a benchmark dataset; the author also noted that temporal dependencies

in sensor data are not captured by static feature extraction [8]. Patel and Shanbhag evaluated multiple ML models on the AI4I 2020 dataset and observed that Random Forest achieved an F1-score of 85% under the standard SMOTE oversampling method but did not investigate whether alternative balancing strategies could further improve or degrade this performance [9]. A broader comparative study by Zaidi et al. found that multi-sensor data fusion—combining vibration, temperature, acoustic emission, current, and speed signals—raised fault classification performance from 84% to over 99% when using kNN, SVM, RF, and Decision Trees, underscoring the importance of feature richness alongside algorithmic choice [10].

Deep Learning Approaches. Temporal sensor data has motivated the application of recurrent and convolutional architectures for PdM. Borré et al. applied a hybrid CNN-LSTM model to vibration signals from electrical motors, using three-axis accelerometer data to detect anomalies via empirical cumulative distribution-based outlier detection; the temporal attention mechanism enabled the model to focus on degradation patterns occurring in specific time windows prior to failure [11]. A comparative evaluation of deep learning architectures by Li and Li demonstrated that hybrid CNN-LSTM models achieve 96.1% accuracy and 95.2% F1-score on industrial sensor datasets, outperforming standalone architectures, while also noting that these gains come at substantially higher computational cost and reduced interpretability [12]. For short-horizon failure forecasting in industrial pumps, Alghtus et al. combined sliding-window time series features with RF and XGBoost classifiers, achieving 65.6% recall at 15- and 30-minute prediction horizons, and found that optimal history window length depends on the target prediction horizon and the specific degradation dynamics of the monitored system [13]. The systematic review by Carvalho et al. further highlighted that while LSTMs and CNN variants achieve state-of-the-art results in PdM, the black-box nature of deep models creates barriers to adoption in safety-critical manufacturing contexts where maintenance decisions must be explainable to operators [2].

Although deep learning architectures such as CNN and LSTM have shown strong performance in sequential fault detection tasks, they are often regarded as black-box models with limited interpretability and higher computational cost. In industrial PdM settings where model transparency is critical for maintenance decision-making, this study focuses on ensemble tree-based methods that offer both competitive performance and interpretability for structured tabular datasets [14,15,16]. SHAP analysis is further employed to support the explanation of feature contributions in the selected models [4].

This paper seeks to address a specific research gap identified in the literature. While ensemble classifiers have

been shown to perform strongly on structured industrial sensor data, and while various class balancing strategies are available, the interaction between learning algorithm, sampling strategy, hyperparameter optimization, metric sensitivity, and interpretability remains insufficiently examined for the AI4I 2020 dataset. Specifically, this study evaluates Random Forest and XGBoost across nine distinct class balancing configurations, integrates domain-informed feature engineering, applies hyperparameter optimization to selected high-performing configurations, and incorporates SHAP-based interpretability analysis. In this way, the study provides practical guidance for designing transparent and metric-aware ML-based PdM pipelines under severe class imbalance.

2. MATERIALS AND METHODS

2.1. Dataset

In this study, we used the AI4I 2020 Predictive Maintenance Dataset [3], accessible through the UCI Machine Learning Repository. The dataset simulates sensor readings from a CNC machine tool in a manufacturing environment, comprising 10,000 observations across six input variables and a binary failure target. It is designed to reflect realistic industrial conditions, including the severe class imbalance characteristic of real PdM scenarios: 9,661 samples (96.6%) correspond to healthy operation, while 339 (3.4%) represent failure events. No missing values are present in the dataset. The resulting class distribution is illustrated in Figure 1. Table 1 summarizes the input variables and the target variable.

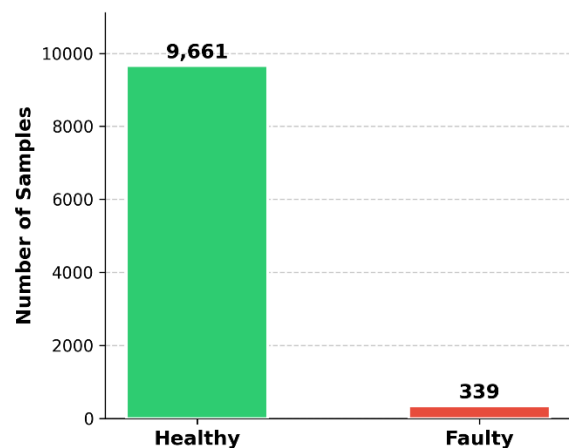


Figure 1. Class distribution of the AI4I 2020 dataset

Table 1. Feature descriptions of the AI4I 2020 dataset

Feature	Type	Description	Range / Values
Air Temperature [K]	Continuous	Ambient air temperature	~295–305 K
Process Temperature [K]	Continuous	Machining process temperature	~305–315 K
Rotational Speed [rpm]	Continuous	Spindle rotational speed	~1000–2800 rpm
Torque [Nm]	Continuous	Applied torque on spindle	~0–77 Nm
Tool Wear [min]	Continuous	Cumulative tool usage duration	0–253 min
Type	Categorical	Product quality class	L / M / H
Target	Binary	Failure label (output)	0: Non-Faulty, 1: Faulty

2.1.1. Feature Engineering

We constructed four derivative features from the raw sensor variables, guided by physical process knowledge of Computer Numerical Control (CNC) machining dynamics [17]. These features are designed to capture non-linear interaction effects that are not directly observable from individual sensor readings. The engineered features and their physical rationale are given in Table 2.

Table 2. Engineered features and their physical rationale

Feature	Formula	Physical Rationale
Delta_T	$T_{\text{process}} - T_{\text{air}}$	Thermal gradient between process and ambient; elevated values may signal abnormal heat generation
Power_Proxy	$\text{Torque} \times \text{RPM}$	Proxy for instantaneous mechanical power; captures combined load effect
Temp_Ratio	$T_{\text{process}} / T_{\text{air}}$	Normalized thermal loading; compensates for ambient temperature drift
Torque_per_Wear	$\text{Torque} / (\text{Wear} + 1)$	Mechanical load normalized by tool age; captures load-wear interaction

2.2. Preprocessing Methods

The categorical product type variable (Type: L, M, H) was transformed into binary indicator variables via One-Hot Encoding. One category was dropped to avoid multicollinearity, yielding two binary columns (Type_L and Type_M). The resulting feature matrix comprises 11 variables after encoding and feature engineering.

The dataset was partitioned into training (80%) and test (20%) sets using stratified random sampling to preserve the 96.6/3.4 class ratio in both subsets, resulting in 8,000 training and 2,000 test samples. All continuous features were scaled using RobustScaler, which scales data by the interquartile range rather than the standard deviation. This choice is motivated by the presence of 418 outliers in rotational speed and 69 in torque, identified via the IQR

method. Because these outliers reflect genuine extreme operating conditions rather than measurement error, they were retained, and the robust scaling strategy mitigates their influence on model training. To prevent data leakage, the scaler was fitted exclusively on the training set and subsequently applied to the test set.

2.3. Machine Learning Methods

Random Forest (RF) [14]. RF is a bagging-based ensemble method that constructs a collection of decision trees on bootstrap samples of the training data, with random feature subsets evaluated at each split. Predictions are aggregated by majority vote. The bootstrap mechanism provides implicit class overrepresentation variation across trees, which confers a degree of natural robustness to class imbalance. RF is computationally efficient, non-parametric, and produces feature importance estimates through mean decrease in Gini impurity.

XGBoost [15]. XGBoost implements gradient boosted trees using a regularized objective function. Unlike bagging, boosting builds trees sequentially, with each tree trained to correct the residual errors of the ensemble. The sequential focus on misclassified samples makes XGBoost sensitive to the initial class distribution; in imbalanced settings, the `scale_pos_weight` parameter provides a direct mechanism for up-weighting minority class errors. XGBoost is consistently competitive on tabular datasets and offers fast inference.

Nine class balancing configurations were applied to each classifier: Baseline (no balancing), class-weight adjustment (balanced), SMOTE [18], ADASYN [19], Random Undersampling (RUS), SMOTE-Tomek, SMOTEENN, SMOTE+RUS (hybrid), and SVMSMOTE [20]. All sampling transformations are encapsulated in imblearn Pipeline objects [21], ensuring that synthetic sample generation occurs exclusively within each training fold of the outer 5-fold stratified cross-validation loop, and that validation and test data are never used in generating synthetic samples. For the selected high-performing RF and XGBoost configurations, hyperparameter optimization was conducted via RandomizedSearchCV (`n_iter`=15, F1-score as the optimization criterion) within the training partition using stratified cross-validation. The RF search space included `n_estimators`, `max_depth`, `min_samples_split`, `min_samples_leaf`, `max_features`, and `class_weight`, whereas the XGBoost search space included `n_estimators`, `max_depth`, `learning_rate`, `subsample`, `colsample_bytree`, `min_child_weight`, and `gamma`. Final test set evaluation was performed only once on the held-out 20% partition that was not used during model selection or hyperparameter search. All experiments used a fixed random seed (`random_state`=42) for reproducibility, with scikit-learn 1.3 and imbalanced-learn 0.11.

2.4. Performance Metrics

The reported performance indicators include Precision, Recall, F1-score, and PR-AUC. Precision measures the proportion of predicted failure cases that are actually failures, while Recall measures the proportion of actual failures that are correctly detected. They are defined in Eq. (1) and Eq. (2), respectively:

$$\text{Precision} = \frac{TP}{TP+FP} \quad (1)$$

$$\text{Recall} = \frac{TP}{TP+FN} \quad (2)$$

where TP , FP , and FN denote true positives, false positives, and false negatives, respectively. The F1-score is the harmonic mean of Precision and Recall and is calculated as given in Eq. (3):

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (3)$$

F1-score provides a balanced measure of false positives and false negatives, which is especially useful for imbalanced classification problems. ROC-AUC, which quantifies the classifier's ability to discriminate between classes across different decision thresholds, is not reported here because it can be overly optimistic under severe class imbalance.

2.5. SHAP-Based Interpretability Analysis

To improve the interpretability of the selected final model, SHAP (SHapley Additive exPlanations) analysis was applied [4]. SHAP assigns an importance value to each feature by quantifying its contribution to the model prediction. In this study, SHAP analysis was performed on the final tuned XGBoost model with the strongest overall minority-class performance, namely Tuned SVMSMOTE + XGB. Mean absolute SHAP values were used to rank the features according to their overall contribution to the model's predictions on the held-out test set.

3. RESULTS AND DISCUSSION

3.1. Random Forest Results

Table 3 presents the performance of all nine Random Forest configurations obtained through stratified 5-fold cross-validation on the training partition and final evaluation on the held-out test set. In line with the imbalanced nature of the dataset, the comparison focuses on recall, F1-score, and PR-AUC rather than accuracy or ROC-AUC, since accuracy can be dominated by the majority class and ROC-AUC may provide overly optimistic interpretations under severe class imbalance. The cross-validation columns summarize the mean fold-wise performance, whereas the test columns reflect the final generalization performance on unseen data.

A methodological caution is necessary when interpreting the test results: the held-out test partition

contains only 68 failure instances, corresponding to 3.4% of the 2,000 test samples. Therefore, a small number of misclassified minority samples can noticeably affect the F1-score, and minor performance differences should not be overinterpreted. With this consideration, Baseline RF achieves the best overall test performance, reaching the highest F1-score of 86.4% and the highest PR-AUC of 0.886. Balanced RF follows as the second-best configuration, with a test F1-score of 82.6% and a PR-AUC of 0.871. These results indicate that Random Forest performs strongly on the original data distribution and that additional sampling strategies do not necessarily improve minority-class performance for this dataset.

Table 3. Cross-validation and test set performance of Random Forest configurations

Model	CV F1 (%)	CV Recall (%)	Test Recall (%)	Test F1 (%)	Test PR-AUC
Baseline RF	83.8	73.5	79.4	86.4	0.886
Balanced RF	79.0	66.4	73.5	82.6	0.871
SMOTE + RF	71.1	81.9	80.9	67.1	0.834
ADASYN + RF	68.6	80.8	82.4	69.6	0.832
RUS + RF	42.1	94.5	95.6	39.4	0.704
SMOTE-Tomek + RF	71.1	81.5	79.4	65.9	0.827
SMOTEENN + RF	59.3	83.0	83.8	58.5	0.781
SMOTE+RUS + RF	75.7	81.5	82.4	73.7	0.853
SVMSMOTE + RF	75.3	76.8	79.4	76.6	0.854

Synthetic oversampling and hybrid methods, including SMOTE, ADASYN, SMOTE-Tomek, and SMOTEENN, generally increased recall but reduced precision, which prevented a net improvement in F1-score. For instance, SMOTE + RF achieved a higher CV recall of 81.9% than Baseline RF, but its CV F1-score fell to 71.1%, which remained considerably lower than the Baseline RF. This indicates that synthetic samples generated by interpolation may introduce additional noise into the feature space and disturb the decision boundaries learned by the Random Forest model.

RUS + RF represents the most extreme precision-recall trade-off. Although it achieved the highest CV recall of 94.5% and the highest test recall of 95.6%, its CV F1-score fell to 42.1% and its test F1-score remained only 39.4%. This means that the model detected many failure cases but also generated a large number of false alarms. Therefore, recall alone is not sufficient for evaluating model performance in imbalanced predictive maintenance tasks. Among the sampling-based configurations, SVMSMOTE + RF and SMOTE+RUS + RF achieved relatively better test F1-scores of 76.6% and 73.7%, respectively; however, they still did not outperform the Baseline RF. Overall, these findings suggest that, for the AI4I 2020 dataset, complex sampling strategies do not improve Random Forest performance compared with the baseline configuration.

3.2. Hyperparameter Optimization Results (RF)

The two highest-performing configurations, Baseline RF and Balanced RF, were selected for subsequent hyperparameter optimization based strictly on their cross-validation F1-scores evaluated within the training partition. The held-out test set remained completely unseen during this selection process. Hyperparameter optimization was conducted using RandomizedSearchCV within the training partition, with inner-fold cross-validation and F1-score as the optimization criterion. Table 4 presents the optimal hyperparameter combinations, while Table 5 compares the post-optimization test set performance of the selected RF models.

This tuning step was performed to examine whether hyperparameter optimization could further improve the already strong performance of Random Forest on the original and class-weighted data distributions. Since Baseline RF and Balanced RF achieved the two best results in the initial RF comparison, focusing on these configurations provides a consistent and data-driven basis for evaluating the effect of tuning.

Table 4. Optimal hyperparameters identified by RandomizedSearchCV for Random Forest

Parameter	Tuned Baseline RF	Tuned Balanced RF
n_estimators	200	400
max_depth	None (unlimited)	20
min_samples_split	2	2
min_samples_leaf	1	2
max_features	log2	sqrt
class_weight	None	balanced

Table 5. Test set performance of the tuned Random Forest models

Model	Test Precision (%)	Test Recall (%)	Test F1 (%)	Test PR-AUC
Tuned Baseline RF	94.7	79.4	86.4	0.886
Tuned Balanced RF	93.1	79.4	85.7	0.875

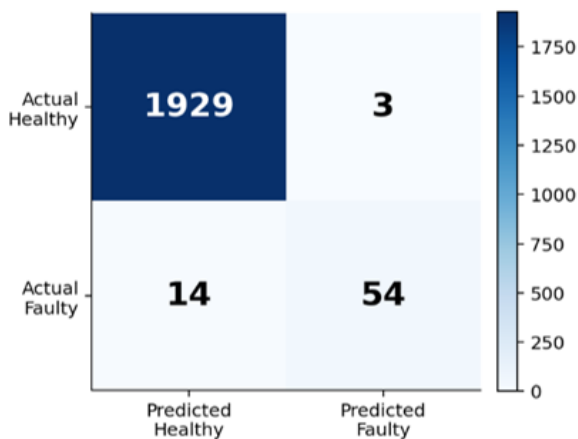


Figure 2. Confusion matrix of the Tuned Baseline RF model

As shown in Table 5, hyperparameter optimization did not alter the performance ranking observed in the initial RF comparison; Tuned Baseline RF retained its leading position. Notably, however, tuning produced a meaningful improvement for Balanced RF, raising its F1-score from 82.6% (pre-tuning) to 85.7%, while Baseline RF remained unchanged at 86.4%. Tuned Baseline RF remained the best-performing RF configuration, achieving a test F1-score of 86.4% and a PR-AUC of 0.886. Tuned Balanced RF achieved a very close F1-score of 85.7% and a PR-AUC of 0.875, with both tuned RF models reaching the same recall value of 79.4%. However, Tuned Baseline RF maintained slightly higher precision, indicating a better precision-recall balance. Figure 2 presents the confusion matrix of the Tuned Baseline RF model on the test set. Overall, these results suggest that Random Forest performs strongly on the original data distribution, while class-weight adjustment provides a competitive but not superior alternative for this dataset. The corresponding precision-recall curve is shown in Figure 3.

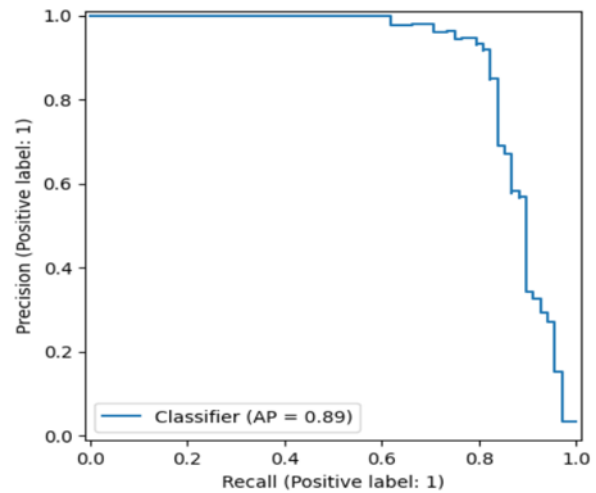


Figure 3. Precision-recall curve of the Tuned Baseline RF model

3.3. XGBoost Results

Table 6 presents the cross-validation and test set performance of all nine XGBoost configurations. To maintain consistency with the Random Forest results, the table reports cross-validation metrics first, followed by test set metrics. Since the dataset is severely imbalanced, the comparison focuses on recall, F1-score, and PR-AUC rather than ROC-AUC. This allows the evaluation to emphasize minority-class detection and the precision-recall trade-off more directly.

Table 6. Cross-validation and test set performance of XGBoost configurations

Model	CV F1 (%)	CV Recall (%)	Test Recall (%)	Test F1 (%)	Test PR-AUC
Baseline XGB	69.8	61.2	70.0	75.4	0.833
Balanced XGB	70.0	69.1	77.1	77.7	0.811
SMOTE + XGB	63.9	74.1	84.3	71.9	0.843
ADASYN + XGB	63.5	76.3	82.9	67.8	0.811
RUS + XGB	35.9	89.9	95.7	43.4	0.648
SMOTE-Tomek + XGB	64.6	75.9	84.3	73.3	0.859
SMOTEENN + XGB	52.9	79.9	91.4	64.0	0.823
SMOTE+RUS + XGB	65.9	74.5	78.6	75.9	0.861
SVMSMOTE + XGB	72.1	75.2	81.4	79.2	0.871

In contrast to the Random Forest results, XGBoost benefited more clearly from selected sampling strategies. SVMSMOTE + XGB achieved the best overall performance before hyperparameter tuning, with the highest test F1-score of 79.2% and the highest test PR-AUC of 0.871. This suggests that, unlike Random Forest, XGBoost can benefit from synthetic augmentation of samples located near decision boundaries through more sophisticated techniques such as SVMSMOTE.

The results also show why recall alone should not be used as the primary performance criterion. RUS + XGB achieved the highest test recall of 95.7%, indicating that it detected most failure cases. However, its test F1-score remained only 43.4%, and its PR-AUC dropped to 0.648, showing that the model produced too many false alarms. Similarly, SMOTEENN + XGB reached a high recall of 91.4%, but its F1-score remained limited at 64.0%. Therefore, a high recall value does not necessarily indicate a well-balanced model in imbalanced predictive maintenance problems.

Among the more balanced configurations, SVMSMOTE + XGB provided the strongest precision-recall trade-off, while SMOTE+RUS + XGB achieved the second highest PR-AUC value of 0.861. Overall, these findings indicate that sampling strategies affect XGBoost differently from Random Forest. While complex sampling methods degraded or failed to improve Random Forest performance, boundary-aware and hybrid sampling approaches provided more competitive results for XGBoost before hyperparameter optimization.

3.4. Hyperparameter Optimization Results (XGBoost)

Based on their initial cross-validation F1-scores evaluated strictly on the training partition, the two top-performing XGBoost configurations (Balanced and SVMSMOTE) were selected for subsequent hyperparameter optimization using RandomizedSearchCV. Table 7 presents the winning parameter combinations and Table 8 compares their post-optimization test set performance. The search space spanned `n_estimators`, `max_depth`, `learning_rate`,

`subsample`, `colsample_bytree`, `min_child_weight`, and `gamma`—parameters governing XGBoost’s tree construction depth, stochastic column and row subsampling, and leaf impurity regularization. The optimal hyperparameter combinations differed between the two tuned XGBoost configurations, indicating that the upstream balancing strategy affected the preferred model complexity. Tuned Balanced XGB selected a shallower and faster-learning configuration, whereas Tuned SVMSMOTE + XGB favored a larger ensemble with deeper trees and a lower learning rate. This suggests that boundary-aware synthetic sampling benefited from a more expressive boosted-tree structure combined with shrinkage regularization.

Table 7. Optimal hyperparameters identified by RandomizedSearchCV for XGBoost

Parameter	Tuned Balanced XGB	Tuned SVMSMOTE + XGB
<code>n_estimators</code>	200	500
<code>max_depth</code>	5	8
<code>learning_rate</code>	0.1	0.05
<code>subsample</code>	0.7	0.9
<code>colsample_bytree</code>	0.8	0.7
<code>min_child_weight</code>	1	1
<code>gamma</code>	0.1	0.1

Table 8. Test set performance of the tuned XGBoost models

Model	Test Precision (%)	Test Recall (%)	Test F1 (%)	Test PR-AUC
Tuned Balanced XGB	81.2	82.4	81.8	0.856
Tuned SVMSMOTE + XGB	79.2	87.1	83.0	0.914

Hyperparameter tuning produced clear improvements for both selected XGBoost configurations. Tuned Balanced XGB achieved a test F1-score of 81.8% and a PR-AUC of 0.856, improving over its pre-tuning performance. However, Tuned SVMSMOTE + XGB achieved the strongest overall performance among the tuned XGBoost models, with a test F1-score of 83.0% and the highest PR-AUC of 0.914. It also provided the highest recall value of 87.1%, indicating stronger failure detection capability.

The comparison between the two tuned models reveals a clear precision-recall trade-off. Tuned Balanced XGB achieved slightly higher precision, whereas Tuned SVMSMOTE + XGB achieved higher recall, F1-score, and PR-AUC. This indicates that boundary-aware synthetic sampling, when combined with hyperparameter optimization, provided a more effective balance between detecting failure cases and maintaining overall minority-class performance. Therefore, unlike Random Forest, for which the baseline configuration remained dominant, XGBoost benefited more from the combination of class-balancing strategy and hyperparameter tuning.

3.5. SHAP-Based Interpretability Results

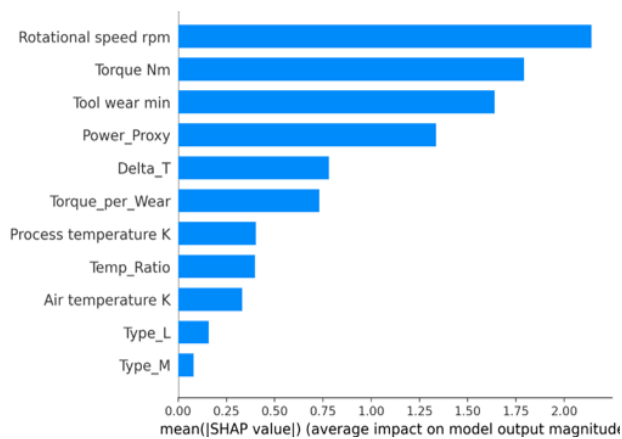


Figure 4. Mean absolute SHAP values of the Tuned SVMSMOTE + XGB model

Figure 4 presents the mean absolute SHAP values for the Tuned SVMSMOTE + XGB model. The results show that Rotational Speed, Torque, and Tool Wear are the most influential original sensor variables in the model's decision process. Among the engineered features, Power_Proxy, Delta_T, and Torque_per_Wear also contribute substantially, indicating that domain-informed feature engineering added useful predictive information beyond the raw sensor measurements. Overall, the SHAP analysis supports the practical relevance of both the original operating variables and the physically motivated engineered features in predicting machine failure.

3.6. Discussion

The most consequential finding of this study is that the effectiveness of class balancing strategies is strongly dependent on the learning algorithm. For Random Forest, the baseline and class-weighted configurations remained dominant. Tuned Baseline RF achieved the best RF performance with an F1-score of 86.4% and a PR-AUC of 0.886, while Tuned Balanced RF achieved a very close F1-score of 85.7% and a PR-AUC of 0.875. These results suggest that the bagging-based structure of Random Forest provides a degree of robustness under class imbalance, and that complex synthetic sampling strategies are not necessarily beneficial for this model.

For XGBoost, the pattern was different. Before hyperparameter optimization, SVMSMOTE + XGB achieved the strongest XGBoost performance, with a test F1-score of 79.2% and a PR-AUC of 0.871. After tuning, Tuned SVMSMOTE + XGB remained the strongest tuned XGBoost configuration, achieving an F1-score of 83.0% and the highest PR-AUC of 0.914. Tuned Balanced XGB also improved after optimization, reaching an F1-score of 81.8% and a PR-AUC of 0.856. These findings indicate that XGBoost benefits more from the combination of boundary-aware synthetic sampling and hyperparameter optimization than Random Forest.

The metric-level results further show that recall alone is not sufficient for model selection in imbalanced predictive maintenance. For example, RUS + XGB achieved the highest test recall of 95.7%, but its F1-score remained only 43.4%, and its PR-AUC dropped to 0.648. This indicates that the model detected many failure cases but generated too many false alarms. Similarly, RUS + RF achieved the highest CV recall among RF configurations, but its precision and F1-score were substantially lower than those of Baseline RF. Therefore, F1-score and PR-AUC provide a more reliable basis for evaluating minority-class performance than recall alone.

The SHAP analysis provides an additional interpretability perspective for the final tuned XGBoost model. Rotational Speed, Torque, and Tool Wear emerged as the most influential original sensor variables, while Power_Proxy, Delta_T, and Torque_per_Wear were the most relevant engineered features. This supports the practical relevance of the feature engineering step and shows that the model relies on physically meaningful operating variables rather than arbitrary patterns. For industrial PdM applications, this interpretability is important because maintenance decisions should be understandable to engineers and operators.

From a practical perspective, the results suggest that class balancing strategies should not be selected independently of the learning algorithm. For Random Forest, the baseline configuration should be considered a strong starting point, and class-weight adjustment may be used as a competitive alternative. For XGBoost, boundary-aware sampling strategies such as SVMSMOTE can provide additional benefit, especially when combined with hyperparameter optimization. Overall, the findings support a metric-aware and interpretable PdM pipeline in which sampling strategy, learning algorithm, hyperparameter tuning, and explainability are evaluated together rather than in isolation.

4. CONCLUSIONS

This study conducted a systematic comparison of nine class balancing strategies applied to Random Forest and XGBoost on the AI4I 2020 Predictive Maintenance Dataset. The results demonstrate that the effectiveness of class balancing strategies is algorithm dependent. For Random Forest, the baseline and class-weighted configurations produced the strongest results. Tuned Baseline RF achieved the best RF performance with an F1-score of 86.4% and a PR-AUC of 0.886, while Tuned Balanced RF achieved a comparable F1-score of 85.7% and a PR-AUC of 0.875. These findings indicate that Random Forest can remain robust under severe class imbalance without requiring complex synthetic resampling strategies.

For XGBoost, the results showed a different pattern.

Boundary-aware sampling combined with hyperparameter optimization provided the most effective tuned XGBoost performance. Tuned SVM SMOTE + XGB achieved an F1-score of 83.0% and the highest PR-AUC of 0.914, outperforming Tuned Balanced XGB in terms of recall, F1-score, and PR-AUC. This confirms that XGBoost benefits more from the combination of sampling strategy and hyperparameter optimization than Random Forest.

The study also confirms that PR-AUC and F1-score are more informative than accuracy or ROC-AUC under severe class imbalance. High recall alone was shown to be insufficient, since some configurations achieved high recall by producing excessive false alarms. Therefore, model selection in imbalanced predictive maintenance should consider the full precision-recall trade-off rather than relying on a single metric.

In addition, SHAP-based interpretability analysis showed that Rotational Speed, Torque, and Tool Wear were the most influential original sensor variables, while Power_Proxy, Delta_T, and Torque_per_Wear were the most relevant engineered features. This supports the practical value of domain-informed feature engineering and improves the transparency of the final tuned XGBoost model.

Overall, the findings suggest that class balancing strategies should not be selected independently of the learning algorithm. For industrial predictive maintenance pipelines, Random Forest can serve as a strong and robust baseline, while XGBoost may benefit from boundary-aware sampling and careful hyperparameter optimization. Future work should extend this framework to additional industrial datasets, investigate cost-sensitive threshold optimization, and evaluate interpretable deep learning models for sequential sensor data.

Declaration of Ethical Standards

The authors declare that this research complied with all applicable ethical guidelines. No experiments involving human subjects or animals were conducted.

Credit Authorship Contribution Statement

Berk Mercan: Conceptualization, Methodology, Software, Formal Analysis, Investigation, Data Curation, Visualization, Literature Review, Writing – Original Draft, Writing – Review and Editing.

Eren Açar: Methodology, Software, Formal Analysis, Validation, Data Curation, Investigation, Writing – Review and Editing.

Mert Çimentepe: Literature Review, Validation, Writing – Review and Editing.

Meltem Apaydın Üstün: Supervision, Conceptualization, Methodology, Project Administration, Writing – Review and Editing.

Declaration of Competing Interest

The authors declare no competing financial or personal interests.

Funding / Acknowledgements

This study was supported by the Scientific and Technological Research Council of Türkiye (TÜBİTAK) under the 2209-A University Students Research Projects Support Program, 2024/1 call, project number 1919B012470616. The authors gratefully acknowledge TÜBİTAK for its support. The responsibility for the content of this publication belongs solely to the authors. Outputs produced within the scope of the program will be reported to TÜBİTAK through the BİDEB Application and Monitoring System.

Data Availability

The AI4I 2020 Predictive Maintenance Dataset used in this study is publicly available through the UCI Machine Learning Repository at <https://doi.org/10.24432/C5HS5C>.

References

- [1] I. Hector and R. Panjanathan, "Predictive maintenance in Industry 4.0: A survey of planning models and machine learning techniques," *PeerJ Computer Science*, vol. 10, Art. no. e2016, 2024. doi: 10.7717/peerj-cs.2016
- [2] T. P. Carvalho et al., "A systematic literature review of machine learning methods applied to predictive maintenance," *Computers & Industrial Engineering*, vol. 137, Art. no. 106024, 2019. doi: 10.1016/j.cie.2019.106024
- [3] S. Matzka, "AI4I 2020 predictive maintenance dataset," UCI Machine Learning Repository, 2020. doi: 10.24432/C5HS5C
- [4] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Proc. 31st Int. Conf. Neural Information Processing Systems (NIPS)*, Long Beach, CA, USA, 2017, pp. 4765–4774. arXiv:1705.07874.
- [5] O. E. Hassan et al., "Induction motor broken rotor bar fault detection techniques based on fault signature analysis – a review," *IET Electric Power Applications*, vol. 12, no. 7, pp. 895–907, 2018. doi: 10.1049/iet-epa.2018.0054
- [6] D. Neupane et al., "Data-driven machinery fault diagnosis: A comprehensive review," *Neurocomputing*, vol. 627, Art. no. 129588, 2025. doi: 10.1016/j.neucom.2025.129588
- [7] A. S. Kalafatelis et al., "An effective methodology for imbalanced data handling in predictive maintenance for offset printing," in *Proc. 11th Int. Conf. Mechatronics and Control Engineering (ICMCE 2023)*, Lecture Notes in Mechanical Engineering, Springer, Singapore, 2024, pp. 89–98. doi: 10.1007/978-981-99-6523-6_7
- [8] A. Hakami, "Strategies for overcoming data scarcity, imbalance, and feature selection challenges in machine learning models for predictive maintenance," *Scientific Reports*, vol. 14, Art. no. 9645, 2024. doi: 10.1038/s41598-024-59958-9
- [9] K. Patel and A. Shanbhag, "Exploring ML for predictive maintenance using imbalance correction techniques and SHAP," in *Proc. 2022 International Conference on Electrical, Computer and Energy Technologies (ICECET)*, Prague, Czech Republic, 2022, pp. 1–10. doi: 10.1109/ICECET55527.2022.9873073
- [10] S. H. H. Zaidi et al., "A systematic review of anomaly and fault detection using machine learning for industrial machinery," *Algorithms*, vol. 19, no. 2, Art. no. 108, 2026. doi: 10.3390/a19020108
- [11] A. Borré et al., "Machine fault detection using a hybrid CNN-LSTM attention-based model," *Sensors*, vol. 23, no. 9, Art. no. 4512, 2023. doi: 10.3390/s23094512
- [12] W. Li and T. Li, "Comparison of deep learning models for predictive maintenance in industrial manufacturing systems using sensor data," *Scientific Reports*, vol. 15, no. 1, Art. no.

- 23545, 2025. doi: 10.1038/s41598-025-08515-z
- [13] K. M. A. Alghtus et al., "Short-horizon predictive maintenance of industrial pumps using time-series features and machine learning," arXiv preprint arXiv:2508.19974, 2025. doi: 10.48550/arXiv.2508.19974
- [14] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001. doi: 10.1023/A:1010933404324
- [15] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*, San Francisco, CA, USA, 2016, pp. 785–794. doi: 10.1145/2939672.2939785
- [16] F. Velasco-Loera, M. Alcaraz-Mejia, and J. L. Chavez-Hurtado, "An interpretable hybrid fault prediction framework using XGBoost and a probabilistic graphical model for predictive maintenance: A case study in textile manufacturing," *Applied Sciences*, vol. 15, no. 18, Art. no. 10164, 2025. doi: 10.3390/app151810164
- [17] Y. Altintas, *Manufacturing Automation: Metal Cutting Mechanics, Machine Tool Vibrations, and CNC Design*, 2nd ed. Cambridge, U.K.: Cambridge Univ. Press, 2012. doi: 10.1017/CBO9780511843723
- [18] N. V. Chawla et al., "SMOTE: Synthetic minority over-sampling technique," *Journal of Artificial Intelligence Research*, vol. 16, pp. 321–357, 2002. doi: 10.1613/jair.953.
- [19] H. He et al., "ADASYN: Adaptive synthetic sampling approach for imbalanced learning," in *Proc. IEEE Int. Joint Conf. Neural Networks (IJCNN)*, Hong Kong, 2008, pp. 1322–1328. doi: 10.1109/IJCNN.2008.4633969.
- [20] H. M. Nguyen et al., "Borderline over-sampling for imbalanced data classification," *International Journal of Knowledge Engineering and Soft Data Paradigms*, vol. 3, no. 1, pp. 4–21, 2011. doi: 10.1504/IJKESDP.2011.039875
- [21] G. Lemaitre et al., "Imbalanced-learn: A Python toolbox to tackle the curse of imbalanced datasets in machine learning," *Journal of Machine Learning Research*, vol. 18, no. 17, pp. 1–5, 2017. [Online]. Available: <http://jmlr.org/papers/v18/16-365.html>.