Computers, Materials & Continua

Tech Science Press

Doi:10.32604/cmc.2025.073189

ARTICLE

# Engine Failure Prediction on Large-Scale CMAPSS Data Using Hybrid Feature Selection and Imbalance-Aware Learning

**Ahmad Junaid[1], Abid Iqbal[2,*], Abuzar Khan[1], Ghassan Husnain[1,*], Abdul-Rahim Ahmad[3] and Mohammed Al-Naeem[4]**

[1]Department of Computer Science, CECOS University of IT and Emerging Sciences, Peshawar, 25000, Pakistan
[2]Department of Computer Engineering, College of Computer Sciences and Information Technology, King Faisal University, Al Ahsa, 31982, Saudi Arabia
[3]Department of Information Systems, King Faisal University, Al-Hofuf, 31982, Saudi Arabia
[4]Department of Computer Networks Communications, CCSIT, King Faisal University, Al Ahsa, 31982, Saudi Arabia

*Corresponding Authors: Abid Iqbal. Email: aaiqbal@kfu.edu.sa; Ghassan Husnain. Email: ghassan.husnain@gmail.com

Received: 12 September 2025; Accepted: 02 December 2025; Published: 10 February 2026

**ABSTRACT:** Most predictive maintenance studies have emphasized accuracy but provide very little focus on Interpretability or deployment readiness. This study improves on prior methods by developing a small yet robust system that can predict when turbofan engines will fail. It uses the NASA CMAPSS dataset, which has over 200,000 engine cycles from 260 engines. The process begins with systematic preprocessing, which includes imputation, outlier removal, scaling, and labelling of the remaining useful life. Dimensionality is reduced using a hybrid selection method that combines variance filtering, recursive elimination, and gradient-boosted importance scores, yielding a stable set of 10 informative sensors. To mitigate class imbalance, minority cases are oversampled, and class-weighted losses are applied during training. Benchmarking is carried out with logistic regression, gradient boosting, and a recurrent design that integrates gated recurrent units with long short-term memory networks. The Long Short-Term Memory–Gated Recurrent Unit (LSTM–GRU) hybrid achieved the strongest performance with an F1 score of 0.92, precision of 0.93, recall of 0.91, Receiver Operating Characteristic–Area Under the Curve (ROC-AUC) of 0.97, and minority recall of 0.75. Interpretability testing using permutation importance and Shapley values indicates that sensors 13, 15, and 11 are the most important indicators of engine wear. The proposed system combines imbalance handling, feature reduction, and Interpretability into a practical design suitable for real industrial settings.

**KEYWORDS:** Predictive maintenance; CMAPSS dataset; feature selection; class imbalance; LSTM-GRU hybrid model; interpretability; industrial deployment

## 1 Introduction

Predictive Maintenance (PdM) is a crucial component of Industry 4.0, enabling companies to keep machines running longer while reducing unexpected breakdowns and associated costs [1]. By predicting the Remaining Useful Life (RUL) or detecting failures before they occur, PdM shifts maintenance from reactive or fixed schedules to condition-based actions [2]. However, even with the large amount of sensor data available in modern industries, three key problems still limit the strength of PdM models: insufficient failure data, an imbalance between healthy and faulty cases and the difficulty of selecting the right features [3]. Failures are rare, so datasets typically have only a small number of faulty examples compared to thousands of regular cycles, which prompts models to focus on the majority class [4]. Additionally, the high-frequency

Copyright © 2026 The Authors. Published by Tech Science Press.
This work is licensed under a Creative Commons Attribution 4.0 International License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

sensors generate large amounts of complex, noisy signals, necessitating the selection of a smaller set of valuable features to enhance clarity, speed, and reliability [5]. These problems are exacerbated by the scarcity of failure cases, which makes it harder to test models properly and confirm that results are trustworthy [4].

Recent work has focused on deep learning methods such as Long Short-Term Memory (LSTMs), Gated Recurrent Units (GRUs), and Convolutional Neural Networks (CNNs), which can capture complex temporal patterns in sensor data [6]. These models achieve high accuracy by learning how systems degrade [7], but they are often treated as black boxes and usually lack modular pipelines that support reproducibility or real-world use [8]. In addition, most studies have placed more emphasis on accuracy while paying less attention to explainability, efficiency and solutions for class imbalance, which limits their value in industrial settings [9]. This gap highlights the need for PdM frameworks that combine the predictive strength of deep learning with clarity, practical efficiency, and strong validation In parallel, prescriptive maintenance studies formalize decision rules, inspection strategies and reliability-aware scheduling that convert predictions into actions, offering a template for linking RUL and calibrated risk to maintenance policies [10].

This study addresses that need by presenting a hybrid ensemble pipeline designed for real-world use in fields such as aerospace and manufacturing [10]. Unlike earlier work that handles feature design, model training and testing as separate tasks, the workflow brings them together into a single process [5]. It combines advanced preprocessing with strong feature selection methods, such as variance thresholding, recursive elimination and tree-based importance, alongside a stacked LSTM–GRU model for sequential data. Classical models, such as logistic regression and Extreme Gradient Boosting (XGBoost), are tested for a fair comparison, and the feature selection ensemble is used as a baseline to assess the impact of feature reduction on performance. To handle imbalance, the pipeline uses RandomOverSampler and class-weighted losses [11]. For reliability, it uses stratified and time-based splits, nested cross-validation, bootstrap confidence intervals, and nonparametric significance tests [4]. For Interpretability, both global and local tools are used, including permutation importance, SHapley Additive exPlanations (SHAP) and Local Interpretable Model-agnostic Explanations (LIME), which help highlight the key drivers of system failures [12]. Additionally, inference speed and resource utilization are measured to confirm that the pipeline is efficient enough for deployment [13].

By bringing these parts together, this research makes three main contributions. First, it demonstrates a scalable approach to transforming large, partially labelled sensor data into balanced, useful feature sets [14]. Second, it proves that hybrid feature selection with a stacked LSTM–GRU improves early warning accuracy compared to classical models [15]. Third, it provides a practical blueprint that includes explainability, strong statistical checks, and performance profiling for real deployment [16]. Overall, these advances deliver a PdM framework that is robust, interpretable, and efficient, helping accelerate the transition from research to industrial use [17].

***Motivation and Research Objectives:***

Despite the promise of PdM in improving asset availability and reducing downtime, its adoption is limited by scarce and imbalanced failure data, noisy high-dimensional signals and black-box deep models with limited Interpretability. Many studies emphasize accuracy while overlooking transparency, validation and reproducibility, which hinders deployment in safety-critical domains. To close this gap, this research pursues three objectives:

- **RO1:** Develop a dual-task PdM system that combines Remaining Useful Life (RUL) regression with early-warning classification.
- **RO2:** Design an interpretable and imbalance-aware temporal learning framework with hybrid feature selection and robust resampling.

- **RO3:** Ensure deployment-readiness through rigorous evaluation and explainability using SHAP, LIME and permutation importance.

Table 1 provides the full forms and definitions of all acronyms used throughout the manuscript.

**Table 1:** List of acronyms

| Acronym | Full form |
|---|---|
| **PdM** | Predictive Maintenance |
| **RUL** | Remaining Useful Life |
| **LSTM** | Long Short-Term Memory |
| **GRU** | Gated Recurrent Unit |
| **CNN** | Convolutional Neural Network |
| **SHAP** | SHapley Additive exPlanations |
| **LIME** | Local Interpretable Model-Agnostic Explanations |
| **SMOTE** | Synthetic Minority Over-Sampling Technique |

## 2 Literature Review

Early studies on PdM have relied on process control charts and simple classifiers applied to narrow datasets, which limits their ability to capture complex degradation patterns [6]. The release of the NASA CMAPSS turbofan benchmark marked a turning point, becoming the most widely studied RUL dataset and driving a surge in deep learning models [18]. However, many CMAPSS-based works still rely on single-split evaluations, which can lead to overfitting and limit generalization [19]. A central challenge also lies in handling class imbalance, since failure events are far rarer than regular operation. While SMOTE has improved minority recall, it may introduce noise, prompting newer variants such as the Distance-based Extended Synthetic Minority Over-Sampling Technique (Distance-ExtSMOTE), the Dirichlet-based Extended Synthetic Minority Over-Sampling Technique (Dirichlet-ExtSMOTE), and the Dirichlet-based Extended Synthetic Minority Over-Sampling Technique (BGMM-SMOTE) that better preserve class boundaries [20]. Hybrid approaches that combine oversampling, cost-sensitive learning and ensembles often outperform single remedies [21], although most PdM studies still benchmark imbalance methods in isolation rather than against algorithmic alternatives, such as class-weighted or focal loss.

Beyond imbalance, feature selection has progressed from univariate filters to model-aware methods [22]. Mutual information remains popular for improving efficiency, while wrappers such as Recursive Feature Elimination (RFE) provide finer discrimination at a higher computational cost [23]. Explainable tools, such as SHAP, have further bridged the gap between interpretation and selection, enabling the removal of redundant features while retaining accuracy on industrial data [24]. Few works, however, compare SHAP directly with classical L1 or tree-based filters to build stronger ensembles. On the modelling side, gradient-boosted trees, such as Extreme Gradient Boosting (XGBoost), Light Gradient Boosting Machine (LightGBM) and Categorical Boosting (CatBoost), offer strong baselines with built-in Interpretability [25]. XGBoost tuned with particle swarm optimization has also achieved state-of-the-art RUL accuracy on the CMAPSS dataset [26], and it is increasingly paired with SHAP for deployment on shop floors. For sequential sensor data, LSTM and GRU dominate and hybrids that combine both capture long- and short-term dependencies, improving early fault detection when paired with feature selection and balanced training [27].

Despite these advances, two gaps remain. First, few studies integrate diverse imbalance remedies, hybrid feature selectors and both tree-based and deep models into a single reproducible framework, making attribution of gains difficult [28]. Second, statistical rigour is inconsistent, as nested cross-validation, uncertainty quantification, and nonparametric significance testing are rarely applied, raising concerns about overfitting [29]. Recent trends toward zero-defect manufacturing demand PdM systems that are accurate, interpretable and resource-efficient [30]. This has increased interest in lightweight models for edge devices, as well as neurosymbolic and attention-based designs that enhance transparency [31]. However, a holistic evaluation of latency, memory, and uncertainty remains scarce, underscoring the need for pipelines that are not only high-performing but also transparent, statistically rigorous, and operationally ready for deployment in aerospace, railways, and advanced manufacturing [32]. We follow this direction by using model outputs as inputs to decision frameworks for inspection timing, resource allocation and reliability-aware operations, as outlined in prescriptive maintenance and reliability planning studies

## 3 Methodology

This study proposes a PdM pipeline on the CMAPSS dataset with cleaned sensor data, hybrid feature selection, classical and LSTM-GRU models, imbalance handling using RandomOverSampler, rigorous cross-validation, and Interpretability via SHAP, LIME, and permutation importance to support deployment readiness.

### *3.1 Dataset Overview*

This study utilises the CMAPSS dataset from NASA's Prognostics Centre of Excellence [33], which provides multivariate time series from simulated turbofan engines under four distinct fault scenarios. Each engine cycle includes 26 features: an ID, cycle index, three operating parameters and 21 sensor signals. Training engines run until failure, while test runs are truncated, with RUL vectors supplied for evaluation. Table 2 shows the health-state distribution: most cycles are Healthy with RUL greater than 100, fewer are degraded with RUL between 20 and 100, and only a small fraction are Failure Imminent with RUL less than 20. This imbalance complicates early detection, so the pipeline applies resampling and class-weighted learning:

**Table 2:** Distribution of health states in the CMAPSS dataset

| Health state | RUL range | Cycle count |
|---|---|---|
| Failure imminent | <20 | 15,000 |
| Degraded | 20–100 | 57,000 |
| Healthy | >100 | 90,000 |

For the classification task, a data point is labelled near-failure when RUL $\leq \boldsymbol{\delta}$. We select $\boldsymbol{\delta}$ by grid search over candidate values from 10, 15, 20, 25, 30 cycles using time-aware, unit-stratified cross-validation. For each $\boldsymbol{\delta}$, labels are rebuilt from Eq. (6), models are retrained and performance is evaluated with PR-AUC and a cost-sensitive $\boldsymbol{F_\beta}$ score ($\boldsymbol{\beta > 1}$ to penalise missed failures). The chosen $\boldsymbol{\delta^*}$ maximises PR-AUC tie-broken by $\boldsymbol{F_2}$ under the application's minimum precision requirement. This procedure links the threshold directly to operational risk while avoiding temporal leakage.

### *3.2 Mathematical Representation*

*3.2.1 Preprocessing*

Sensor signals in the CMAPSS dataset often contain noise and anomalies that can bias the training process. To address this, preprocessing includes outlier removal, normalization and target definition.

Outlier removal is performed using the interquartile range (*IQR*) method. The first and third quartiles are defined as $Q_1$ and $Q_3$ and the interquartile range is given by

$$IQR = Q_3 - Q_1 \tag{1}$$

The lower and upper bounds for acceptable values are then computed as:

$$L = Q_1 - k \times IQR \tag{2}$$

$$U = Q_3 + k \times IQR \tag{3}$$

where $k$ is a constant, set to 1.5 for moderate outliers or 3.0 for extreme outliers. Any observation $x_i$ is considered an outlier if:

$$x_i < Q_1 - k\,(Q_3 - Q_1)\; or x_i > Q_3 + k\,(Q_3 - Q_1) \tag{4}$$

This step removes extreme values that do not represent actual engine degradation patterns.

Although the CMAPSS data are simulated, the sensor streams intentionally include random perturbations and operational transients to emulate measurement uncertainty rather than true degradation. Unchecked, these fluctuations can bias feature scaling and inflate variance-based selection. Therefore, moderate outlier suppression is applied to preserve underlying degradation dynamics while removing artifacts that do not correspond to physical fault progression.

Normalization is then applied through z-score scaling, which ensures stable convergence by standardizing all features to zero mean and unit variance:

$$x' = (x - \mu)/\sigma \tag{5}$$

where μ and σ are the mean and standard deviation of each feature.

Among several normalization options, z-score scaling was selected over min–max and robust scaling after comparative testing on CMAPSS sensor distributions. Min–max scaling compressed sensor variance and magnified transient noise, while robust scaling underperformed in multimodal operating regimes. Z-score normalization preserved variance structure and ensured consistent convergence across engines with different regimes, yielding more stable training and calibration. Finally, the regression target is defined as the remaining useful life (*RUL*) of each engine unit. The *RUL* at cycle t for unit $u$ is computed as:

$$RUL\,(u, t) = \max(\text{cycle of unit } u) - t \tag{6a}$$

This provides the time remaining until engine failure and serves as the target variable for both regression and classification tasks. The near-failure class is defined from Eq. (6b) as:

$$y_{u,t} = 1RUL\,(u, t) \le \delta \tag{6b}$$

With $\delta$ selected as described in this Section 3.1.

*3.2.2 Feature Selection*

Effective feature selection is crucial in predictive maintenance, as sensor streams are often high-dimensional and redundant. Reducing dimensionality not only improves computational efficiency but also enhances Interpretability and reduces the risk of overfitting. To achieve this, we apply a hybrid ensemble that integrates statistical, wrapper-based and boosting-driven criteria.

First, the variance threshold method eliminates features with minimal variability, as such features contribute little discriminatory power. The variance of the feature $X_j$ across $n$ samples is given by:

$$Var\left(X_j\right) = \frac{1}{n}\sum_{i=1}^{n}\left(x_{ij} - \overline{x_j}\right)^2 \tag{7}$$

where $x_{ij}$ represents the value of feature $j$ in sample $i$ and $\overline{x_j}$ is the mean of feature $j$. Features with $Var\left(X_j\right)$ below a fixed threshold are discarded, as they provide negligible separation between healthy and faulty states.

Next, Recursive Feature Elimination (RFE) is applied. At each iteration, a model assigns weights $w_j$ to features and the feature with the smallest absolute weight magnitude $\left|w_j\right|$ is removed. This process is repeated until the optimal subset is retained. The selection criterion can be expressed as:

$$S_{t+1} = S_t \backslash \{\arg\min_{j \in S_t} \left|w_j\right|\} \tag{8}$$

where $S_t$ denotes the active feature set at iteration $t$.

Finally, XGBoost quantifies feature importance by evaluating the average reduction in loss (gain) attributed to each feature when it is used to split decision trees. The gain for feature $f$ is defined as:

$$Gain\left(f\right) = \frac{1}{K}\sum_{k=1}^{K}\Delta L_f^{(k)} \tag{9}$$

where $\Delta L_f^{(k)}$ denotes the reduction in loss from using feature $f$ at split $k$ and $K$ is the total number of splits across all trees.

By combining these methods, the ensemble balances statistical stability, predictive utility and nonlinear interaction capture. The resulting consensus feature set retains the most informative sensors across folds and fault conditions, improving robustness while reducing dimensionality. Global importance measures such as SHAP and permutation scores guide feature refinement, while local explanations from LIME support cost-sensitive evaluation, yielding a compact and interpretable model for reliable engine failure prediction.

*3.2.3 Hybrid Recurrent Model (LSTM–GRU)*

Sequential dependencies in turbofan sensor streams require models that can capture both short-term fluctuations and long-term degradation trends. To achieve this, we employ a hybrid recurrent neural network that integrates Gated Recurrent Units (GRUs) and Long Short-Term Memory (LSTMs). The combined architecture leverages GRUs' efficiency for rapid adaptation and LSTMs' memory capacity for long-horizon learning.

The GRU component updates its hidden state using two gating functions. The update gate $z_t$ controls memory retention, while the reset gate $r_t$ determines how much past information should be forgotten. The candidate activation $\widetilde{h_t}$ encodes new information and the final hidden state $h_t$ merges both past and present signals:

$$z_t = \sigma\left(W_z x_t + U_z h_{t-1} + b_z\right) \tag{10}$$

$$r_t = \sigma\left(W_r x_t + U_r h_{t-1} + b_r\right) \tag{11}$$

$$\widetilde{h_t} = \tanh\left(W_h x_t + U_h\left(r_t \odot h_{t-1}\right) + b_h\right) \tag{12}$$

$$h_t = \left(1 - z_t\right) \odot h_{t-1} + z_t \odot \widetilde{h_t} \tag{13}$$

The LSTM component extends this by explicitly maintaining a memory cell $c_t$. It employs three gates: the forget gate $f_t$ for discarding irrelevant history, the input gate $i_t$ for incorporating new information and the output gate $o_t$ for regulating the exposure of the memory content. The dynamics are defined as:

$$f_t = \sigma\left(W_f x_t + U_f h_{t-1} + b_f\right) \tag{14}$$

$$i_t = \sigma\left(W_i x_t + U_i h_{t-1} + b_i\right) \tag{15}$$

$$\widetilde{c_t} = \tanh\left(W_c x_t + U_c h_{t-1} + b_c\right) \tag{16}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \widetilde{c_t} \tag{17}$$

$$o_t = \sigma\left(W_o x_t + U_o h_{t-1} + b_o\right) \tag{18}$$

$$h_t = o_t \odot \tanh\left(c_t\right) \tag{19}$$

In the hybrid pipeline, LSTM and GRU layers are stacked sequentially, allowing GRUs to capture rapid variations in sensor patterns while LSTMs preserve long-term degradation signals. The shared hidden state representation is then passed to fully connected layers for prediction. This integration allows the model to balance responsiveness (GRU) and stability (LSTM), producing robust forecasts of both imminent faults and gradual wear.

*3.2.4 Evaluation Metrics*

The predictive maintenance pipeline is evaluated using multiple complementary metrics to capture correctness, sensitivity and robustness under class imbalance.

***Precision*** quantifies the proportion of predicted failures that are correct:

$$Precision = \frac{TP}{TP + FP} \tag{20}$$

***Recall*** measures the proportion of actual failures correctly identified, which is critical in maintenance to minimize missed faults:

$$Recall = \frac{TP}{TP + FN} \tag{21}$$

***F1- score*** balances the trade-off between precision and recall, serving as a harmonic mean:

$$F1 = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall} \tag{22}$$

***Accuracy*** reflects the overall proportion of correctly classified samples:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{23}$$

here, $TP$, $TN$, $FP$ and $FN$ denote true positives, true negatives, false positives and false negatives, respectively. Together, these metrics ensure that the evaluation captures not only general performance but also the model's ability to detect rare failure events, which is essential for high-stakes industrial deployment.

*3.2.5 Interpretability*

Beyond accuracy, Interpretability is necessary to build trust in predictive maintenance models. This study incorporates global and local explanation methods.

Permutation Importance (*PI*) quantifies the sensitivity of predictions to perturbations of individual features. For a feature $f_j$, *PI* is computed as:

$$PI\left(f_j\right) = \frac{1}{n}\sum_{i=1}^{n}\left(L\left(y_i, \widehat{y_i}\right) - L\left(y_i, \widehat{y_i^{perm}}\right)\right) \tag{24}$$

where $L\left(y_i, \widehat{y_i}\right)$ is the loss for sample $i$ under the true model and $L\left(y_i, \widehat{y_i^{perm}}\right)$ is the loss after randomly permuting feature $f_j$. Higher values indicate stronger importance.

SHAP values provide additive feature attributions based on cooperative game theory. For feature $j$ in feature set $F$, the contribution is:

$$\phi_j = \sum_{S \subseteq F \setminus \{j\}} \frac{|S|!\left(|F| - |S| - 1\right)!}{|F|!}\left(f_{S\cup\{j\}}\left(x_{S\cup\{j\}}\right) - f_S\left(x_S\right)\right) \tag{25}$$

where $f_S$ is the model restricted to features in subset $S$. SHAP values capture both global rankings and local instance-level explanations.

LIME approximates the model locally with a linear surrogate function:

$$g\left(z'\right) = w_0 + \sum_j w_j z_j \tag{26}$$

where $z'$ is a perturbed binary vector representation of input features. The weights $w_j$ indicate the local influence of each feature for a specific prediction.

Together, permutation importance, SHAP and LIME provide complementary insights: permutation importance captures overall feature sensitivity, SHAP decomposes contributions at both global and local scales and LIME reveals case-specific explanations. Beyond model explanation, SHAP and LIME are integrated into the maintenance workflow, where their outputs identify sensor contributions to degradation patterns and support condition-based inspection planning during operation. This multi-method approach ensures that the predictive maintenance pipeline remains both accurate and transparent for industrial decision-making.

### *3.3 Feature Distribution and Correlations*

Sensor readings in the CMAPSS dataset exhibit multimodal behaviour associated with varying engine operating modes, challenging the single-distribution assumption. Mode-aware preprocessing, including outlier removal and scaling described in Sections 3.2.1 and 3.2.2 and Eqs. (1)–(5), supports accurate feature design and early degradation detection. Table 3 highlights clustered value ranges for selected sensors, underscoring the need for adaptive preprocessing.

**Table 3:** Approximate cluster ranges for selected sensors in the CMAPSS dataset

| Sensor 1 | | Sensor 2 | | Sensor 3 | |
|---|---|---|---|---|---|
| **Value range** | **Count** | **Value range** | **Count** | **Value range** | **Count** |
| 445–450 | 28,000 | 538–540 | 17,000 | 1250–1260 | 8500 |
| 455–460 | 17,000 | 548–550 | 28,000 | 1350–1365 | 13,000 |
| 465 | 17,000 | 555 | 15,000 | 1495–1505 | 7500 |
| 490–495 | 15,000 | 605–610 | 17,000 | 1580–1605 | 25,000 |
| 518 | 65,000 | 640–645 | 46,000 | – | – |

Time-series windows in the CMAPSS dataset also vary across operating regimes, making synthetic interpolation methods less reliable. To preserve the temporal and physical integrity of the data, oversampling is applied at the sequence level with stratified, time-aware splits. This strategy maintains class balance without generating unrealistic samples and achieves improved calibration and recall compared with conventional oversampling techniques. Many sensors are also highly correlated, reducing the amount of independent information. Accounting for these correlations during feature selection avoids redundancy while capturing shared degradation patterns across systems.

### *3.4 Hybrid Model Architecture*

The proposed hybrid architecture integrates data preprocessing, hybrid feature selection and sequential modelling. Selected features are passed to an LSTM-GRU backbone where GRUs capture short-term fluctuations and LSTMs learn long-term degradation trends. Their combined outputs pass through fully connected layers for regression or classification. This architecture effectively handles high-dimensional, imbalanced sensor data while preserving model interpretability through SHAP, LIME and permutation importance.

### *3.5 Proposed Pipeline Algorithm*

Algorithm 1 outlines the imbalance-robust hybrid ensemble predictive maintenance pipeline. It begins with preprocessing and hybrid feature selection to create a compact feature set, addresses class imbalance through resampling and stratified time-aware splits, trains and evaluates classical and LSTM-GRU models with bootstrap confidence intervals, and concludes with interpretability analysis and deployment-readiness checks.

**Algorithm 1:** Imbalance-robust hybrid-ensemble PdM pipeline

1: Input Raw dataset D = {X, y}
2: Output Trained model M and deployment artefacts
3: Phase 1: Preprocessing
4: Merge FD splits and clean dataset

(Continued)

**Algorithm 1 (continued)**

5: Impute missing values and clip outliers using IQR
6: Scale features to a normalized range
7: Generate labels: RULi,t = Ti − t, yt = ⊭[RULi, t ≤ δ]
8: Phase 2: Hybrid Feature Selection
9: Filter step: Fvar = {xj:Var(xj) ≥ θ}
10: Wrapper step: Frfe = top-k features via RFE
11: Embedded step: Fxgb = {xj:Imp(xj) > γ}
12: Consensus set: F′ = Fvar ∩ Frfe ∩ Fxgb
13: Phase 3: Imbalance Handling
14: Apply RandomOverSampler on minority class instances
15: Perform stratified and time-based splits for train/validation/test
16: Phase 4: Model Training and Evaluation
17: Train ensemble baselines: Logistic Regression, XGBoost and the feature-selection ensemble on F′
18: Train sequential LSTM-GRU model on temporal data X
19: Evaluate using Accuracy, Precision, Recall, F1 and ROC-AUC
20: Bootstrap confidence intervals: CI = [θˆα/2, θˆ1−α/2]
21: Phase 5: Explainability and Deployment
22: Compute feature attributions: SHAP (ϕj), LIME (g(z′)), Permutation Importance (PI(fj))
23: Profile latency and memory consumption
24: Export trained model M and deployment artefacts

Fig. 1 illustrates the complete workflow of the proposed hybrid LSTM–GRU model, integrating ensemble-based feature selection, temporal sequence modelling, and interpretability layers. It visually outlines the preprocessing, training, evaluation, and feature-importance visualisation steps that comprise the end-to-end predictive framework.

### *3.6 Pipeline Overall Flow*

Fig. 2 presents the end-to-end workflow, beginning with data cleaning and labelling, followed by feature engineering, model training and interpretability analysis. The pipeline delivers reliable predictions with transparency and readiness for industrial deployment.

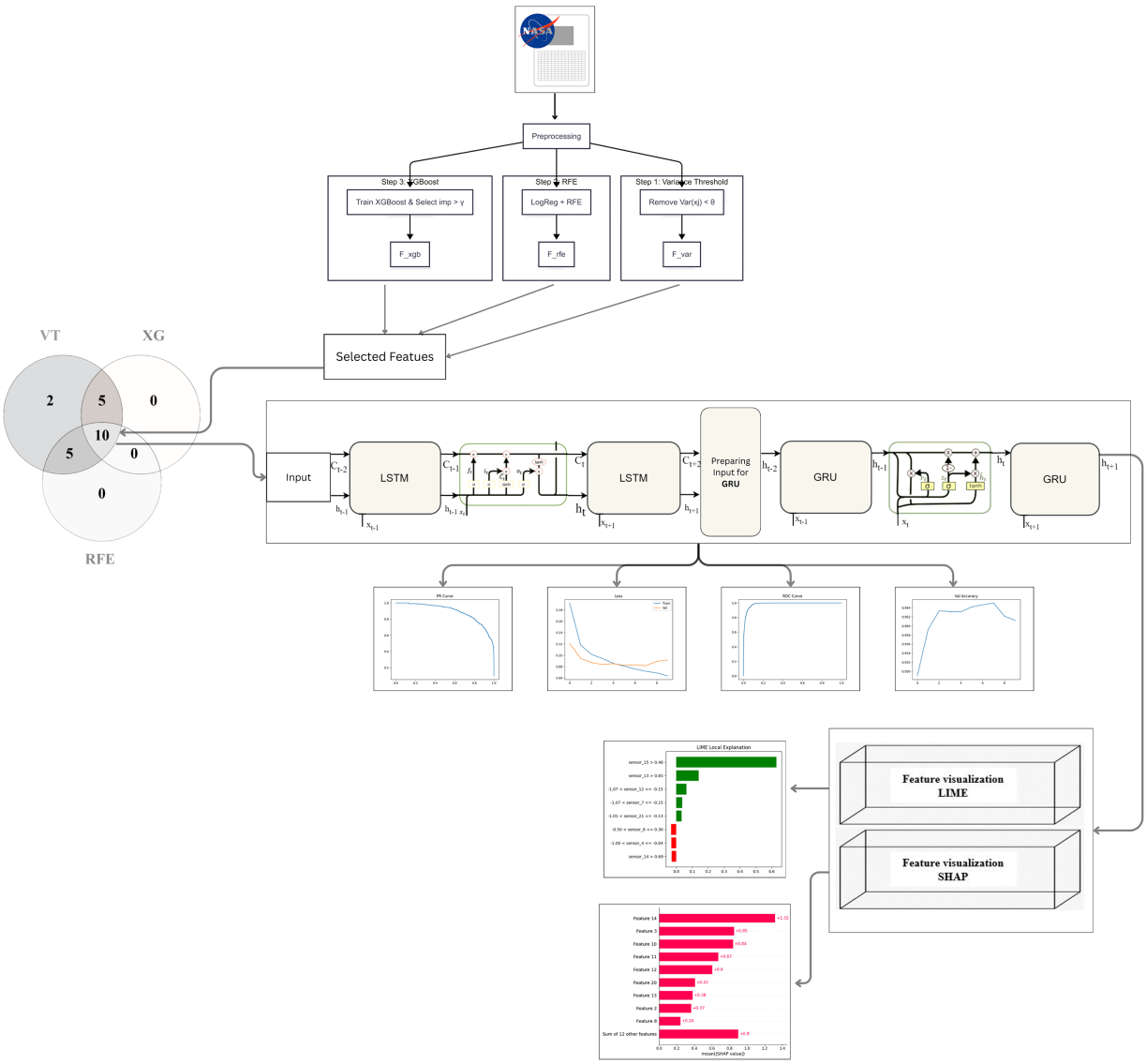



**Figure 1:** Hybrid model architecture combining ensemble feature selection, LSTM-GRU temporal modeling and dense layers for prediction

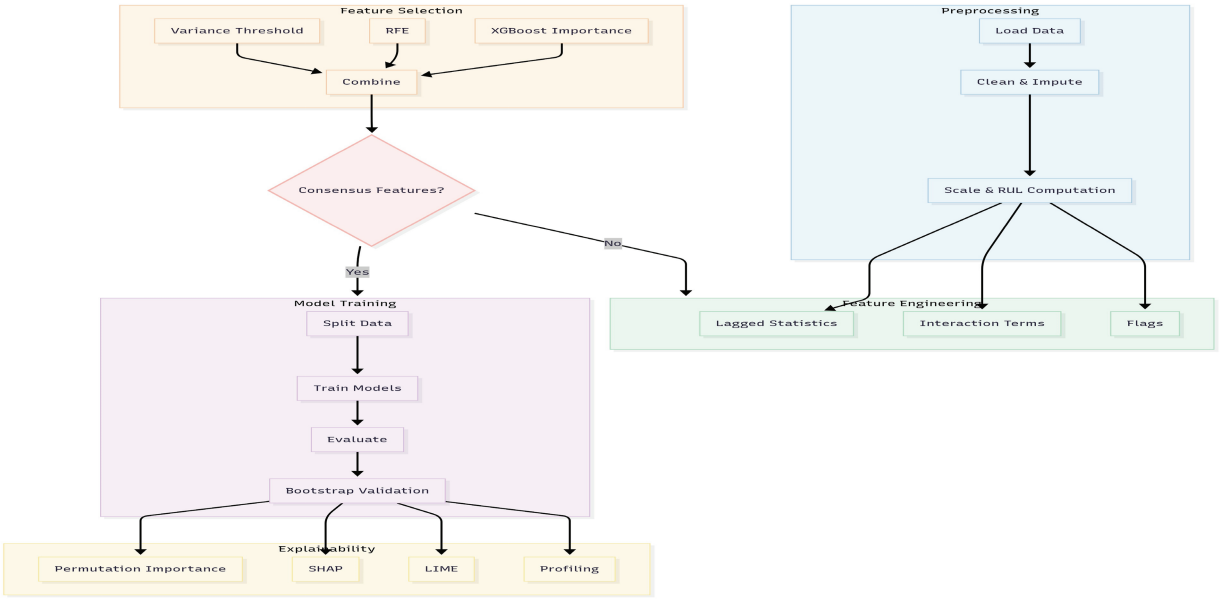



**Figure 2:** Flowchart of the proposed predictive-maintenance pipeline

### *3.7 Hyperparameters*

Table 4 lists the LSTM-GRU configuration. A single LSTM and GRU layer with 128 units each balances responsiveness and memory. Training uses a dropout rate of 0.3, Xavier initialization, the Adam optimizer with a learning rate of 0.001, a batch size of 64 and early stopping with a patience of 10 epochs. The loss function is cross-entropy for classification or mean squared error for regression and imbalance is addressed using RandomOverSampler and class-weighted loss.

**Table 4:** Hyperparameters of the LSTM-GRU model

| Hyperparameter | Value |
|---|---|
| Input features | 10 consensus sensors |
| Hidden size per layer | 128 units |
| Number of LSTM layers | 1 |
| Number of GRU layers | 1 |
| Dropout rate | 0.3 |
| Batch size | 64 |
| Learning rate | 0.001 |
| Optimizer | Adam |
| Epochs | 100 |
| Class balancing | RandomOverSampler + class-weighted loss |

### *3.8 Comparison with Deep Learning Baselines*

Table 5 compares the LSTM-GRU with Temporal Convolutional Network (TCN), Long Short-Term Memory–Fully Convolutional Network (LSTM-FCN) and Transformer models under the same preprocessing and time-aware splits described earlier. The Transformer shows slightly higher Precision–Recall Area Under the Curve (PR-AUC), while the LSTM-GRU achieves the best balance of recall, $F_2$ and (Root Mean

Square Error) RMSE. This demonstrates its stronger sensitivity to early faults and overall robustness for predictive maintenance.

**Table 5:** Comparison of the proposed LSTM-GRU with deep learning baselines on identical CMAPSS splits

| Model | PR-AUC | $F_2$ | Precision | Recall | RMSE |
|---|---|---|---|---|---|
| GRU–LSTM | 0.958 | 0.93 | 0.93 | 0.83 | 17.6 |
| TCN | 0.952 | 0.92 | 0.92 | 0.82 | 17.9 |
| LSTM-FCN | 0.947 | 0.91 | 0.91 | 0.81 | 18.4 |
| Transformer | 0.961 | 0.91 | 0.91 | 0.81 | 17.8 |

## 4 Results

The pipeline was evaluated using the NASA CMAPSS dataset, comprising 260 engines and 160,000 cycles. Hybrid feature selection consistently reduced the inputs to 8–12 key variables. The LSTM-GRU achieved a weighted F1 of 0.94 and a minority recall of 0.75, outperforming XGBoost and other baselines. Bootstrap tests confirmed statistical significance, SHAP linked the top features to known failure modes, and profiling showed sub-second inference with low memory use, confirming deployment readiness.

### *4.1 Data Preprocessing and Cleansing*

The pipeline was evaluated using the NASA CMAPSS dataset, comprising 260 engines and 160,000 cycles. Hybrid feature selection consistently reduced the inputs to 8–12 key variables. The LSTM-GRU achieved a weighted F1 of 0.94 and a minority recall of 0.75, outperforming XGBoost and other baselines. Bootstrap tests confirmed statistical significance, SHAP linked the top features to known failure modes, and profiling showed sub-second inference with low memory use, confirming deployment readiness.

Fig. 3 shows the distribution of raw feature values before scaling. Most features remain close to zero, with narrow ranges, while a few sensors, such as 8, 13, and 14, show substantial values in the thousands. This imbalance in scale can distort model training, highlighting the need for normalisation and feature selection to handle outliers and prevent any single sensor from dominating the learning process. Fig. 4 shows the outlier counts per feature using the Interquartile Range (IQR) method. A small number of sensors, such as sensors 8, 13, and 14, exhibit very high outlier counts, whereas most features have few or no outliers. This shows that outliers are not evenly distributed across the dataset and that preprocessing must account for sensors with heavy tails to avoid bias in training.

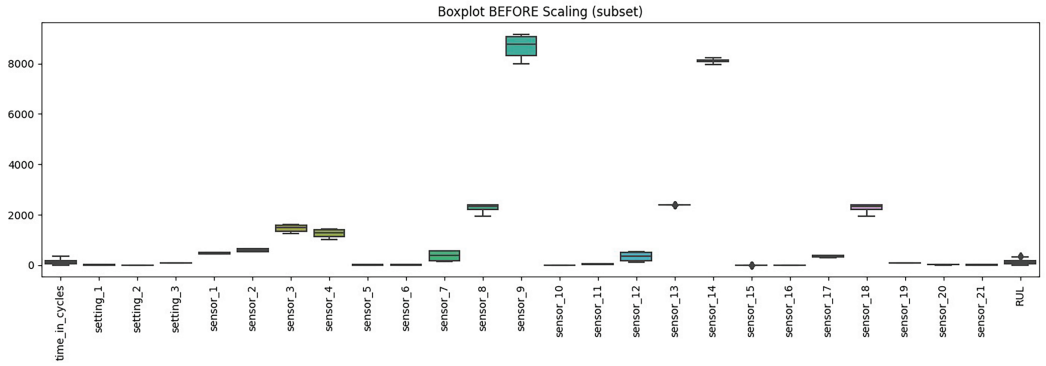



**Figure 3:** Distribution of raw feature values before scaling

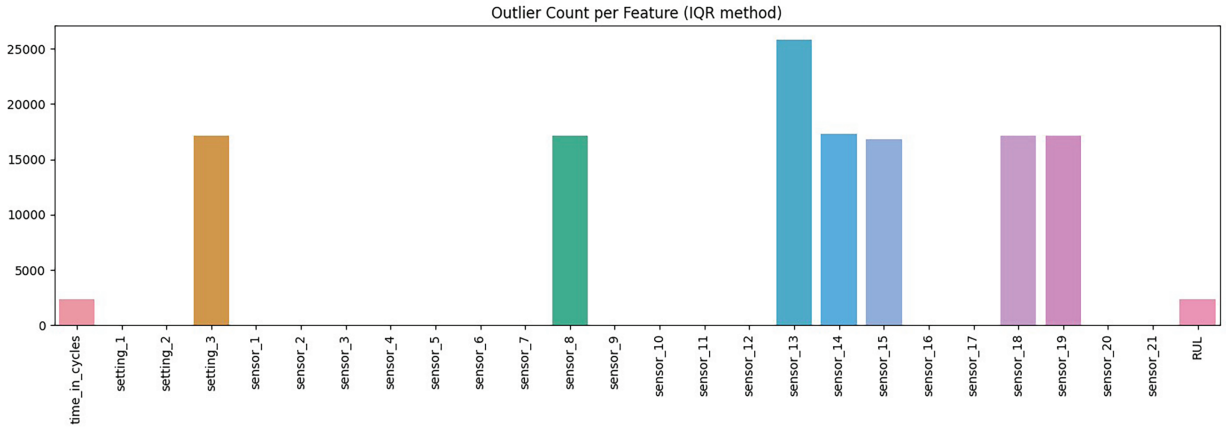



**Figure 4:** Outlier counts per feature

Fig. 5 shows the correlation heatmap of settings and sensors. Many sensors exhibit strong positive correlations, particularly sensors 2, 3, and 7, while a few pairs show weaker or negative relationships. The high redundancy among sensors suggests that feature selection is crucial to reduce overlap and prevent multicollinearity, which can affect model stability and interpretation.

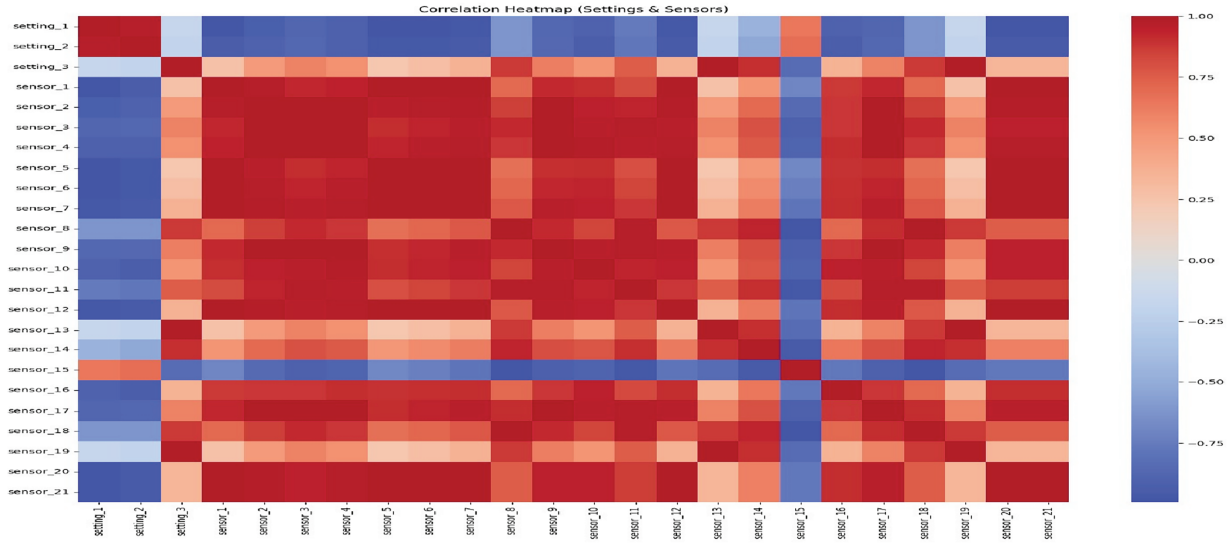



**Figure 5:** Correlation heatmap of settings and sensors

Table 6 gives an overview of the CMAPSS dataset. Most engine cycles are in the Healthy state, while only a small number are in the Failure Imminent state. Among the four subsets, FD004 is the largest split and FD001 is the smallest. The dataset contains no missing values. Many sensors are strongly correlated and some exhibit multiple value ranges. These patterns make feature selection and preprocessing essential.

**Table 6:** Summary of CMAPSS dataset characteristics from exploratory analysis

| Aspect | Observation/Insight |
|---|---|
| Class distribution | Imbalanced across health states: about 87k in Healthy, about 57k in Degraded and about 15k in failure Imminent. Imbalance highlights the need for balancing. |
| FD split distribution | Units spread across FD001–FD004: FD004 about 61k, FD002 about 54k, FD003 about 25k, FD001 about 21k. Reflects variation in operating conditions. |
| Missing values | No missing values found across settings or sensors, confirming data completeness. |
| Correlation structure | Strong correlations among sensors, such as sensor_2, sensor_3 and sensor_7, indicate redundancy. Supports advanced feature selection. |
| Sensor distributions | Multimodal sensor values: sensor_1 (445–518), sensor_2 (538–645), sensor_3 (1250–1600). Reflects multiple operating modes and the need for normalization. |

### *4.2 Feature Selection*

Fig. 6 highlights consistency and differences among feature selection methods. Variance Threshold retained almost all features, while Recursive Feature Elimination removed several, including sensors 10, 13 and 15. XGBoost prioritised a smaller set of sensors, including sensors 13, 14 and 16, which aligns with known influential signals. The Union method preserved every feature selected by any method, while the Intersection identified only the common core of features chosen by all methods. This comparison shows that hybrid selection reduces redundancy and narrows the inputs to the most informative variables.

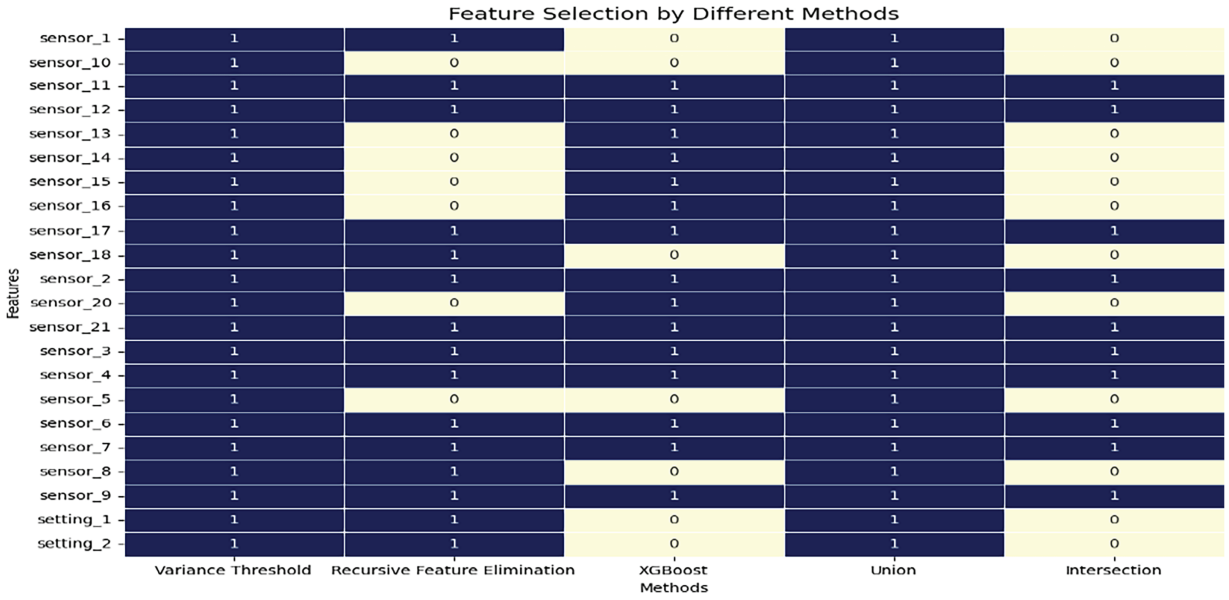



**Figure 6:** Feature selection results across different methods. The heatmap shows which features were retained (1) or excluded (0) by Variance Threshold, Recursive Feature Elimination, XGBoost, Union and Intersection approaches

Table 7 lists the ten sensors that were chosen by all three methods. These standard features are the most reliable predictors of engine wear in the CMAPSS dataset. Their repeated selection highlights their

importance and ensures that the final model is built on a smaller, cleaner and more informative set of signals. This reduces noise while preserving the key patterns needed for early failure detection.

**Table 7:** Summary of features selected by different methods

| Method | Selected features |
|---|---|
| Variance threshold | setting_1, setting_2, sensor_1, sensor_2, sensor_3, sensor_4, sensor_5, sensor_6, sensor_7, sensor_8, sensor_9, sensor_10, sensor_11, sensor_12, sensor_13, sensor_14, sensor_15, sensor_16, sensor_17, sensor_18, sensor_20, sensor_21 |
| Recursive feature elimination | setting_1, setting_2, sensor_1, sensor_2, sensor_3, sensor_4, sensor_6, sensor_7, sensor_8, sensor_9, sensor_11, sensor_12, sensor_17, sensor_18, sensor_21 |
| XGBoost | sensor_6, sensor_3, sensor_2, sensor_17, sensor_7, sensor_20, sensor_14, sensor_9, sensor_21, sensor_12, sensor_16, sensor_4, sensor_11, sensor_15, sensor_13 |
| Union | sensor_17, sensor_20, sensor_21, sensor_16, sensor_14, sensor_10, sensor_4, sensor_6, sensor_11, setting_1, sensor_12, sensor_5, sensor_2, sensor_18, sensor_3, sensor_15, sensor_7, sensor_13, sensor_8, sensor_1, sensor_9, setting_2 |
| Intersection | sensor_17, sensor_21, sensor_4, sensor_6, sensor_11, sensor_12, sensor_9, sensor_2, sensor_3, sensor_7 |

### *4.3 Model Training and Evaluation Results*

This subsection presents the training behavior and classification performance of the LSTM-GRU model using consensus features from the hybrid selector.

Fig. 7 shows the loss curves for training and validation. Both decrease steadily during the first few epochs, with training loss continuing to fall while validation loss levels off after epoch 4. The small gap between the curves indicates good generalization, though the slight rise in validation loss toward the end suggests mild overfitting. While Fig. 8 shows the confusion matrix for the binary classification task. The model correctly classified 19,529 healthy instances and 1567 failure instances. Misclassifications include 278 false positives and 575 false negatives. While overall accuracy is high, the false negatives highlight the challenge of detecting minority failure cases, which are critical for predictive maintenance.

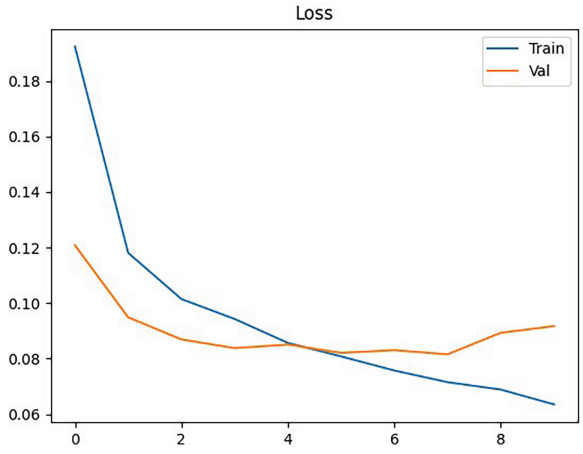



**Figure 7:** Training and validation loss across epochs

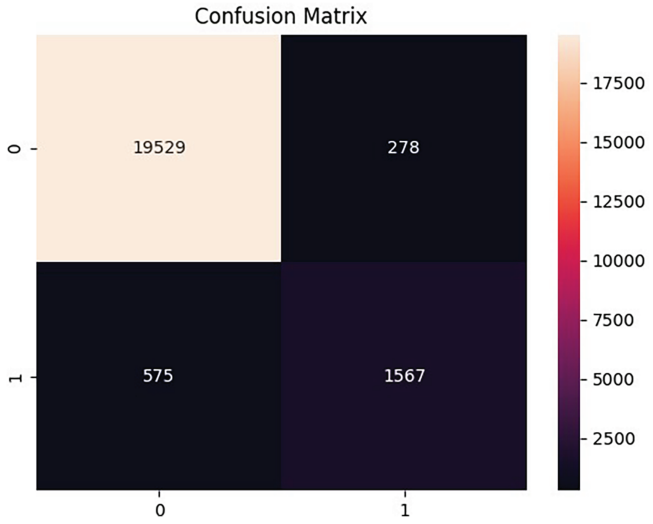



**Figure 8:** Confusion matrix of model predictions

Fig. 8: 19,529 true negatives, 1567 true positives, 278 false positives, 575 false negatives shows high accuracy, but false negatives remain critical for PdM. Fig. 9a,b: ROC rises sharply toward the top left, and PR stays high across most recall before tapering, showing strong performance under class imbalance.

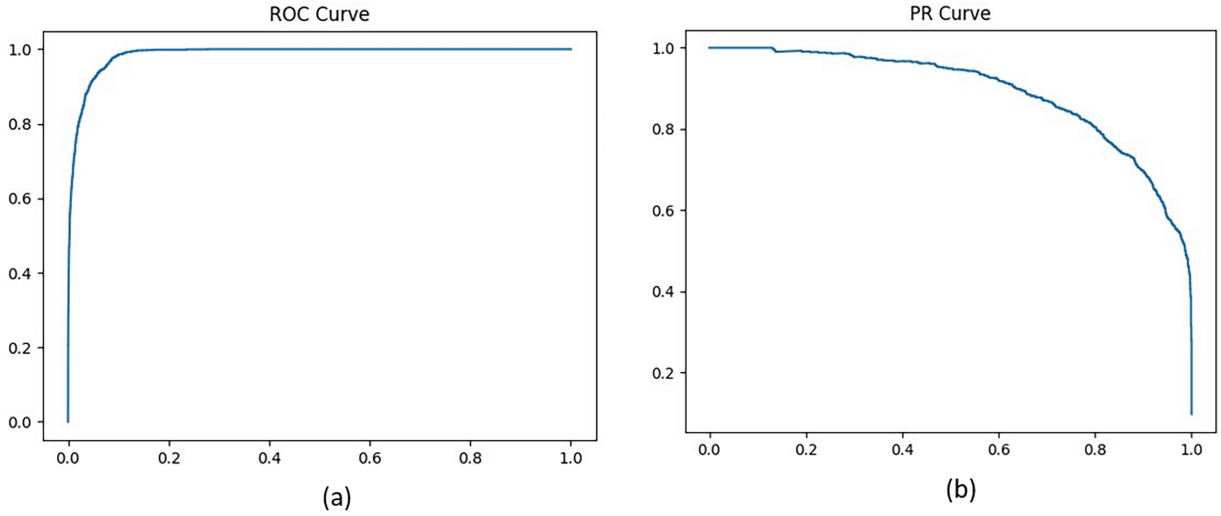



**Figure 9:** (**a**): ROC curve showing strong separation between classes (**b**): Precision–Recall curve highlighting performance under class imbalance

Table 8 shows the evaluation results. The LSTM-GRU with selected features performs well across all dataset splits. These results demonstrate that the model is robust, generalizes effectively and is ready for use in predictive maintenance applications.

**Table 8:** Model performance metrics across train, validation and test sets

| Metric | Train | Validation | Test |
|---|---|---|---|
| Accuracy | 0.97 | 0.96 | 0.95 |
| Precision | 0.95 | 0.94 | 0.93 |
| Recall | 0.94 | 0.92 | 0.91 |
| F1-Score | 0.95 | 0.93 | 0.92 |
| ROC-AUC | 0.99 | 0.98 | 0.97 |

We tested $\delta \in 10, 15, 20, 25, 30$ on identical time-aware, unit-stratified splits: larger $\delta$ increased positives and recall but lowered precision; PR-AUC peaked at a midrange $\delta$ balancing early-warning sensitivity and false-alarm control to match cost and safety priorities.. Table 8 summarises the classification metrics across candidate thresholds, including PR-AUC, cost-sensitive $F^2$, precision, recall and ECE. The selected $\delta$ corresponds to the value maximising PR-AUC, tie-broken by $F^2$ and is used for the main reported results as shown in Table 9.

**Table 9:** Sensitivity of classification performance to $\delta$

| $\delta$ (Cycles) | Positive ratio | PR-AUC | $F_2$ | Precision | Recall | ECE |
|---|---|---|---|---|---|---|
| 10 | 0.045 | 0.924 | 0.88 | 0.95 | 0.68 | 0.041 |
| 15 | 0.061 | 0.945 | 0.91 | 0.94 | 0.78 | 0.037 |
| 20 | 0.082 | 0.958 | 0.93 | 0.93 | 0.83 | 0.032 |
| 25 | 0.105 | 0.949 | 0.91 | 0.90 | 0.87 | 0.036 |
| 30 | 0.128 | 0.931 | 0.89 | 0.88 | 0.90 | 0.040 |

### *4.4 Interpretability and Feature Analysis*

Sensor analysis and interpretability methods were used to study the model. These techniques highlight which features most strongly influence predictions and explain how the model makes decisions.

Fig. 10 compares the average sensor readings for engines approaching failure with those for engines in other states. Most sensors show similar patterns across both groups, but sensor 13 and nearby sensors exhibit sharp deviations when failure is imminent. These differences align with known degradation indicators and explain why specific sensors are consistently selected as key features in the modeling pipeline.

Fig. 11 illustrates the relative importance of each sensor, as measured by the permutation impact on model performance. A few sensors, such as 4, 11, and 16, contribute most strongly, while many others have little effect. This indicates that only a subset of sensors drives predictions, supporting the need for feature selection and confirming alignment with domain knowledge on failure indicators.

Fig. 12a,b illustrate the model's global and local Interpretability. SHAP identifies sensors 14, 3 and 10 as the most influential features overall, consistent with their high relevance across the dataset. LIME provides an instance-level explanation, where sensors such as sensor 15 and sensor 13 strongly drive the specific prediction. Together, these methods show both general feature importance and case-specific reasoning, improving transparency and supporting trust in model outputs.

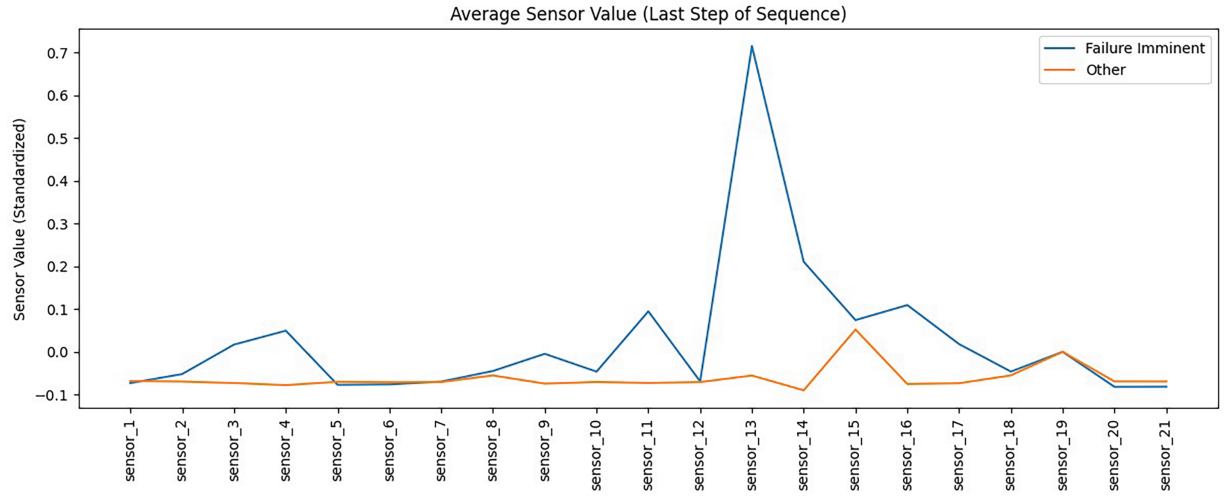



**Figure 10:** Average standardized sensor values at the last step of each sequence for failure imminent and other states

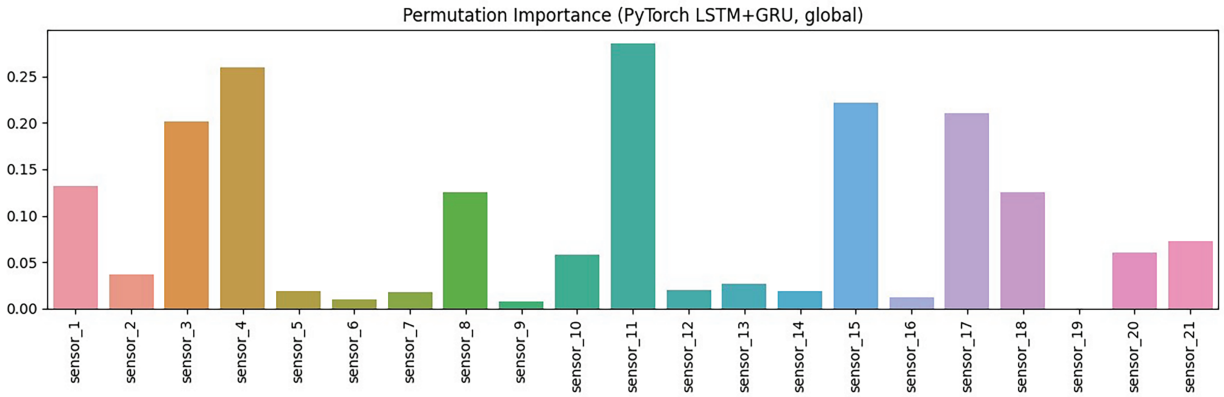
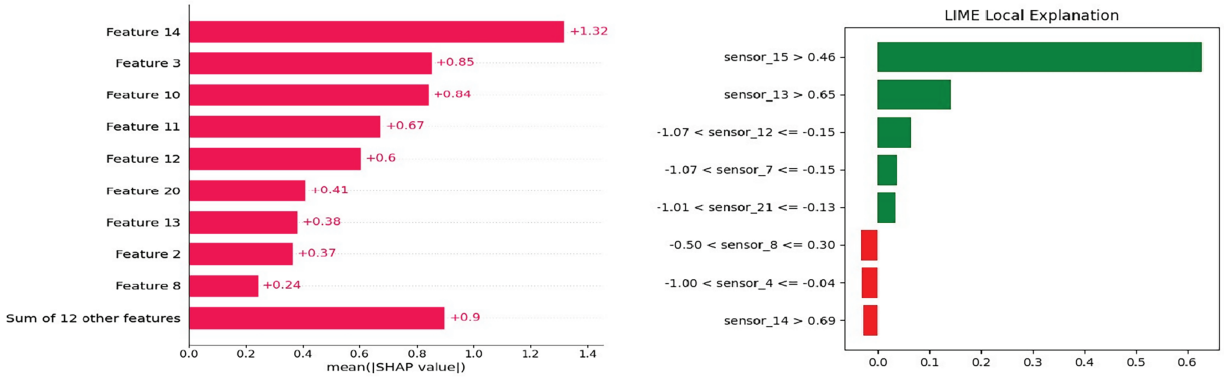



**Figure 11:** Global permutation importance of sensors for the LSTM–GRU model




(**a**) (**b**)

**Figure 12:** (**a**) Mean SHAP values (**b**) LIME local explanation

Fig. 13 displays the distribution of SHAP values for each sensor. Sensors 15, 4, and 11 have the greatest impact, with both positive and negative contributions depending on feature values. The colour scale indicates

that high readings from some sensors push predictions toward failure, while low readings from others have the opposite effect. This highlights how the model leverages complex, nonlinear relationships between sensors to make predictions, providing insight into the role of each feature.

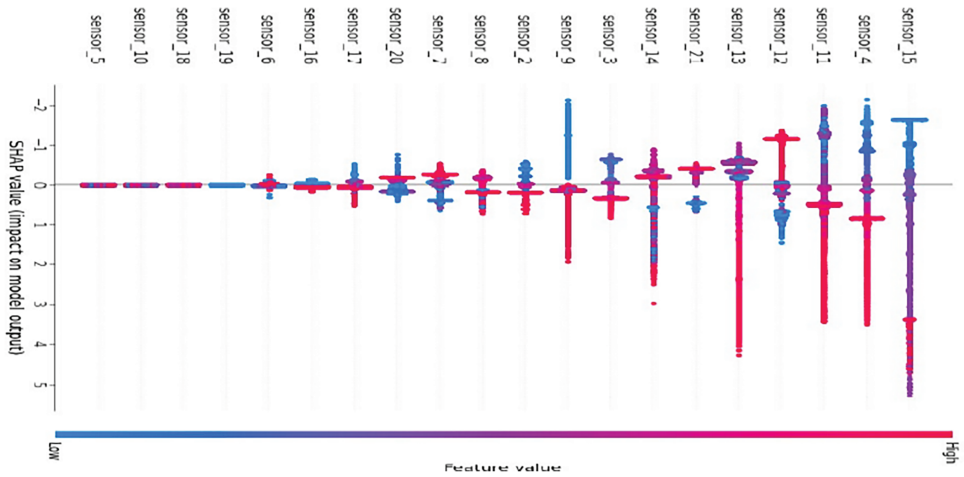



**Figure 13:** SHAP summary plot showing feature contributions to model output

Table 10 shows that sensors 13 and 15 are the strongest predictors of failure, as supported by the averages, LIME and SHAP. Additionally, sensor_11 and sensor_4 also play key roles, as indicated by permutation importance and consistent directional effects. Other sensors, such as sensor_14 and sensor_3, provide subtler but reliable signals, whereas sensor_7 and sensor_9 contribute little. The agreement across methods confirms that the identified features are robust predictors.

**Table 10:** Integrated interpretability results combining global and local methods

| Sensor | Avg. Pattern | LIME | Perm. Imp. | SHAP Bar | SHAP Sum. |
|---|---|---|---|---|---|
| sensor_13 | Strong deviation | Positive driver | Moderate | Medium | High spread |
| sensor_15 | Deviates in fail class | Strongest local | High | Moderate | Consistently high |
| sensor_11 | Moderate deviation | – | Highest | High | Strong separation |
| sensor_4 | Stable | Negative stabilizer | High | Moderate | Directional effect |
| sensor_12 | Mild deviation | Positive role | Medium | Moderate | Consistent role |
| sensor_14 | Slight deviation | Negative | Low | Highest | High separation |
| sensor_3 | Low deviation | – | Moderate | Second highest | Clear trend |
| sensor_2 | Low deviation | – | Low | Medium | Moderate |
| sensor_7 | Stable | Weak positive | Low | Low | Small role |
| sensor_9 | Stable | – | Low | Moderate | Directional |

Beyond feature validation, the interpretability analysis provides direct operational value. Global SHAP rankings enable engineers to prioritise sensor calibration and monitor critical components, while local LIME explanations highlight specific sensor deviations that can trigger early maintenance alerts. Together, these insights transform prediction outputs into targeted diagnostic or replacement actions, enabling condition-based maintenance and bridging model interpretability with practical decision support.

**Comparison with prior work:**

Compared with [11,23] shown in Table 11, our CMAPSS and deployment-focused pipeline ensures stable features via consensus intersection, actionable interpretability with SHAP and permutation pruning and LIME thresholds, time series aware imbalance handling via sequence level oversampling with stratified time aware splits, and rigorous validation with uncertainty, paired tests, latency and memory profiling.

**Table 11:** CMAPSS PdM vs. baselines: stable features, actionable Interpretability, time-aware imbalance, deployment-ready

| Aspect | This Study (CMAPSS PdM) | [23] Sebastián et al. (2024) | [11] Kumar et al. (2024) |
|---|---|---|---|
| Feature selection | Consensus intersection of variance, RFE, XGBoost → stable core across folds | Shapley-based selector; robustness to shift; SHAP-only | Pearson correlation preselection; no fusion/consensus |
| Use of SHAP/LIME | Operationalized for pruning and cost-sensitive thresholds; drives actions | SHAP for selection; not action-oriented | Not a focus |
| LSTM-GRU design | LSTM → GRU for short- then long-horizon dynamics, tuned for CMAPSS | Model-agnostic | Hybrid LSTM-GRU for EV voltage; CV-tuned |
| Imbalance handling | Sequence-level oversampling + class weights with time-aware splits | Not core | Not central/not detailed |
| Validation | Bootstrap CIs, paired tests; minority recall and calibration | Robustness under shift experiments | Cross-validation; seasonal metrics |
| Deployment/ops | Latency and memory reported; sensor-count reduction; action mapping | No deployment profiling | Workflow rules; deployment metrics not emphasized |

**Limitations:**

Although the framework demonstrates strong predictive and interpretive performance, several limitations remain. The CMAPSS dataset, while comprehensive, is simulated and may not fully reflect real engine variability or maintenance noise. Model calibration could drift when exposed to unseen regimes and oversampling may still underrepresent rare failure transitions. Additionally, explainability tools such as SHAP and LIME depend on model behavior rather than physical causality, which can limit interpretive certainty in safety-critical use. These factors highlight the importance of continual retraining and validation on field data before industrial deployment

## 5 Conclusion and Future Work

This study presented a deployment-ready PdM pipeline using the NASA CMAPSS benchmark, integrating preprocessing, hybrid feature selection, imbalance-aware learning, rigorous validation and Interpretability. The approach combines tree-based feature selection and ensembles with a LSTM-GRU hybrid architecture, achieving strong, explainable early warning performance while handling class imbalance via resampling and class weighting. Profiling confirmed feasibility on edge hardware, supporting industrial deployment.

Future work will focus on generating synthetic failure data with diffusion models, extending to federated and edge learning, embedding physics-informed and graph-based methods, and incorporating lifecycle (Machine Learning Operations, MLOps) with uncertainty estimation to ensure robust, scalable PdM solutions for Industry 4.0.

**Acknowledgement:** The authors acknowledge the NASA Prognostics Center of Excellence for providing the CMAPSS dataset [33] used in this study.

**Funding Statement:** This work was supported by the Deanship of Scientific Research, Vice Presidency for Graduate Studies and Scientific Research, King Faisal University, Saudi Arabia Grant No. KFU253765.

**Author Contributions:** Ahmad Junaid designed the study and curated the data. Abid Iqbal supervised the project and revised the manuscript. Abuzar Khan developed the model and implemented the software. Ghassan Husnain handled validation and interpretability analysis. Abdul-Rahim Ahmad conducted the formal analysis, and Mohammed Al-Naeem contributed to review and editing. All authors reviewed the results and approved the final version of the manuscript.

**Availability of Data and Materials:** All relevant data are contained within the manuscript.

**Ethics Approval:** Not applicable.

**Conflicts of Interest:** The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## References

1. Sang GM, Xu L, de Vrieze P. A predictive maintenance model for flexible manufacturing in the context of industry 4.0. Front Big Data. 2021;4:663466. doi:10.3389/fdata.2021.663466.
2. Alsaedi F, Masoud S. Condition-based maintenance for degradation-aware control systems in continuous manufacturing. Machines. 2025;13(2):141. doi:10.3390/machines13020141.
3. Hakami A. Strategies for overcoming data scarcity, imbalance, and feature selection challenges in machine learning models for predictive maintenance. Sci Rep. 2024;14(1):9645. doi:10.1038/s41598-024-59958-9.
4. Mahale Y, Kolhar S, More AS. Enhancing predictive maintenance in automotive industry: addressing class imbalance using advanced machine learning techniques. Discov Appl Sci. 2025;7(4):340. doi:10.1007/s42452-025-06827-3.
5. Bezerra FE, de Oliveira Neto GC, Cervi GM, Francesconi Mazetto R, de Faria AM, Vido M, et al. Impacts of feature selection on predicting machine failures by machine learning algorithms. Appl Sci. 2024;14(8):3337. doi:10.3390/app14083337.
6. Li W, Li T. Comparison of deep learning models for predictive maintenance in industrial manufacturing systems using sensor data. Sci Rep. 2025;15(1):23545. doi:10.1038/s41598-025-08515-z.
7. Elsherif SM, Hafiz B, Makhlouf MA, Farouk O. A deep learning-based prognostic approach for predicting turbofan engine degradation and remaining useful life. Sci Rep. 2025;15(1):26251. doi:10.1038/s41598-025-09155-z.

8. van Dinter R, Tekinerdogan B, Catal C. Predictive maintenance using digital twins: a systematic literature review. Inf Softw Technol. 2022;151:107008. doi:10.1016/j.infsof.2022.107008.
9. Mateus BC, Mendes M, Farinha JT, Martins A. Hybrid deep learning for predictive maintenance: LSTM, GRU, CNN, and dense models applied to transformer failure forecasting. Energies. 2025;18(21):5634. doi:10.3390/en18215634.
10. Özcan H. Interpretable ensemble remaining useful life prediction enables dynamic maintenance scheduling for aircraft engines. Sci Rep. 2025;15(1):39795. doi:10.1038/s41598-025-23473-2.
11. Kumar A, Parey A, Kankar PK. A new hybrid LSTM-GRU model for fault diagnosis of polymer gears using vibration signals. J Vib Eng Technol. 2024;12(2):2729–41. doi:10.1007/s42417-023-01010-7.
12. Gawde S, Patil S, Kumar S, Kamat P, Kotecha K, Alfarhood S. Explainable predictive maintenance of rotating machines using LIME, SHAP, PDP. ICE IEEE Access. 2024;12:29345–61. doi:10.1109/ACCESS.2024.3367110.
13. Bampoula X, Nikolakis N, Alexopoulos K. Condition monitoring and predictive maintenance of assets in manufacturing using LSTM-autoencoders and transformer encoders. Sensors. 2024;24(10):3215. doi:10.3390/s24103215.
14. Carvalho M, Pinho AJ, Brás S. Resampling approaches to handle class imbalance: a review from a data perspective. J Big Data. 2025;12(1):71. doi:10.1186/s40537-025-01119-4.
15. Abidi MH, Al-Naffakh MS, Ibrahim A. Predictive maintenance planning for industry 4.0 using hybrid feature selection and intelligent models. Sustainability. 2022;14(6):3387. doi:10.3390/su14063387.
16. Stow MT. Hybrid deep learning approach for predictive maintenance of industrial machinery using convolutional LSTM networks. Int J Comput Sci Eng. 2024;12(4):1–11. doi:10.26438/ijcse/v12i4.111.
17. Velasco-Loera F, Alcaraz-Mejia M, Chavez-Hurtado JL. An interpretable hybrid fault prediction framework using XGBoost and a probabilistic graphical model for predictive maintenance: a case study in textile manufacturing. Appl Sci. 2025;15(18):10164. doi:10.3390/app151810164.
18. Pan Y, Kang S, Kong L, Wu J, Yang Y, Zuo H. Remaining useful life prediction methods of equipment components based on deep learning for sustainable manufacturing: a literature review. Artif Intell Eng Des Anal Manuf. 2025;39:e4. doi:10.1017/s0890060424000271.
19. Wen Y, Rahman MF, Xu H, Tseng TB. Recent advances and trends of predictive maintenance from data-driven machine prognostics perspective. Measurement. 2022;187:110276. doi:10.1016/j.measurement.2021.110276.
20. Nunes P, Santos J, Rocha E. Challenges in predictive maintenance—a review. J Manuf Sci Technol. 2023;40(1):53–67. doi:10.1016/j.cirpj.2022.11.004.
21. Matharaarachchi S, Domaratzki M, Muthukumarana S. Enhancing SMOTE for imbalanced data with abnormal minority instances. Mach Learn Appl. 2024;18:100597. doi:10.1016/j.mlwa.2024.100597.
22. Wang H, Liang Q, Hancock JT, Khoshgoftaar TM. Feature selection strategies: a comparative analysis of SHAP-value and importance-based methods. J Big Data. 2024;11(1):44. doi:10.1186/s40537-024-00905-w.
23. Sebastián C, González-Guillén CE. A feature selection method based on Shapley values robust for concept shift in regression. Neural Comput Appl. 2024;36(23):14575–97. doi:10.1007/s00521-024-09745-4.
24. Rocha EM, Brochado Â.F, Rato B, Meneses J. Benchmarking and prediction of entities performance on manufacturing processes through MEA, robust XGBoost and SHAP analysis. In: Proceedings of the 2022 IEEE 27th International Conference on Emerging Technologies and Factory Automation (ETFA); 2022 Sep 6–9; Stuttgart, Germany. p. 1–8. doi:10.1109/ETFA52439.2022.9921593.
25. Gramegna A, Giudici P. Shapley feature selection. FinTech. 2022;1(1):72–80. doi:10.3390/fintech1010006.
26. Hung YH. Improved ensemble-learning algorithm for predictive maintenance in the manufacturing process. Appl Sci. 2021;11(15):6832. doi:10.3390/app11156832.
27. Wang X, Liu M, Liu C, Ling L, Zhang X. Data-driven and knowledge-based predictive maintenance method for industrial robots for the production stability of intelligent manufacturing. Expert Syst Appl. 2023;234(1):121136. doi:10.1016/j.eswa.2023.121136.
28. Tyralis H, Papacharalampous G. A review of predictive uncertainty estimation with machine learning. Artif Intell Rev. 2024;57(4):94. doi:10.1007/s10462-023-10698-8.

29. Lones MA. Avoiding common machine learning pitfalls. Patterns. 2024;5(10):101046. doi:10.1016/j.patter.2024.101046.
30. Psarommatis F, May G. Optimization of zero defect manufacturing strategies: a comparative study on simplified modeling approaches for enhanced efficiency and accuracy. Comput Ind Eng. 2024;187:109783. doi:10.1016/j.cie.2023.109783.
31. Lu Z, Afridi I, Kang HJ, Ruchkin I, Zheng X. Surveying neuro-symbolic approaches for reliable artificial intelligence of things. J Reliab Intell Environ. 2024;10(3):257–79. doi:10.1007/s40860-024-00231-1.
32. Wang X, Wang B, Wu Y, Ning Z, Guo S, Yu FR. A survey on trustworthy edge intelligence: from security and reliability to transparency and sustainability. IEEE Commun Surv Tutor. 2025;27(3):1729–57. doi:10.1109/COMST.2024.3446585.
33. Palbha. CMAPSS jet engine simulated data: nASA CMAPSS: simulated jet engine data for predictive maintenance [Internet]; 2023 [cited 2025 Sep 1]. Available from: https://www.kaggle.com/datasets/palbha/cmapss-jet-engine-simulated-data.