Jurnal Teknik Industri, Vol. 28, No. 2, December 2026 DOI: 10.9744/jti.28.2.198-214..
ISSN 1411-2485 print / ISSN 2087-7439 online

Fusing Operational Downtime Logs with Maintenance Records for Predictive Maintenance: An Evaluation of the Precursor Signal Hypothesis in High-Volume Manufacturing

Yemima Christabella Chen[1], Andreas Handojo[1*], Tanti Octavia[2]
[1)] Faculty of Industrial Technology, Informatics Engineering Department, Petra Christian University,
[2)] Faculty of Industrial Technology, Industrial Engineering Department, Petra Christian University,
Jl. Siwalankerto 121-131 Surabaya, Indonesia
Email: handojo@petra.ac.id*
*Corresponding author

Abstract: Manufacturing operational data remain underutilized for predictive maintenance because record-keeping is split between the Computerized Maintenance Management System (CMMS) and the Operational Downtime Log (ODL). This study integrates these two sources and tests whether accumulated operational disturbances can serve as early indicators of machine failure. It covers 12 filling machines over one calendar year, producing 2,734 machine-day observations with 32 features derived from 14,316 ODL entries and 232 CMMS tickets. Time-to-failure was predicted using Random Forest regression under temporal validation. The integration succeeded, with 98.1% of breakdowns preceded by recorded operational activity within 30 days. The predictive contribution of the ODL was not established: mean absolute error fell by 1.57 days on a single temporal split, but the ODL reduced error in only two of four rolling-origin folds, for an average reduction of 0.22 days. The validation strategy alone shifted R^2 by 0.934, from −0.195 under temporal splitting to +0.739 under random splitting. An operational backtest nevertheless showed that the model performed comparably to a calendar-based rule on detection coverage while requiring roughly one-third fewer alert machine-days. These findings show that the two record systems can be integrated reliably, that the predictive value of operational logs remains unproven at this data volume, and that temporal validation is essential for credible performance estimates.

Keywords: Predictive maintenance; random forest; data fusion; micro-stoppages; temporal validation; data leakage

Introduction

Industry 4.0 has moved maintenance in production towards a more anticipatory and strategic approach, intended to secure operational continuity [1]. Manufacturers do not adopt this approach at the same pace, and readiness varies considerably among Indonesian firms.

The extent of that variation has been measured directly. Using the companies' 2018 annual reports, Tanjung et al. [2] classified manufacturing companies listed on the Indonesia Stock Exchange (IDX), condensing 38 financial and operational factors through principal component analysis before grouping the companies by cluster analysis. The analysis identified three distinct levels of readiness rather than a single verdict, and a decision tree showed that the total book value of machinery was the variable that best separated them, with labour cost, revenue per employee, and the cost-to-revenue ratio acting only as secondary discriminators.

Readiness among these firms is thus closely associated with capital intensity. That association bears directly on the present study, because a readiness gap defined by capital intensity cannot be closed by recommending further capital expenditure. Firms in the lower readiness tiers need approaches that draw on the records they already keep rather than on new instrumentation. This is the situation examined here.

Sensor-based architectures assumed by smart manufacturing illustrate this gap. Such architectures rely on cyber-physical systems equipped with Internet of Things (IoT) sensors, whose readings are analyzed in real time and exchanged through the cloud [3], [4]. Plants without this infrastructure cannot process data in the same way. The plant studied here stores most daily operational data in isolated local spreadsheets, and the logs kept by operators are never linked to the maintenance database that holds the technical history of each machine.

Submitted: July 28, 2026; Revised: Sept 28, 2026; Accepted: Oct 12, 2026

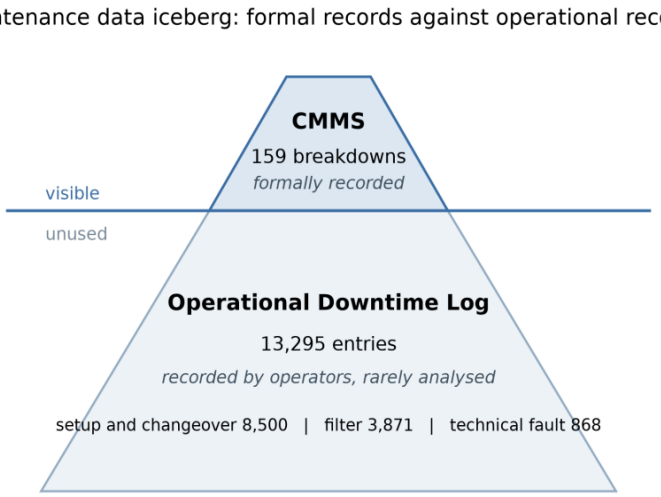


Figure 1. Maintenance data iceberg: formal CMMS records compared with the informal operational downtime log.

The result is dark data (records that are collected but never analyzed), which undermine reliability metrics and obscure the true condition of machines on the shop floor.

The effects of this fragmentation are financial as well as informational. An unexpected line stop costs the plant saleable output, delays downstream commitments, and wastes the energy already spent on partially processed material. Lost output of this kind is among the largest controllable risks for organizations running several parallel lines [5], and early warning systems are intended to reduce it. Such systems are difficult to build from existing records, however, because formal maintenance databases capture only a fraction of operational disruptions [6], [7], a disparity referred to here as the maintenance data iceberg. Serious failures are documented and entered into the Computerized Maintenance Management System (CMMS), whereas brief disruptions are typically cleared by operators at the line and recorded, if at all, only in an operational downtime log (ODL). These brief disruptions are known as minor stops or micro-stoppages, and their cumulative total remains invisible to the CMMS. Indicators such as overall equipment effectiveness and mean time between failures (MTBF) are therefore computed from incomplete records. The broader result can be described as a form of maintenance myopia, in which organizations track what breaks conspicuously and overlook what degrades gradually.

Figure 1 depicts this disparity in the plant examined here. The CMMS holds 232 tickets for the year, of which 159 describe genuine breakdowns, whereas the ODL records 14,316 entries over the same period, roughly 90 operational entries for every recorded breakdown. Reliability engineering treats such minor events not as noise but as weak signals of progressive mechanical decline, and the data-driven maintenance literature has examined their diagnostic value [6]. A rising rate of minor disruptions has been reported to precede major component failure on complex lines, which suggests that models disregarding these signals may be limited as early warning systems [8]. Whether that reasoning holds for a given plant is an empirical question rather than a presumption, and this study sets out to test it.

Two considerations make the test worthwhile. Operational logs are already collected, so exploiting them requires no capital expenditure on instrumentation. The records are also manual and heterogeneous, which leaves it uncertain whether they contain a usable signal at all. A negative outcome is informative.

This research presents four main contributions. The first is a replicable procedure for integrating a CMMS with an operational downtime log at daily granularity, along with quantitative measures of how well the two sources link. The second is a classification of downtime codes based on the physical nature of each event, validated by plant maintenance personnel, which separates scheduled changeover from mechanical deterioration. Only technical-fault entries show a positive association with subsequent failures, and even that association is weak, so treating all operational entries as degradation signals would dilute it. The third is an assessment of the Precursor Signal hypothesis under temporal validation, which shows that the predictive benefit of operational features does not persist across evaluation periods. The fourth is a measurement of the effect of the validation

Table 1. Positioning relative to representative prior work

Study	Data source	Validation reported	Contribution
Vallim Filho et al. [9]	Sensor and load-cycle data	Not specified	Load cycle as a health indicator superior to chronological age
Achouch et al. [3]	Review of Industry 4.0 practice	Not applicable	Multi-source fusion identified as a key enabler
Theissler et al. [8]	Automotive sensor data	Reported	Labelling difficulty in early degradation and high field noise
Tiddens et al. [6]	Organizational case studies	Not applicable	Data silos and narrow focus on major failures
Azeta et al. [10]	Review of AI in maintenance	Not applicable	Ensemble methods effective on noisy industrial data
This study	Manual logs and CMMS records without added sensors	Time-based split, rolling-origin cross-validation, explicit comparison against random splitting	Reproducible fusion with integration metrics, a downtime taxonomy, an evaluation of the Precursor Signal hypothesis, and quantification of validation-design effects

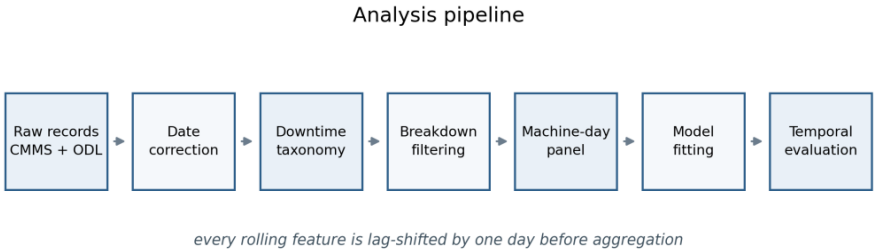


Figure 2. Analysis pipeline: seven stages from raw records to evaluation, with the lag-shift applied before every rolling aggregation.

design itself, showing that a random train–test split inflates the coefficient of determination by 0.934 relative to a time-based split on identical data, features, and model. Table 1 positions these contributions against representative prior work.

Methods

The analysis proceeds in seven stages, summarized in Figure 2. Data acquisition and timestamp correction come first, followed by the downtime taxonomy and the definition of the prediction target. A machine-day panel with lagged features is then built, with model fitting and evaluation completing the sequence. The analysis code is available from the corresponding author upon reasonable request, although a confidentiality agreement with the plant prevents sharing the underlying operational records.

Study Site and Data Source

The data were obtained from a coating-materials plant that operates continuous process-based production lines. The study covers 12 automatic filling machines, which dispense liquid product into containers of 1–25 kilograms. Mixing and labeling equipment lies outside the scope of the study. Table 2 lists the principal components of these machines; these components are later used in component-level reliability reporting.

Records were extracted from the CMMS and the ODL for the period from 1 January to 31 December 2024. The CMMS holds formal maintenance tickets, each carrying a machine identifier, a category, request and close timestamps, and a free-text description. Line operators complete the ODL once per shift, recording each production interruption with a standardized downtime code, start and end times, and a duration in minutes. Both sources are therefore event-level records with minute-resolution timestamps. Annual production totals per machine were also available, but their annual aggregation prevented them from supporting time-varying features, so they were excluded.

Table 2. Principal components of the filling machines studied

Component	Function	Component	Function
Filling nozzle	Dispenses product into the container	Auto sealer	Closes and seals containers after filling
Ball valve (ballmatic)	Controls flow during dispensing	Weighing unit, load cell	Verifies final weight for volume control
Chain drive	Transmits motor rotation to conveyor	Star wheel	Positions containers beneath the nozzle
Conveyor	Moves containers between stations	Limit switch, control panel	Senses position and provides electrical control
Pump	Delivers product at controlled pressure	Filter	Removes contaminants before filling

Machine identifiers differ between the two systems: the CMMS uses the prefix M, whereas the ODL uses the prefix F (for filling). The mapping was established from the machine master list, in which the numeral at the end of each machine name corresponds to its CMMS code. ODL entries without a machine code (587) and entries whose codes have no counterpart in the master list (434) were excluded, removing 1,021 records (7.1% of the ODL).

Correction of Recorded Timestamps

Both logs contain a systematic defect in the date fields that must be resolved before any temporal analysis. The spreadsheet program misparsed a portion of data as mm-dd-yyyy and switched the day and month whenever the day value was 12 or less, even though the source systems record dates in the dd-mm-yyyy format. The defect affects 93 of 232 CMMS records and 5,357 of 14,316 ODL records. A further eight CMMS records carry dates that cannot be parsed, so they were excluded.

The diagnosis rests on a simple arithmetic regularity. Entries stored as text always carry a leading day value between 13 and 31, whereas entries converted to a date type always carry a day field between 1 and 12. A correctly recorded set of dates would distribute days from 1 to 31 across both groups. The observed separation is instead complete and without overlap, which is consistent with conditional reinterpretation by the spreadsheet. The correction restores the intended reading by transposing the day and month for date-typed entries, while the remaining text entries are parsed with the day placed first.

The effect on aggregate statistics is material. The uncorrected timeline assigns 1,149 operational events to April and 1,401 to October, whereas after correction these figures become 804 and 1,629, a decrease of 345 events for April and an increase of 228 for October. Figure 3 compares the monthly distribution of operational entries before and after the correction. Both timelines contain the same entries, so the defect distorts the apparent seasonality of the log rather than its total volume. Any rolling window computed on the uncorrected data is therefore unreliable, as are the temporal alignment between the two logs and any monthly ranking. All metrics and aggregations reported in this study are computed from the corrected timeline.

Taxonomy of Downtime Codes

The ODL shows that many recorded interruptions do not represent component faults. The 12 machines account for 13,295 entries in total, and five codes alone contribute 12,128 of them (91%). Among the scheduled and operator-driven codes, production setup contributes the most entries (5,560), followed by product changeovers (1,638) and auto-sealer overflows caused by delayed stacking (1,059). These are scheduled or operator-driven activities, so treating them as degradation signals would measure batch-change frequency rather than equipment health. Filter replacements (2,285 entries) and filter pressings (1,586) differ in kind, as discuss in the following sub section.

The codes were categorized into the four groups presented in Table 3 according to the physical mechanism underlying each event, with the component structure in Table 2 used as a reference. Two classifications warrant brief comment. Filter replacement and pressing were assigned to the degradation category because filter servicing is triggered by rising flow resistance; their frequency tracks component condition rather than a fixed maintenance schedule. Auto-sealer overflow was assigned to the operational-load category instead, since its recorded cause is operator-delayed stacking. Plant maintenance personnel validated the full classification and confirmed that filter servicing becomes more frequent as filters begin to deteriorate.

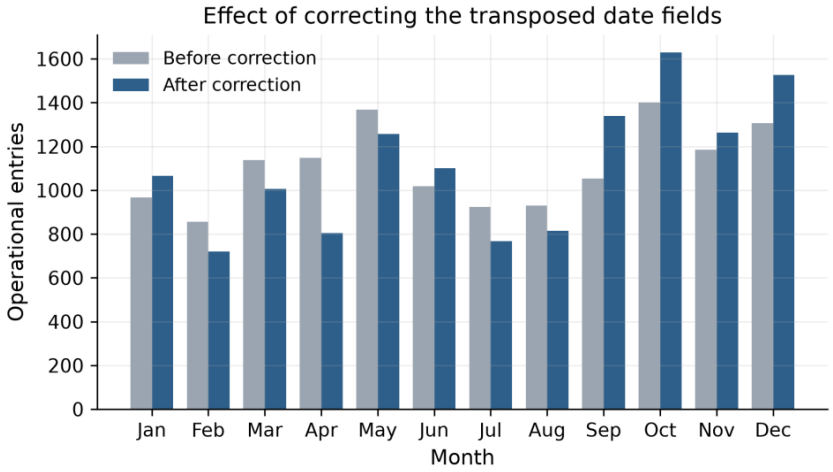


Figure 3. Monthly distribution of operational entries before and after correction of the transposed date fields.

Table 3. Grouping of operational downtime codes

Group	Representative codes	Entries	Interpretation
Operational load	Production setup, product changeover, auto-sealer overflow, final packing	8,500	Proxy for daily production intensity
Filter events	Filter replacement, filter pressing	3,871	Slow degradation signal
Technical fault	Filling malfunction, valve wear, chain breakage, pump failure, conveyor motor fault	868	Acute degradation signal
External utility	Power interruption, compressor pressure loss	56	Cause external to machine condition

The observed correlations support separating technical faults from the other groups. Using Pearson's coefficient, plant-wide monthly counts of each group were correlated with the number of breakdowns in the following month. The technical-fault group yields a coefficient of +0.21, whereas the filter, operational-load, and external-utility groups yield coefficients near zero or negative (−0.01, −0.08, and −0.11). Only technical faults show a positive, albeit weak, association with subsequent breakdowns, so treating all entries as a single signal would likely dilute that association. The near-zero coefficient for filter events does not contradict their assignment to the degradation category, since monthly plant-wide totals are too coarse to capture slow, machine-specific deterioration. The assignment rests on the physical mechanism confirmed by plant personnel rather than on these correlations. The coefficients derive from only 11 monthly pairs, and none reaches statistical significance, so they are reported to inform feature design rather than as conclusive evidence.

Definition of Breakdowns and Prediction Target

Not all CMMS tickets represent actual equipment failures. Valid dates and machine identifiers appear on 224 of the 232 recorded tickets. The Electrical and Mechanical categories account for 178 of these, while the remaining 46 comprise project, utility, or building work. A further 19 tickets within the technical categories describe procurement, installation, relocation, or adjustment requests rather than repairs. These were identified by Indonesian action terms at the start of the ticket titles, such as *pembelian* (purchase), *penambahan* (addition), and *perapian* (tidying). Each was then checked manually to confirm the absence of fault symptoms. Their exclusion leaves 159 genuine breakdown events.

These 159 tickets do not correspond one-to-one with maintenance events. Tickets associated with a single machine frequently cluster within short intervals and describe different components serviced during one technician visit. For one machine, for instance, maintenance on a ball valve and a load cell was recorded on the same date. Tickets raised on the same machine within seven days of the first ticket in a group were therefore merged into a single maintenance episode, which reduced the 159 individual tickets to 110 distinct episodes.

Table 4. Feature inventory

Feature family	Source	Windows	Count
Technical fault event count	ODL, fault group	1, 3, 7, 14, 30 days	5
Technical fault duration (minutes)	ODL, fault group	1, 3, 7, 14, 30 days	5
Filter event count	ODL, filter group	1, 3, 7, 14, 30 days	5
Filter event duration (minutes)	ODL, filter group	1, 3, 7, 14, 30 days	5
Operational load event count	ODL, load group	1, 3, 7, 14, 30 days	5
Operational load duration (minutes)	ODL, load group	1, 3, 7, 14, 30 days	5
Days since last maintenance episode	CMMS	—	1
Cumulative episode count	CMMS	—	1
Total			32

This aggregation reflects an interpretive choice rather than neutral data cleaning, and the number of episodes depends on the threshold, with thresholds of 0, 1, 3, 7, and 14 days yielding 148, 140, 132, 110, and 88 episodes, respectively.

The prediction target is time-to-failure (TTF), defined as the number of calendar days from a given observation to the start of the next maintenance episode on the same machine. The task is a continuous regression rather than a classification problem. One clarification follows from this single-target design. The urgency categories introduced in the Maintenance Prioritization subsection do not derive from a secondary classification model but are constructed by applying decision thresholds to the predicted TTF.

Panel Construction and Feature Engineering

The unit of analysis is the machine-day observation. A panel of 4,392 machine-days was constructed from the 12 machines over 366 days, and operational events were aggregated to this panel by downtime group. Three groups entered the feature set, namely technical faults, filter events, and operational load, while external-utility events were excluded because their cause lies outside machine condition. Each of the three groups contributes an event count and a total duration in minutes, summed over rolling windows of 1, 3, 7, 14, and 30 days. Two further variables capture maintenance history, namely the number of calendar days since the most recent episode and the cumulative episode count. The feature space therefore comprises 32 input variables (3 groups × 2 measures × 5 windows, plus 2 history variables), itemized in Table 4.

Every rolling feature is lag-shifted by one day before aggregation, so an observation on day H draws only on events up to day $H-1$. This shift prevents same-day information from entering the predictors, including events recorded on the day of a breakdown. The panel formulation also removes two ambiguities that arise when downtime events are matched directly to subsequent failures, namely how far back an event may be linked to a failure and how to treat stoppages that are never followed by one. Every stoppage in the panel contributes to the features of the machine-day on which it occurs, whether a failure follows, and no linkage horizon needs to be specified.

Three types of observation were excluded from model fitting, namely machine-days preceding the first recorded episode on each machine, machine-days with no subsequent episode within the study period, and machine-days with a TTF exceeding 60 days. The 60-day limit matches the two-month procurement planning period at the site. These exclusions reduce the 4,392 machine-days to 2,734 observations. The full test-period panel was retained separately for the operational back test, where conditioning on target validity would bias the assessment.

Two variables from the original feature design could not be built. Dynamic workload indicators, namely cycle-based workload intensity and the Cumulative Stress Index, require continuous daily records of production volume. Production data at the study site were available only as annual totals per machine, and these totals remained constant within each machine, so they would act as an identity marker rather than a dynamic measure of load. The daily count and duration of operational-load events, dominated by production setups and product changeovers, therefore serve as the proxy for operational load, capturing batch-transition frequency rather than throughput.

Missingness elsewhere in the records required no comparable compromise. No imputation was applied to the machine identifier field. Entries without a valid identifier were excluded rather than imputed, since schemes such as k-nearest-neighbour imputation [11], [12] would introduce arbitrary machine attribution. The 5,472

entries (38.2%) with a duration of zero indicate interruptions shorter than one minute rather than missing data since duration in the ODL is reported in full minutes. These entries contribute to the event-count features but not to the duration features. Two entries whose duration could not be parsed were assigned a duration of zero, a substitution affecting about 0.01% of the log.

Resampling

Synthetic Minority Over-sampling Technique for Regression (SMOTE-R) [13], [14] was applied to the lower quartile of the TTF distribution, which contains the observations recorded shortly before a breakdown. Each synthetic instance was generated by interpolating, at a random position, between a minority observation and one of its five nearest neighbours, across both the feature vector and the target. An oversampling factor of 2.0 doubles the number of lower-quartile observations, and factors of 1.5, 2.0, and 3.0 enlarge the training partition from 2,127 to 2,419, 2,711, and 3,295 observations, respectively. The target is a continuous variable rather than a rare class, so the benefit of resampling is uncertain and the application is exploratory, testing whether emphasizing observations close to failure improves prediction.

Resampling was executed strictly within the training partition, after the train–test split rather than before it. The sequence matters, because generating synthetic points before partitioning risks interpolating between a training observation and a test observation, which would introduce test information into the training set. The Effect of the Validation Design subsection reports the results for all three factors and quantifies the impact of reversing that sequence.

Models and Hyperparameters

Random Forest regression [15] serves as the primary predictive architecture, chosen for its robustness to noisy, high-dimensional tabular data and for the interpretability of its feature attributions [10]. The ensemble averages B independently constructed decision trees, each trained on a bootstrap resample. Averaging across trees reduces model variance relative to a single tree.

$$\hat{Y} = \frac{1}{B} \sum_{b=1}^{B} \quad T_b(x) \text{ (1)}$$

Here $T_b(x)$ denotes the output of the b-th decision tree for the feature vector x, and B is the ensemble size. The baseline configuration uses $B = 400$ trees, a minimum leaf size of 10 observations, unrestricted tree depth, a fixed random seed, and all p features as split candidates (the scikit-learn default). The ensemble therefore corresponds to bagged regression trees, since all features are considered at each split.

This default configuration was compared with the best configuration from a systematic grid search over 32 hyperparameter combinations. The search space covered tree counts of 200 and 400, minimum leaf sizes of 2, 5, 10, and 20, a maximum depth of 10 or no depth limit, and $\sqrt{p}$ or $p/2$ features per split. Parameter selection used a three-way temporal partition. Candidates were trained on data from January to July (1,520 observations), and the combination with the lowest mean absolute error on August and September (607 observations) was selected, while the October–December test set (607 observations) was not used during tuning. The selected configuration (400 trees, a minimum leaf size of 5, a maximum depth of 10, and $p/2$ features per split) did not improve on the default, yielding an R^2 of −0.219 against −0.195 on the test set, so the default configuration was retained. Selecting on the validation set rather than the test set avoids reporting an optimistic test-set maximum. The selected configuration is also more constrained than the default, which is consistent with the results in Table 6, where the simplest model performs best.

Four benchmarks were evaluated on identical features and temporal partitions. A naive mean predictor, which returns the training-set mean, serves as the reference for R^2. Three further benchmarks complete the suite: Linear Regression, a Decision Tree with a maximum depth of 6, and Gradient Boosting with default settings. A CMMS-only variant restricted to the two maintenance-history features isolates the incremental value of the operational log, and it was compared with the full feature set under an identical Random Forest configuration of 300 trees and a minimum leaf size of 10. The result of this comparison was averaged over three random seeds on the single temporal split, whereas the rolling-origin comparison used one seed. Other analyses use different Random Forest configurations, each with a fixed random seed. Tables 6 and 7 use 500 trees with a minimum leaf size of 5, while Table 8, the rolling-origin mean, and the operational backtest use the default of 400 trees with a minimum leaf size of 10. Table 9 uses 300 trees with a minimum leaf size of 10, and Table 10 uses 500 trees with a minimum leaf size of 10.

Validation Design and Metrics

Random data partitioning is unsuitable for this panel structure. Adjacent machine-day observations are strongly autocorrelated, because consecutive rows belong to the same failure episode and carry near-identical feature values. A 30-day rolling window, for instance, shifts by only one day between adjacent records. Random allocation therefore places near-duplicate observations in both partitions, so the model can interpolate between observations it has effectively already seen instead of predicting out of sample. For this reason, the primary design uses a temporal split, training on data from January to September (n = 2,127) and testing on October to December (n = 607).

The results are further checked using a four-fold rolling-origin cross-validation, which reduces the risk that performance estimates reflect a single arbitrary cut point. Each fold trains on all data up to the end of one month and tests on the following month, with August, September, October, and November as the four test months (n = 335, 272, 292, and 240). hyperparameters were selected only on the separate three-way temporal partition described above, so no hyperparameter was chosen on the test data. The problems of applying random splits to time-dependent data are well established in the literature [16]–[18].

Predictive performance is evaluated with three metrics, namely the coefficient of determination (R^2), mean absolute error (MAE, in days), and root mean square error (RMSE) [19], [20]. A negative R^2 indicates that the predictions perform worse than a constant equal to the mean of the evaluated set, which is why even the naive training-mean predictor yields a negative value. A negative R^2 does not, however, imply the absence of an actionable operational signal, as examined in the Operational backtest subsection. Mean absolute percentage error is not used, because the target falls to as little as one day immediately before a failure, which makes percentage-based metrics numerically unstable.

$$R^2 = 1 - \frac{\sum(y_i - \hat{y}_i)^2}{\sum(y_i - \bar{y})^2} \quad (2)$$

$$MAE = \frac{1}{n}\sum|y_i - \hat{y}_i| \quad (3)$$

Equations (2) and (3) use the following notation: y_i denotes the observed TTF in days for the i-th machine-day observation, $\hat{y}_i$ the corresponding model prediction, $\bar{y}$ the mean of the observed values in the evaluated set, and n the number of observations in that set. The numerator of (2) is the residual sum of squares, and its denominator is the total sum of squares about the mean.

Feature attributions are evaluated with two complementary methods. Mean decrease in impurity (MDI) is computed from the training data and measures the total variance reduction achieved by splits on each feature across the ensemble, with variance reduction serving as the impurity criterion because the task is regression. MDI is known to favor high-cardinality continuous features [21], and because it is computed on the training data, it may also reflect patterns that do not generalize. Permutation importance is computed out of sample on the test partition as the mean decrease in R^2 when the values of a feature are randomly shuffled, averaged over 20 repetitions. The Results section presents both methods side by side, which allows a direct comparison between how much a feature contributes to splitting during training and how much it contributes to predictions on unseen data.

Results and Discussions

Quality of the Data Integration

Table 5 reports the proportion of breakdowns preceded by at least one operational entry on the same machine. A 30-day look-back window links 156 of 159 breakdowns (98.1%) to at least one prior operational entry. The proportion falls to 109 of 159 (68.6%) at a one-day window, leaving 50 breakdowns (31.4%) with no operational entry on the preceding day. These breakdowns have one-day features of zero, which cannot distinguish the day before failure from any other day, a structural limitation examined in the Length of the Observation Window subsection.

The integrated dataset comprises 2,734 machine-day observations with no missing values and no duplicate machine-date pairs, covering all 12 machines on 352 of the 366 calendar days. Record attrition is documented at each pre-processing stage. CMMS records fall from 232 tickets to 224 after date and identifier validation, to 178 within the Electrical and Mechanical categories, and to 159 genuine breakdown events after non-repair

Table 5. Proportion of breakdowns with prior operational records

Look-back window	Breakdowns linked	Percentage	Median entries per breakdown
1 day	109 of 159	68.6	3
3 days	135 of 159	84.9	7
7 days	150 of 159	94.3	20
14 days	155 of 159	97.5	37
30 days	156 of 159	98.1	73

Table 6. Model performance under time-based validation

Model	R^2	MAE (days)	RMSE (days)
Mean predictor (reference)	−0.149	11.47	13.41
Linear Regression	−0.056	10.69	12.85
Decision Tree (depth 6)	−0.998	13.85	17.68
Gradient Boosting	−0.440	12.43	15.01
Random Forest	−0.209	11.64	13.76

Table 7. Ablation study and contribution of the operational log (Random Forest)

Feature set	Features	R^2	MAE (days)
A. Maintenance history only (CMMS only)	2	−0.730	13.22
B. A + operational load	12	−0.341	11.75
C. B + filter events	22	−0.298	11.88
D. C + technical faults (complete)	32	−0.209	11.64

Table 8. Effect of splitting strategy on reported performance

Configuration	R^2	MAE (days)
Time-based split	−0.195	11.57
Time-based split with SMOTE-R in training partition	−0.064	11.01
Random 78/22 split	+0.739	5.87
SMOTE-R before splitting, then random split	+0.754	5.33

requests are removed. The ODL falls from 14,316 raw entries to 13,295 after the exclusion of 1,021 entries (7.1%) without a valid machine code. Together with the linkage rates in Table 5, the complete panel and the documented attrition indicate that the integration objective has been met.

Predictive Performance and the Contribution of the Operational Log

Table 6 reports performance for the October–December test period. The difference between the Random Forest values in Tables 6 and 8 (−0.209 and −0.195) reflects the different configurations detailed in the Models and hyperparameters subsection.

No model achieves a positive R^2, and only Linear Regression outperforms the naive mean predictor. The relative ranking across architectures is nonetheless instructive. Linear Regression, the simplest trained model, achieves the highest R^2, whereas Gradient Boosting and the single Decision Tree, the two learners most prone to fitting noise, perform worst. This pattern is consistent with a low signal-to-noise setting, in which additional model flexibility captures noise in the training set rather than structure that generalizes. The expanding-window rolling-origin scheme gives the Random Forest a mean R^2 of −0.205, with fold values ranging from −0.400 to +0.272, and its MAE is lower than that of the mean baseline in three of the four folds. Table 7 reports the change in performance as feature groups are added sequentially.

The largest single improvement occurs when the operational-load group is added, although in a sequential ablation the first group added tends to capture shared signal that later groups would otherwise provide. The progression is not strictly monotonic in mean absolute error, since stage C shows a marginal increase over stage B.

The contribution of the operational log was isolated by comparing the CMMS-only variant with the full feature set, as described in the Models and hyperparameters subsection. The single temporal split, averaged over three

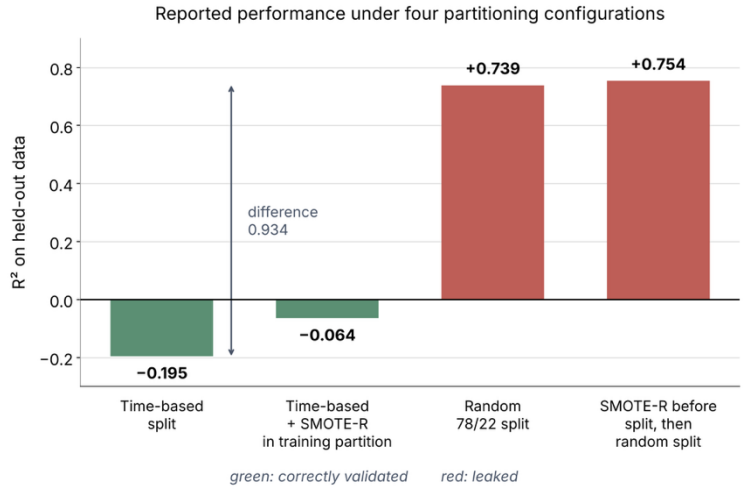


Figure 4. Coefficient of determination under four partitioning configurations. Correctly validated configurations are shown in the left pair, leaked configurations in the right pair.

seeds, shows mean absolute error decreasing by 1.57 days, from 13.19 to 11.62. The rolling-origin evaluation gives a more mixed picture. The operational features reduce error in the September and October folds but increase it in August and November. The mean across folds shifts only from 11.27 days without the operational log to 11.05 days with it, a reduction of 0.22 days.

The effect of the operational log is therefore small and varies in direction across folds, so no definitive predictive benefit is claimed. The larger gain on the single temporal split may partly reflect the particular partition date, and the contribution of operational records remains inconclusive at the present data volume. Reporting only the single-split estimate would overstate the support that the evidence provides.

Effect of the Validation Design

Table 8 and Figure 4 report the results obtained by varying the partitioning procedure and the placement of SMOTE-R while holding the features, model, and hyperparameters constant.

The first and third configurations in Table 8 differ by 0.934 in R^2 and by more than five days in mean absolute error. The gap arises from how observations are allocated. Temporally adjacent machine-day observations belong to the same failure episode and have near-identical feature vectors, as described in the Validation Design and Metrics subsection. Random allocation places such near-duplicates on both sides of the split, which allows the model to interpolate between them, and it also removes the need to predict a later period, since the random test set is spread across the whole year. Applying synthetic oversampling before partitioning raises R^2 slightly further, from +0.739 to +0.754.

Evaluations without temporal separation on panel data of this kind can therefore produce strongly positive R^2 values that do not reflect performance on future periods. The random-split results are retained alongside the temporal baseline to document the size of this effect. Leakage of this kind has been identified as a widespread problem in machine-learning-based research [18], and partitioning procedures are not always reported.

SMOTE-R applied within the training partition improves R^2 for all three oversampling factors tested, reaching −0.101, −0.064, and −0.116 for factors of 1.5, 2.0, and 3.0, compared with −0.195 without resampling. Table 8 shows the result for a factor of 2.0. The gain is not monotonic in the oversampling factor, every R^2 remains negative, and all three results come from a single seed, so the improvement cannot yet be distinguished from random variation.

Length of the Observation Window

Short observation windows capture only recent operational activity, whereas mechanical degradation may develop over weeks, so windows from 1 to 120 days were evaluated. Three window lengths were designated in advance to match planning periods at the site, namely 7 days (weekly maintenance scheduling), 30 days

Table 9. All evaluated observation windows, with the planning periods marked

Window (days)	R^2	MAE (days)	Window (days)	R^2	MAE (days)
1	−0.353	12.45	30 (monthly)	−0.430	12.56
3	−0.466	12.91	45	−0.902	13.99
5	−0.511	13.00	60 (two-monthly)	−0.213	11.46
7 (weekly)	−0.564	13.32	90	−0.504	11.61
14	−0.703	13.71	120	−0.134	9.94

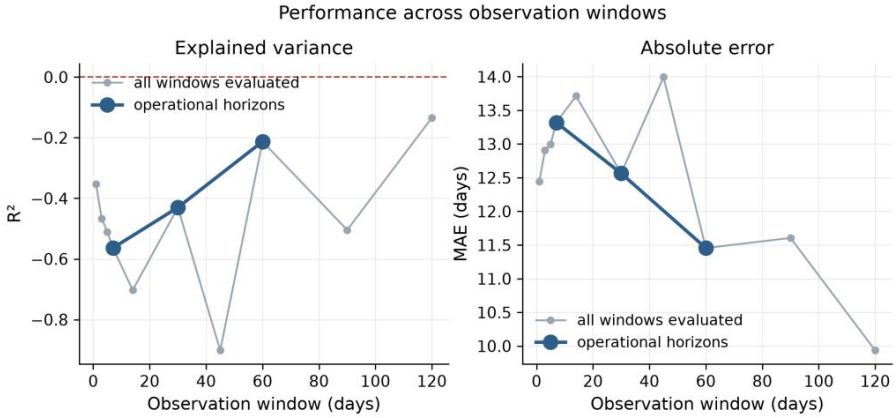


Figure 5. Performance across observation windows from 1 to 120 days. The three planning windows are highlighted, and all evaluated windows are shown for context.

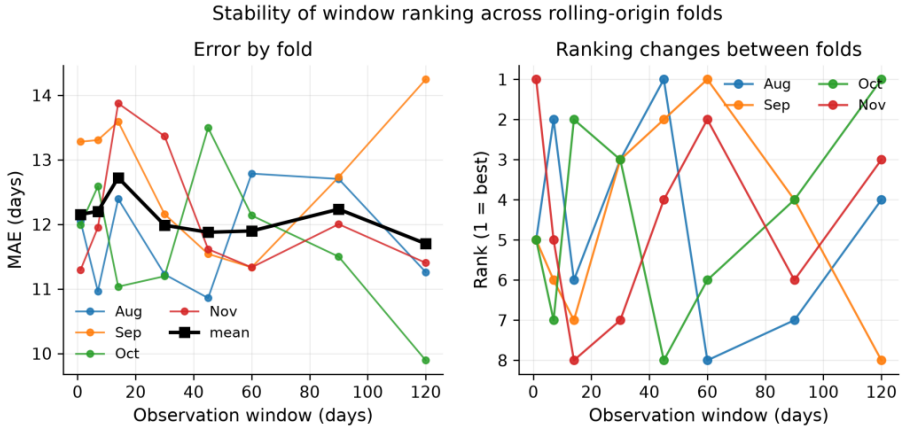


Figure 6. Stability of window ranking across four rolling-origin folds. The best-performing window differs in every fold.

(monthly operational planning), and 60 days (two-month procurement planning). Each window model uses the count and duration features of the three event groups computed over that window alone, together with the number of days elapsed since the last episode. Table 9 reports all evaluated windows.

Performance in Table 9 does not change monotonically with window length. Figure 5 plots this relationship over the full range from 1 to 120 days, with the three planning windows highlighted among all the windows evaluated. The irregularity is visible across the whole range rather than only at the lengths listed in Table 9. Eight of the ten windows were re-evaluated across the four rolling-origin folds to determine whether the ranking in Table 9 reflects the data or the particular test period. The best window differs in every fold: 45 days in August, 60 days in September, 120 days in October, and 1 day in November. Figure 6 presents the same instability as a ranking, with the order of windows changing from one-fold to the next. The difference in MAE between the best and worst windows averages 1.02 days within a fold, whereas across folds the MAE of a single window varies with a standard deviation of 1.05 days on average and up to 1.83 days.

Variation across evaluation periods is therefore at least as large as variation across window lengths, and no single window performs consistently best. The three planning windows are retained for their relevance to site-level planning rather than for any empirical advantage.

Table 10. Feature attribution by two methods

Feature	Permutation, ΔR^2 (test)	MDI (training)	Rank, permutation	Rank, MDI
Technical fault count, 30-day window	0.141	0.063	1	8
Filter duration, 30-day window	0.134	0.082	2	3
Technical fault duration, 30-day window	0.126	0.131	3	2
Days since last episode	0.115	0.151	4	1
Operational load count, 7-day window	0.022	0.018	5	14

The one-day window also faces a structural limitation. Table 5 shows that only 68.6% of breakdowns have any recorded operational activity on the preceding day, so for the remaining third of breakdowns the one-day features are zero. The fact that the one-day window nevertheless outperforms several longer windows in Table 9 further indicates that the window ranking is unstable.

Feature Attribution

Table 10 reports the five leading features by permutation importance on the test set, together with their rank under mean decrease in impurity (MDI).

The two attribution methods diverge substantially. Permutation importance ranks the 30-day technical-fault count first, whereas MDI ranks it eighth. Conversely, the days-since-the-last-episode feature ranks first under MDI but fourth under permutation importance, and the leading operational-load feature ranks fifth under permutation importance but fourteenth under MDI. The discrepancy is systematic rather than incidental. MDI credits a feature for the variance reduction its splits achieve on the training data, a measure known to favour high-cardinality continuous variables [21], whereas permutation importance measures the loss of predictive accuracy when a feature is perturbed on unseen data. A feature may therefore score highly under MDI yet contribute little out of sample. Correlated features also share attribution, so the scores of several features cannot be summed to estimate their joint share of the model's decisions.

The ablation in Table 7 is consistent with this picture. The operational-load group produces the largest single gain when it is added first, despite the low permutation ranks of its individual features. This suggests substantial redundancy across groups, possibly because operational load, maintenance history, and event counts all partly reflect overall machine usage. Permuting one feature at a time understates the importance of such correlated groups, since the remaining features carry much of the same information. Individual variables may therefore contribute little at the margin, while their combination carries a larger share of the signal.

The three highest-ranked features under permutation importance all use the 30-day window. This does not contradict the observation window analysis, which evaluated each window in isolation and found no consistently best length, but it indicates that the 30-day features carry the most information when all windows are available together. One caveat applies to their interpretation. The top-ranked feature, the 30-day technical-fault count, draws 85.1% of its total variance from temporal variation within machines and only 14.9% from differences between machines, so its contribution does not merely reflect machine identity. Its lag-7 autocorrelation reaches 0.85, which indicates a slowly moving signal. The feature is therefore better read as a marker of persistent disturbance levels than as a short-term precursor immediately before failure.

Component Reliability

Breakdowns were attributed to components by matching ticket titles and descriptions against a list of Indonesian component keywords, such as *ballmatic* (ball valve), *timbangan* (weighing unit), and *rantai* (chain). Each ticket was assigned to the first matching component, so a ticket that mentions several components is counted once. MTBF was computed as the total number of machine-days in the study period (12 machines × 366 days) divided by the number of breakdowns attributed to each component. Table 11 reports the results.

Table 11 lists the eight components with at least five breakdowns. The remaining 27 breakdowns are spread across the conveyor (3), motor and pump (3), and solenoid (2), together with 19 tickets that could not be mapped to a specific component. These figures are descriptive and do not depend on model performance. They assume continuous operation throughout the period, since actual running hours per machine are not recorded daily. The resulting values should therefore be read as upper-bound estimates that rank component reliability in

Table 11. MTBF by component

Component	Breakdowns	MTBF (days)	Component	Breakdowns	MTBF (days)
Ball valve (ballmatic)	36	122.0	Float and dipstick	7	627.4
Weighing unit, load cell	28	156.9	Press and auto sealer	7	627.4
Chain drive, gearbox	25	175.7	Hose and pneumatic	6	732.0
Control panel, push button	18	244.0	Limit switch	5	878.4

Table 12. Operational backtest over the full grid of 1,104 machine-days

Strategy	Episodes detected	Median lead (days)	Alert machine-days	Share of days	Precision
Model, 14-day threshold	3 of 26	6	23	2.10%	17.4%
Calendar rule, 14-day window	21 of 26	13	560	50.7%	28.8%
Model, 21-day threshold	21 of 26	18	377	34.1%	43.2%
Calendar rule, 21-day window	22 of 26	14	560	50.7%	35.0%
Model, 30-day threshold	25 of 26	30	929	84.1%	51.1%
Calendar rule, 30-day window	23 of 26	20	560	50.7%	39.8%

relative terms. The ball valve accounts for 36 of the 159 breakdowns (22.6%) and is a logical first target for component-level attention.

Maintenance Prioritization

The predicted TTF ($\hat{Y}$) is combined with two further indicators to construct a composite maintenance priority score. The first is the number of days elapsed since the most recent maintenance episode, divided by the MTBF of the machine's dominant component. The dominant component is the one with the most breakdowns on the machine. The second is recent disturbance intensity, measured as the number of technical-fault and filter events in the preceding 30 days. Each indicator is min–max normalized across the 12 machines, so the score ranks machines relative to one another, and the three are weighted at 0.4, 0.4, and 0.2, respectively. Combining a model prediction with descriptive indicators means that scheduling does not rest on the model alone. The independence from the model is only partial, however, since two of the three indicators also enter the model as features. The weights and thresholds are initial values that require calibration with site maintenance managers, so the output functions as a decision-support heuristic rather than an automatic dispatch instruction. The system output presents each predicted TTF with an uncertainty band of ±11.6 days, equal to the mean absolute error from temporal validation. Machines are classified as high urgency when the predicted TTF is at most 14 days and as medium urgency when it is between 15 and 30 days. The resulting band of about 23 days is wider than the 16-day interval between these thresholds, so a point prediction alone cannot reliably place a machine in a category. Presenting point estimates without this band would therefore overstate the precision achieved.

Operational Backtest

Statistical metrics do not answer the questions a maintenance department would ask, namely how many failures a running system would have flagged, how much warning it would have given, and how often it would have been wrong. The model was therefore trained on data from January to September and applied to each day of October to December, raising a warning whenever the predicted TTF was at or below a threshold.

The baseline is a calendar-based rule that flags a machine once the days elapsed since its last episode reach the median of that quantity over the test period. The rule uses a single fixed interval, so its number of alert machine-days is identical at all three thresholds, and deriving the interval from the test period gives the rule a slight advantage over the model. The thresholds of 14, 21, and 30 days also define the evaluation window. An episode counts as detected when at least one alert occurs within that window before it, and the lead time is measured from the first such alert. Precision is the share of alert machine-days followed by an episode on the same machine within the same window.

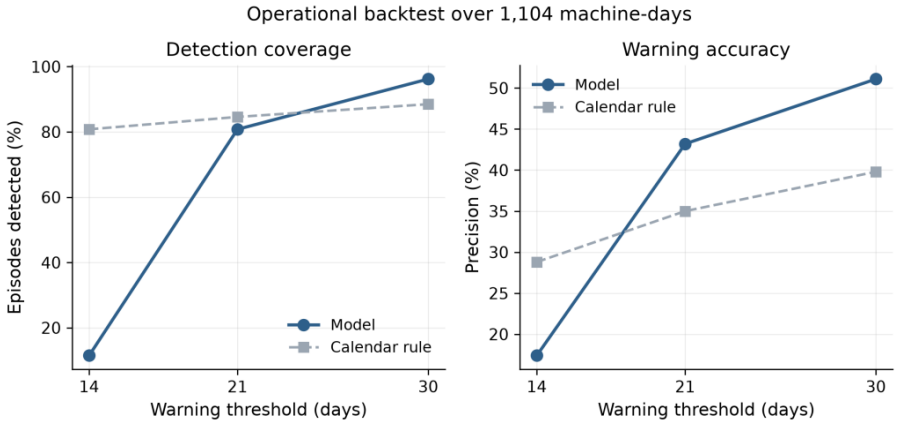


Figure 7. Operational backtest: detection coverage and precision across warning thresholds, compared with the calendar-based rule.

The simulation covers the full operating grid of 12 machines over 92 days, or 1,104 machine-days, which reflects deployment conditions in which every machine is assessed daily. Twenty-six maintenance episodes occurred during the test period, and Table 12 reports the results.

The model detects 21 of 26 episodes at the 21-day threshold, against 22 for the calendar rule. It requires 377 alert machine-days rather than 560, achieves 43.2% precision rather than 35.0%, and provides four additional days of warning, so comparable coverage requires roughly one-third fewer alert machine-days. The 30-day threshold detects more episodes but raises warnings on 84% of all machine-days, which offers little prioritization and would not be usable in practice. Figure 7 traces detection coverage and precision across the full range of warning thresholds for both strategies. The coverage of the model rises steeply with the threshold while that of the calendar rule stays nearly flat, and the precision advantage of the model appears only at the 21-day threshold and above.

A second version of the simulation, restricted to the 607 machine-days used in model evaluation, reports 83.6% precision at the 21-day threshold. The restricted panel excludes days with no breakdown in the following 60 days, which removes most of the days that generate false alarms, so the apparent improvement reflects the smaller denominator. Both versions are reported, with the full grid treated as the primary result. Presenting only the restricted figure would overstate performance in a way structurally similar to the random-split effect described earlier.

These backtest results should be read with caution. The period contains only 26 episodes, so the detection and precision estimates carry wide uncertainty. The calendar rule simplifies preventive practice, which normally applies component-specific intervals.

Discussion

Three results deserve separate treatment, since they point in different directions. The first is that the integration succeeded. Operational logs kept by production staff can be aligned with engineering maintenance records at daily resolution, with 98.1% of breakdowns linked to prior operational activity within 30 days and a fused dataset free of missing and duplicated records. This data-engineering result does not depend on any model, and it matters because the barriers to using operational records are often described as organizational [6], whereas the present case shows that the technical integration is tractable with modest effort.

The second result is the limited predictive power observed under out-of-time validation. No model achieves a positive R^2, and the advantage of operational features does not persist across folds. Several explanations fit the evidence and are not mutually exclusive. The number of failure episodes is small, at 110 across 12 machines and 12 months. The median spacing between breakdowns is 13 days, and TTF values are concentrated at short horizons, with a median of 16 days. Separating gradual from sudden failure modes further reduced performance, with mean rolling-origin R^2 values of −0.254 and −0.309 for gradual and sudden failures, respectively, compared with −0.205 for all breakdowns combined. This most plausibly reflects the smaller number of episodes in each group, which prevents a firm conclusion about the failure modes themselves.

A further possibility is that the Precursor Signal hypothesis does not hold uniformly. The hypothesis assumes that failures are preceded by a detectable degradation phase, which is plausible for wear in a valve or a chain drive but implausible for a push button or a limit switch, where failure is almost instantaneous. Operational data cannot predict breakdowns that have no latent phase, and 53 of the 159 breakdowns (33%) involve components classified as prone to sudden failure.

The methodological result is the most transferable of the three. The 0.934 difference in R^2 attributable to the partitioning procedure exceeds every difference between feature sets or window lengths and is close to the largest difference between algorithms (0.942). The corresponding 5.7-day gap in mean absolute error exceeds every other difference observed in the study. Such leakage is well documented across machine-learning research [18], so headline figures cannot be compared across studies unless the partitioning procedure is stated. A result obtained under random splitting reveals little about performance in deployment.

Metric choice also warrants caution. The model appears to fail under R^2, yet the backtest shows that it performs comparably to a calendar rule on detection coverage while requiring roughly one-third fewer alert machine-days and giving four additional days of warning. The two assessments are compatible because they ask different questions of the model. A TTF estimate with a mean absolute error of about 12 days cannot be used to schedule a specific intervention date, but it can still rank which machines to examine over the coming weeks, which is the decision maintenance planners typically face. R^2 alone is therefore an insufficient measure of usefulness in this setting.

The approach adds value from an industrial analytic perspective because it exploits records that already exist, without the capital expenditure of sensor retrofits. The contrast with sensor-based work is instructive. Vallim Filho et al. [9] showed that load-based health indicators can outperform chronological age when derived from sensor data, whereas the present results indicate that comparable signals are considerably harder to recover from manual operational logs.

Limitations

The study covers one facility, one machine type, and one calendar year. The plant produces coating materials on continuous lines, and filling machines elsewhere will differ in duty cycle, component mix, and failure behaviour. No claim of generality beyond this setting is intended.

The study has 110 failure episodes and therefore lacks the power to resolve differences of the size observed between feature sets and window lengths. This limitation likely contributes to both inconclusive results, alongside the possibility that many breakdowns have no detectable latent phase.

Installation dates are not recorded, so no age-based comparison is possible. Production volumes are available only as annual totals, so operational load is represented by the count and duration of operational-load events, which reflect batch turnover rather than throughput. The Cumulative Stress Index could not be computed because daily production volume histories were unavailable. The target is limited to TTF values of at most 60 days, and machine-days without a subsequent episode were excluded rather than treated as censored observations.

The authors constructed the downtime-code taxonomy from the recorded descriptions and the known component structure, and maintenance personnel at this plant validated it. Downtime codes and recording practices may differ between facilities, and a misassigned frequent code would propagate errors through the feature set, so the taxonomy should be remapped with local technicians before it is reused elsewhere. Merging breakdowns within seven days into single episodes is likewise an interpretive judgement about what constitutes one maintenance event. Component attribution relies on keyword matching against free-text descriptions, assigns each ticket to a single component, and inherits any inconsistency in how technicians record their work. MTBF assumes continuous operation and is consequently an upper bound.

The backtest covers three months and 26 episodes, so its detection and precision estimates are highly uncertain. The calendar comparison is also simplified relative to actual preventive practice, and the SMOTE-R results derive from a single random seed.

Conclusion

This study integrated 13,295 operational downtime records with 159 maintenance breakdowns across 12 filling machines over one year and evaluated whether accumulated minor stoppages anticipate major failures.

The integration succeeded, with 98.1% of breakdowns linked to prior operational activity within 30 days and a machine-day panel that is complete and reproducible. A taxonomy of downtime codes proved necessary, since most operational entries record scheduled changeover rather than degradation, and only technical-fault entries showed a positive, albeit weak, association with subsequent breakdowns. The classification rests on the physical mechanism behind each code and has been validated by plant maintenance personnel. Drawing such a distinction is an essential step before operational records can be used for maintenance analytic.

The predictive capability of operational downtime logs remains unproven under rigorous temporal validation. Adding operational features reduces mean absolute error by 1.57 days on a single split, yet the gain falls to 0.22 days across four folds, and no observation window is consistently optimal. This result is inconclusive, which likely reflects the limited number of failure episodes and possibly the absence of a detectable latent phase in many breakdowns. No precursor effect is claimed beyond what the data support.

The most transferable finding concerns evaluation rather than prediction. A random train–test split raises R^2 by 0.934 relative to a time-based split with features, model, and hyperparameters held constant, because adjacent machine-day observations share failure episodes and near-identical features. The corresponding 5.7-day gap in mean absolute error exceeds every other difference measured in the study, which implies that reported performance cannot be interpreted without knowing the partitioning procedure.

Practical value nevertheless remains. An operational backtest showed that the model performed comparably to a calendar-based rule on detection coverage while requiring roughly one-third fewer alert machine-days and providing four additional days of warning. Component-level analysis identified the ball valve as the dominant contributor, accounting for 22.6% of breakdowns. A system built on records the plant already keeps can therefore support inspection prioritization, even where precise failure timing remains out of reach.

Future work should extend the observation period to increase the number of failure episodes, although a longer horizon may introduce concept drift as operators, procedures, and equipment change over time. Recording production volume at daily granularity would allow cycle-based workload intensity to be constructed. Survival analysis may suit the time-to-event structure better than regression, particularly because it can retain censored observations. Targeted instrumentation of the components with the shortest MTBF would show whether the predictive limitation lies in the signal itself or in the resolution of the manual records.

Declaration of Generative AI and AI-Assisted Technologies in the Writing Process

The authors used Claude (Anthropic) to improve the readability and language quality of this manuscript during its preparation. The authors reviewed and edited the content as necessary and accept full responsibility for the published work.

Acknowledgement

The authors thank the industrial partner that provided access to the maintenance and operational records analyzed in this study. The partner is not named at its request, and the underlying records cannot be made publicly available under the terms of the confidentiality agreement. The authors declare that no external funding was received for this research and report no conflicts of interest.

References

[1] M. A. Hoffmann and R. Lasch, "Roadmap for a successful implementation of a predictive maintenance strategy," in *Smart and Sustainable Supply Chain and Logistics*, Cham: Springer, 2020, pp. 10–25, doi: https://doi.org/10.1007/978-3-030-61947-3_29.

[2] S. Y. Tanjung, K. Yahya, and S. Halim, "Predicting the readiness of Indonesia manufacturing companies toward Industry 4.0: A machine learning approach," *Jurnal Teknik Industri: Jurnal Keilmuan dan Aplikasi Teknik Industri*, vol. 23, no. 1, pp. 1–10, May 2021, doi: https://doi.org/10.9744/jti.23.1.1-10.

[3] M. Achouch et al., "On predictive maintenance in Industry 4.0: Overview, models, and challenges," *Applied Sciences*, vol. 12, no. 16, p. 8081, Aug. 2022, doi: https://doi.org/10.3390/app12168081.
[4] T. Bitam, A. Yahiaoui, D. E. Boubiche, R. Martinez-Pelaez, H. Toral-Cruz, and P. Velarde-Alvarado, "Artificial intelligence of things for next-generation predictive maintenance," *Sensors*, vol. 25, no. 24, p. 7636, Dec. 2025, doi: https://doi.org/10.3390/s25247636.
[5] C. F. Chien and C. C. Chen, "Data-driven framework for tool health monitoring and maintenance strategy for smart manufacturing," *IEEE Transactions on Semiconductor Manufacturing*, vol. 33, no. 4, pp. 644–652, Nov. 2020, doi: https://doi.org/10.1109/TSM.2020.3024284.
[6] W. W. Tiddens, A. J. Braaksma, and T. Tinga, "Unlocking the potential of predictive maintenance for intelligent manufacturing," *Sustainability*, vol. 14, no. 11, p. 6567, May 2022, doi: https://doi.org/10.3390/su14116567.
[7] A. Benhanifia, Z. Ben Cheikh, P. Moura Oliveira, A. Valente, and J. Lima, "Systematic review of predictive maintenance practices in the manufacturing sector," Intelligent Systems with Applications, vol. 26, p. 200501, Apr. 2025, doi: https://doi.org/10.1016/j.iswa.2025.200501.
[8] A. Theissler, J. Perez-Velazquez, M. Kettelgerdes, and G. Elger, "Predictive maintenance enabled by machine learning: Use cases and challenges in the automotive industry," Reliability Engineering & System Safety, vol. 215, p. 107864, Nov. 2021, doi: https://doi.org/10.1016/j.ress.2021.107864.
[9] A. R. A. Vallim Filho, D. F. Moraes, M. V. B. A. Vallim, L. S. Silva, and L. A. Silva, "A machine learning modeling framework for predictive maintenance based on equipment load cycle," *Energies*, vol. 15, no. 10, p. 3724, May 2022, doi: https://doi.org/10.3390/en15103724.
[10] J. Azeta et al., "Artificial intelligence and robotics in predictive maintenance: A comprehensive review," *Frontiers in Mechanical Engineering*, vol. 11, p. 1722114, 2026, doi: https://doi.org/10.3389/fmech.2025.1722114.
[11] W. S. Loh, L. Ling, R. J. Chin, S. H. Lai, K. K. Loo, and C. S. Seah, "A comparative analysis of missing data imputation techniques on sedimentation data," *Ain Shams Engineering Journal*, vol. 15, no. 6, p. 102717, Jun. 2024, doi: https://doi.org/10.1016/j.asej.2024.102717.
[12] S. Zhang, "Nearest neighbor selection for iteratively k-nearest neighbor imputation," *Journal of Systems and Software*, vol. 174, p. 110887, Apr. 2021, doi: https://doi.org/10.1016/j.jss.2020.110887.
[13] P. Branco, L. Torgo, and R. P. Ribeiro, "A survey of predictive modeling on imbalanced domains," *ACM Computing Surveys*, vol. 49, no. 2, pp. 1–50, Aug. 2016, doi: https://doi.org/10.1145/2907070.
[14] L. Torgo, R. P. Ribeiro, B. Pfahringer, and P. Branco, "SMOTE for regression," in *Progress in Artificial Intelligence*, vol. 8154, Berlin: Springer, 2013, pp. 378–389, doi: https://doi.org/10.1007/978-3-642-40669-0_33.
[15] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, Oct. 2001, doi: https://doi.org/10.1023/A:1010933404324.
[16] C. Bergmeir and J. M. Benitez, "On the use of cross-validation for time series predictor evaluation," *Information Sciences*, vol. 191, pp. 192–213, May 2012, doi: https://doi.org/10.1016/j.ins.2011.12.028.
[17] D. R. Roberts et al., "Cross-validation strategies for data with temporal, spatial, hierarchical, or phylogenetic structure," *Ecography*, vol. 40, no. 8, pp. 913–929, Aug. 2017, doi: https://doi.org/10.1111/ecog.02881.
[18] S. Kapoor and A. Narayanan, "Leakage and the reproducibility crisis in machine-learning-based science," *Patterns*, vol. 4, no. 9, p. 100804, Sep. 2023, doi: https://doi.org/10.1016/j.patter.2023.100804.
[19] T. O. Hodson, "Root-mean-square error (RMSE) or mean absolute error (MAE): When to use them or not," *Geoscientific Model Development*, vol. 15, no. 14, pp. 5481–5487, Jul. 2022, doi: https://doi.org/10.5194/gmd-15-5481-2022.
[20] D. Chicco, M. J. Warrens, and G. Jurman, "The coefficient of determination R-squared is more informative than SMAPE, MAE, MAPE, and RMSE in regression analysis evaluation," *PeerJ Computer Science*, vol. 7, p. e623, Jul. 2021, doi: https://doi.org/10.7717/peerj-cs.623.
[21] C. Strobl, A. L. Boulesteix, A. Zeileis, and T. Hothorn, "Bias in random forest variable importance measures: Illustrations, sources and a solution," *BMC Bioinformatics*, vol. 8, p. 25, Jan. 2007, doi: https://doi.org/10.1186/1471-2105-8-25.