scientific reports

Check for updates

OPEN

An explainable machine learning framework for railway predictive maintenance using data streams from the metro operator of Portugal

Silvia García-Méndez[1]✉, Francisco de Arriba-Pérez[1], Fátima Leal[2], Bruno Veloso[3,4], Benedita Malheiro[4,5] & Juan Carlos Burguillo-Rial[1]

The public transportation sector generates large volumes of sensor data that, if analyzed adequately, can help anticipate failures and initiate maintenance actions, thereby enhancing quality and productivity. This work contributes to a real-time data-driven predictive maintenance solution for Intelligent Transportation Systems. The proposed method implements a processing pipeline comprised of sample pre-processing, incremental classification with Machine Learning models, and outcome explanation. This novel online processing pipeline has two main highlights: (i) a dedicated sample pre-processing module, which builds statistical and frequency-related features on the fly, and (ii) an explainability module. This work is the first to perform online fault prediction with natural language and visual explainability. The experiments were performed with the MetroPT data set from the metro operator of Porto, Portugal. The results are above 98 % for F-measure and 99 % for accuracy. In the context of railway predictive maintenance, achieving these high values is crucial due to the practical and operational implications of accurate failure prediction. In the specific case of a high F-measure, this ensures that the system maintains an optimal balance between detecting the highest possible number of real faults and minimizing false alarms, which is crucial for maximizing service availability. Furthermore, the accuracy obtained enables reliability, directly impacting cost reduction and increased safety. The analysis demonstrates that the pipeline maintains high performance even in the presence of class imbalance and noise, and its explanations effectively reflect the decision-making process. These findings validate the methodological soundness of the approach and confirm its practical applicability for supporting proactive maintenance decisions in real-world railway operations. Therefore, by identifying the early signs of failure, this pipeline enables decision-makers to understand the underlying problems and act accordingly swiftly.

Keywords Explainable sensor-driven computational intelligence, Intelligent transportation systems, Online supervised machine learning, Predictive maintenance, Railway sector safety and reliability

The digitalization of critical infrastructures and the adoption of Industry 4.0 technologies have driven profound transformations in sectors such as healthcare[1], manufacturing[2,3], and transportation[4,5]. In the latter, the massive deployment of sensors and monitoring systems has generated large volumes of data, allowing for continuous monitoring of asset status. This new scenario has led to the development of predictive maintenance (PdM) strategies, which aim to anticipate failures in real time by analyzing historical patterns. Compared to corrective approaches, PdM allows for intervention just before a breakdown occurs, thereby improving system availability, reducing operating costs, and enhancing safety standards.

Online monitoring generates large volumes of heterogeneous sensor data at a high pace, falling into the category of big data[6,7]. These endless data streams, when adequately mined, can anticipate failures and support decision-making in multiple industry domains[8,9], playing an essential role in the implementation of PdM[10].

[1]Information Technologies Group, atlanTTic, University of Vigo, Vigo, Spain. [2]REMIT, Universidade Portucalense, Porto, Portugal. [3]Faculty of Economics, University of Porto, Porto, Portugal. [4]INESC TEC, Porto, Portugal. [5]ISEP, Porto, Portugal. ✉email: sgarcia@gti.uvigo.es

By anticipating failures, PdM allows companies to increase productivity and reduce operational costs[11]. In the railway context, characterized by dynamic environments, high-reliability demands, and components subject to constant wear, the effective implementation of predictive solutions represents a key opportunity to optimize the operation and maintenance of critical transportation infrastructures.

In the public passenger transportation domain (air, rail, and road), failures cause delays and cancellations, lower the reputation, and increase the costs of the involved operators[12,13]. Specifically, in railway transports, wheel and door failures are a significant cause of delays[10]. Nowadays, trains generate countless sensor data streams, *e.g.*, light, position, pressure, temperature, or vibration, which online predictive algorithms can use. Data stream analysis continuously monitors and detects relevant changes in the status of components and systems, alerting decision-makers to take action. Therefore, data stream processing is essential for real-time PdM, namely, for Intelligent Transportation System (ITS) applications to make transportation networks safer and more sustainable[14,15].

In this context, PdM relies extensively on Machine Learning (ML) techniques. However, many of the employed techniques are opaque[16,17], leaving decision-makers unclear about the reasoning behind them. Transparent predictive techniques, whether interpretable or explainable, provide intelligible learning results. In this context, interpretable models are those whose reasoning can be easily understood by humans, such as decision trees, regression, or rule-based predictors. Conversely, black box models, like support vector machines or deep neural networks, require ad hoc explainability methods[18,19]. Therefore, early detection and explanation of faults allow maintenance experts to identify the underlying problems and apply corrective measures.

In the specific case of the railway sector, the focus of our research, PdM has assumed a crucial role to enhance service reliability, reduce operating costs, and improve system safety[20]. In this sense, existing solutions aim to continuously monitor data from onboard sensors to anticipate failures, enabling efficient planning of maintenance tasks and preventing unexpected service downtime. Specifically, the works in state-of-the-art have been developed to detect anomalies in critical components, including air production units, wheels, doors, brakes, and electrical systems, among others[21,22]. However, the most recent work in this area has incorporated ML techniques applied to real-time data to identify behavioral patterns that precede operational failures[23]. In contrast, there also exist solutions that approach the problem from a non-classification point of view[21,24–26]. However, despite these advances, the field continues to face significant challenges, including the lack of interpretability in models, the limited integration of real-time explanations, and the need to adapt methods to variable conditions and scenarios with highly unbalanced classes. This has motivated the development of more transparent approaches that can combine predictive accuracy with operational explainability, as in our case, thereby facilitating their practical adoption in real-world railway environments.

By classifying sensor data streams on the fly with the help of ML algorithms and explainability techniques, the current work aims to address both factors simultaneously. Overall, it enhances transportation operators' quality of service, reputation, productivity, and profits through real-time natural language and visual explainability. The ultimate objective of this research is to mine the incoming train sensor data stream from a public railway operator to anticipate failures and support maintenance decisions. The solution exploits a transparent, data-driven PdM method for the public transportation sector. It is composed of three main modules: (*i*) data pre-processing, encompassing feature engineering, analysis, and selection; (*ii*) stream-based failure classification; and (*iii*) failure explanation supported by ML algorithms[27,28] and Natural Language Processing (NLP) techniques[29]. Therefore, compared to the literature, the proposed system contributes a novel real-time data-driven PdM solution, comprising dedicated sample pre-processing and classification, together with textual and visual outcome explanations for both ML and maintenance experts. The method was tested using the MetroPT data set collected from trains in Metro do Porto, Portugal. The empirical results with the Adaptive Random Forest Classifier exhibit competitive accuracy and micro and macro *F*-measure values. Moreover, each incoming event is classified, and its class label is explained by detailing the sensors' abnormal behavior and other relevant features.

The rest of this paper is organized as follows. Section "Related work" reviews the PdM background, focusing on stream-based and explainable ML. Section "Proposed method" describes the proposed method composed of data processing, classification, and explainability modules. Section "Experimental results" presents the experiments performed and the results of the empirical evaluation. Finally, Section "Conclusion" summarizes and discusses the outcomes of this work.

Related work

The Internet of Things and Industry 4.0 has contributed to the exponential increase in data generated by devices, machines, manufacturing lines, and industrial processes[30,31]. This big data scenario presents new challenges to researchers. The current work, which focuses on the processing of such data streams for PdM, aims to identify and explain early signs of failure. Early detection of a failure improves process management and reduces the associated economic, environmental, and social costs[32,33]. In the railway sector, PdM reduces operational costs and train downtime as well as boosts the quality of service.

The literature identifies four maintenance categories[9]: (*i*) corrective—maintenance takes place after the fault has occurred; (*ii*) preventive—uses time slots to replace components; (*iii*) predictive—performs the early detection of failures; and (*iv*) prescriptive— provides valuable suggestions for extending the life of the equipment. As a data-intensive approach, PdM requires specialized pre-processing techniques to resolve inconsistencies and redundancies and prepare the highly heterogeneous data for the subsequent phases[34]. Moreover, according to Xie et al.[35], as most approaches are based on opaque models, maintenance professionals remain unaware of how the results were obtained.

Predictive maintenance for the railway sector

Existing surveys on PdM for railway operators analyze and compare multiple data-driven approaches[36,37], including the prediction of failures as well as of the remaining service life. Most of these works employ classical ML algorithms, such as decision trees, neural networks, outlier detectors, regressors, and rule-based methods (see Table 1). The latter suggests designing online processing pipelines that comprise dedicated pre-processing and transparent predictive algorithms to help maintenance specialists make well-informed decisions. This related work compares the most recent PdM works in terms of: (*i*) data pre-processing; (*ii*) stream-based classification; and (*iii*) transparency.

Pre-processing comprises tasks such as data cleansing, feature engineering, information fusion, data balancing, feature selection, and feature analysis. Pre-processing is a time-consuming task for knowledge discovery in the context of a data stream. Ramírez-Gallego et al.[38] analyze pre-processing techniques for stream-based applications, contemplating concept drifts, data reduction, ensemble learning, and sliding windows. PdM requires dedicated online feature engineering methods for failure detection. In this context, the literature encompasses stream-based data manipulation approaches that summarize behavior frames using tuples of statistical data[39] or segment the data into intervals to generate features from the analog and digital sensors[21]. *Data-driven predictive maintenance* monitors incoming sensor data in real time to establish mechanical or equipment indicators by exploiting advanced ML methods to dynamically detect functioning patterns and operating conditions[37]. PdM has been explored in the streaming scenario for:

- *Anomaly detection* using outlier detection methodologies[39].
- *Estimation of the remaining useful life* of rail axle bearings through Online Support Vector Regression (OSVR)[40]. To address the big data scenario, the authors propose a heuristic trade-off between accuracy, runtime, and resources required by ML models.
- *Failure prediction* trough sparse autoencoders, using unsupervised methods based on Deep Learning[21,41].

Transparent PdM models help decision-makers to understand the need to replace or repair a particular component. Predictive ML models can be intrinsically interpretable (*e.g.*, decision trees, graph-based approaches, regressors, and rule-based methods) or opaque (*e.g.*, neural networks)[42]. Interpretable models allow users to choose whether or not to trust them, reduce bias, and help discover additional insights[43]. Graph-based approaches can achieve promising performance on PdM perception tasks by revealing the dependency relationship among parts and components of the equipment. Opaque models can be coupled with post hoc techniques for explainability purposes. Explainability techniques can be model-agnostic or model-specific and can be either global (explaining the entire model) or local (explaining each prediction). Examples of local post hoc model-agnostic techniques are the Local Interpretable Model-agnostic Explanations (LIME)[44] and SHapley Additive exPlanations (SHAP)[45]. Alternatively, explanations can be classified under the example-based, feature-importance, knowledge-extraction, and visual categories[17]. The main explainable PdM works found in the literature are listed below.

- Manco et al.[39] propose an unsupervised fault explanation system based on features with abnormal values.
- Allah Bukhsh et al.[46] present an explainable classifier for railway switches using decision trees, random forests, and gradient-boosted trees. They adopt the LIME technique to identify the features that contribute positively or negatively to each class label.
- Ribeiro et al.[41] describe a neural symbolic explainer, which combines oversampling with the AMRules algorithm to describe anomalies detected by an autoencoder.

As previously mentioned, recent PdM solutions seek continuous monitoring of data, regardless of the sector. A representative example is the work by Niu et al.[47], which focused on the airline flight delays problem and addressed the challenge using data-driven dynamics, similar to our study. Within the railway sector, authors have proposed stream-based monitoring, anomaly detection, and anomaly prediction, as seen in the work by Le-Nguyen et al.[24], which led to the development of the InterCE system and subsequent investigations[25,26] for the passenger access system, among others. However, the latter works addressed PdM from a non-classification perspective. The sole exception is the study by Meira et al.[23]. The authors presented an unsupervised real-time anomaly detection system.

Authorship	Objective	Data processing	Decision technique	Decision processing	Explainability
[39]	Fault prediction	Offline	Outlier Detector	Offline	Feature importance
[21]	Fault detection	Offline	Sparse Variational Autoencoder	Offline	–
[46]	Maintenance prediction	Offline	Tree-based classifiers	Offline	Feature importance
[40]	Fault detection	NA	Online Support Vector Regressor	Online	–
[41]	Fault detection	Online	Sparse Autoencoder	Offline	Natural language
[23]	Maintenance prediction	Online	XStream	Online	–
Current proposal	Fault prediction	Online	Explainable classifiers	Online	Natural language and visual

Table 1. Comparison of PdM research.

Research contribution

The research presented in this paper is based on a combination of theoretical and technical foundations that converge on the need for effective, explainable PdM solutions adapted to real-time data processing. First, it draws on existing empirical evidence demonstrating the positive impact of PdM on reducing costs, downtime, and critical failures in railway systems. Second, it builds on the accumulated knowledge of ML techniques applied to data flow analysis, which enables the construction of incremental models capable of adapting to dynamic environments. Furthermore, this research draws on the growing interest in the explainability of models, integrating mechanisms that not only predict failures with high accuracy but also justify their decisions through understandable textual and visual descriptions. Note that the work is based on real data obtained from the railway operator in Portugal, which ensures that the proposed solution responds to real operating conditions.

In short, we propose a novel, transparent, and operational solution for real-time PdM in the railway sector. The proposed method encompasses stream-based data pre-processing, failure classification, and failure explanation supported by NLP techniques. Its main contributions are articulated around three interdependent axes: (*i*) processing, (*ii*) prediction, and (*iii*) explainability. First, a pre-processing module is introduced that is capable of dynamically generating statistical and frequency characteristics by applying FIR filters to sliding windows adapted to the nature of the analog and digital signals in the input data. Second, an incremental classification architecture is applied based on explainable models optimized under a prequential protocol and variance thresholding dimensionality reduction. Third, the explainability module works with global feature-relevance descriptions (textual) and local feature-relevance descriptions (textual and visual), thereby generating automatic natural language and interpretable visualizations that explain the causes of each prediction. The solution was experimentally validated on the MetroPT dataset by Metro do Porto, the operator. The promising results obtained not only significantly exceed the values reported in previous works with the same dataset but also confirm the robustness and practical applicability of the proposed methodology.

While the literature presents a significant number of PdM approaches for the industry in general, scant research has been produced on the online pre-processing, classification, and explanation of railway data. Other approaches have proven effective in controlled or offline environments, achieving good levels of accuracy in identifying anomalies. Among their main advantages is the ability to model complex, nonlinear relationships in the data, which is useful for detecting subtle patterns of degradation. However, many of these proposals have significant limitations in real-life scenarios. In particular, they often lack real-time processing capabilities, making them difficult to apply directly in continuous operating environments, such as railways. It is the case of the works by Manco, G. et al.[39], Allah Bukhsh et al.[46], and Davari et al.[21]. Furthermore, many of these models operate as black boxes, offering no comprehensible explanations for the decisions made, which reduces their acceptability by maintenance technicians, as in Fumeo et al.[40] and Meira et al.[23].

Table 1 compares the research that addresses online and/or transparent PdM for the railway sector. In light of the latter comparison, this work is the first to perform fault prediction in streaming mode, combining natural language and visual explainability.

Proposed method

The online PdM cycle is presented in Fig. 1. It is composed of three main modules: (*i*) data pre-processing (Section "Online data pre-processing"), (*ii*) ML classification (Section "Online classification"), and (*iii*) explainability (Section "Online explainability"). All three modules work over data streams.

Online data pre-processing

Data pre-processing is crucial to ensure high-quality input data and, consequently, accurate classification results. This module comprises feature engineering, feature analysis, and selection procedures.

Feature engineering

Sensor data, especially analog signals, may exhibit random oscillations from sample to sample, corresponding to white noise. To remove this undesirable noise, the analog signals are processed by four sliding-window finite impulse response (FIR) filters[48,49]. A sliding-window FIR filter transforms an original analog signal into a white-noise-free signal as long as its length is adequate[50]. Thus, the initial step is to automatically determine the minimal number of samples or sliding window length of each FIR filter using a subset of the first two days of data:

- Determine the relative minimum values of each feature in the experimental data set.
- Compute the number of samples between relative minimum values of each feature.
- Merge the number of samples of each feature into a list.
- Compute, using the previous list, the average and quartile distribution of the following four intervals: (*i*) from minimum to the first quartile (Q_1), (*ii*) from Q_1 to the median, (*iii*) from median to the third quartile (Q_3), and (*iv*) from Q_3 to the maximum.
- Define the size of the sliding windows of the FIR filters based on the average, Q_1, Q_2, Q_3 values.

The resulting sliding window lengths correspond to the number of samples required to overcome a cold start for each filter. Then, for each sample in the experimental data set, six new features are engineered per window filter—average, standard deviation, Q_1, Q_2, Q_3 and Fast Fourier Transform (FFT[1])—according to Equation 1, where n represents the identifier of the last incoming sample and $X[n]$ the vector holding the values of a particular feature. Note that these statistics were selected based on experimental tests. The first five engineered

[1] The FFT represents the signal in the frequency domain48.

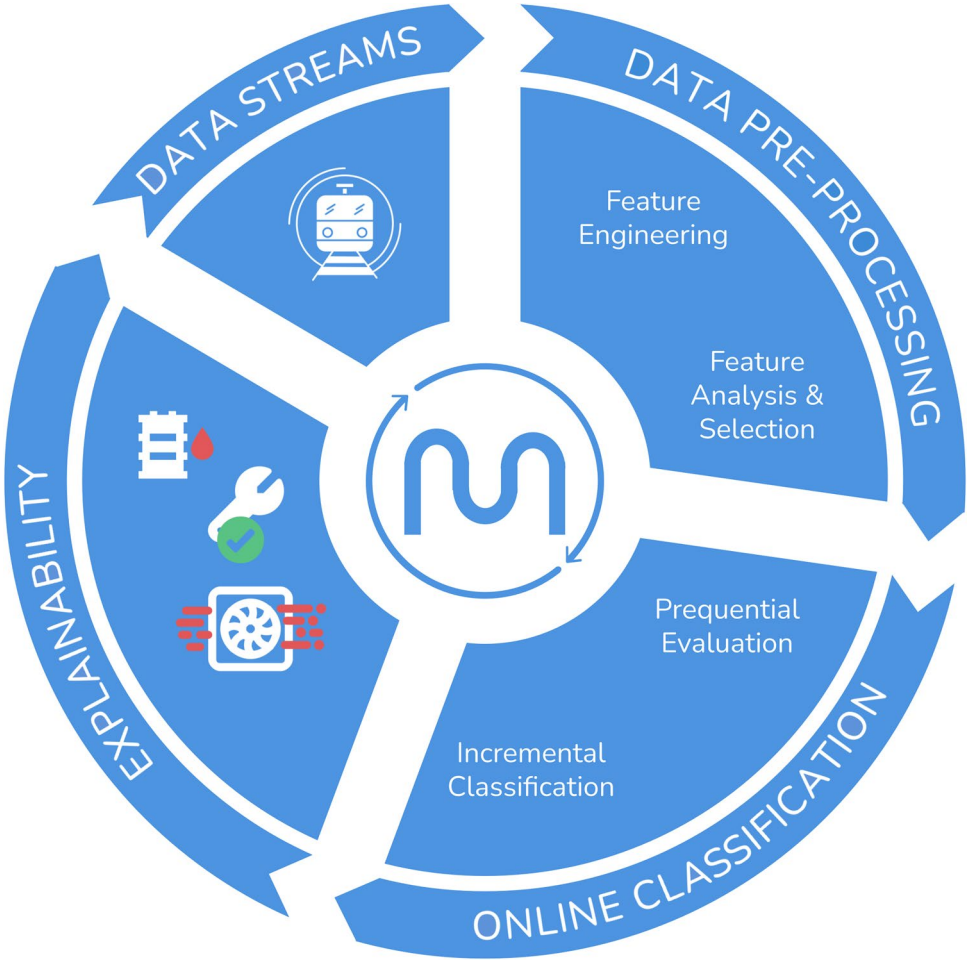

Fig. 1. Transparent online PdM cycle.

features characterize the signal distribution function, whereas the last detects anomalous frequencies, all within the sliding window of each FIR filter. This process is illustrated in Fig. 2.

$$
\begin{aligned}
X[n] &= \{x[n-w+1], \ldots, x[n]\}. \\
Y[n] &= \{y_0[n], y_1[n], \ldots, y_{w-1}[n]\} \mid \\
& y_0[n] \leq y_1[n] \leq \ldots \leq y_{w-1}[n], \\
& \text{where } \ \forall x \in X[n],\ x \in Y[n].
\end{aligned}
$$

$$
\begin{aligned}
avg^w[n] &= \frac{1}{w}\sum_{i=0}^{w} y_i[n] \\
std^w[n] &= \sigma(X[n]) \\
Q_1^w[n] &= y_{\lfloor \frac{1}{4}w \rceil}[n] \\
Q_2^w[n] &= y_{\lfloor \frac{2}{4}w \rceil}[n] \\
Q_3^w[n] &= y_{\lfloor \frac{3}{4}w \rceil}[n] \\
F^w[n] &= |FFT(X[n])|
\end{aligned}
\tag{1}
$$

Consequently, the proposed solution presents a robust configuration against errors or outliers. Each sliding window reduces signal noise at different frequencies and interpolates the absence of data. Moreover, quartiles in the sliding windows and the engineered features purge spurious sensor measurements.

Feature analysis and selection

In the case of stream-based ML classification, reducing the number of features by removing those less relevant to the task is essential. The chosen feature analysis and selection technique— variance thresholding—relies on

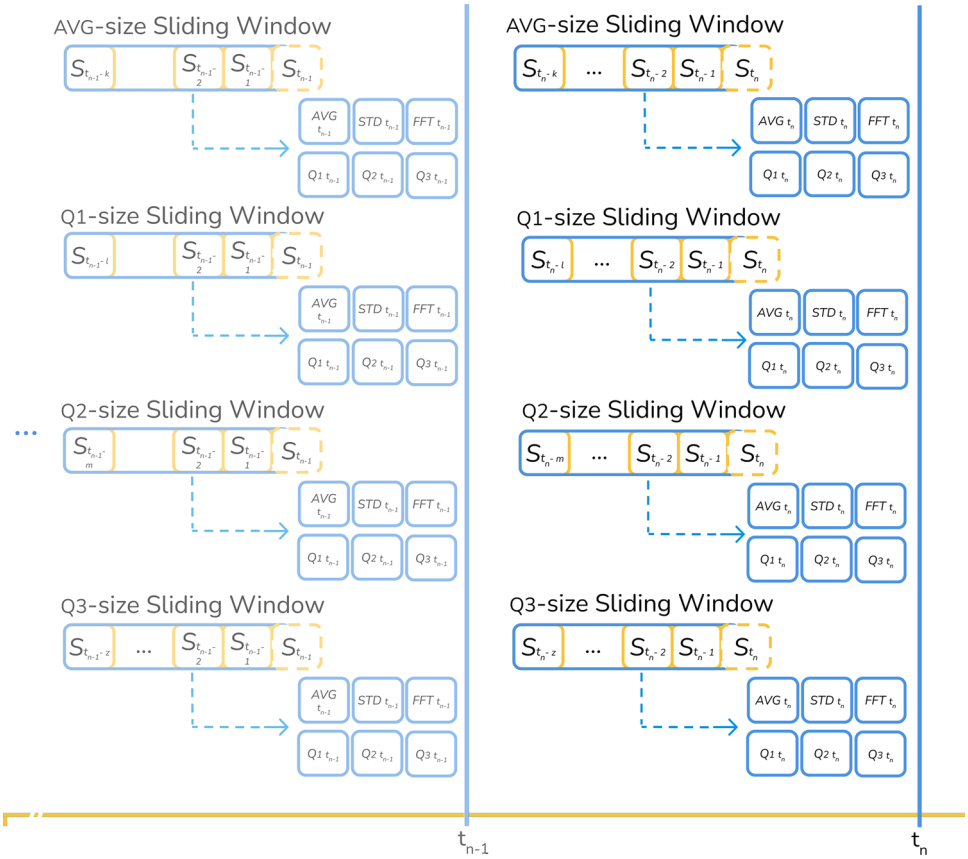

Fig. 2. Features engineered using the FFT and FIR filters.

feature variance. It assumes that features with low variance are less likely to be relevant than features with high variance[51,52]. This filtering technique first calculates the variance of each feature and then selects the features whose variance meets an experimentally defined threshold.

Online classification

The stream-based ML models were selected based on their interpretability and promising performance in similar classification problems[21,41].

- **Gaussian Naive Bayes** (GNB)[53] is a variant of the Naive Bayes (NB) model that processes streams by following a Gaussian normal distribution.
- **Hoeffding Tree Classifier** (HTC)[54] is a single tree induction model that exploits small amounts of samples to perform optimal splitting.
- **Hoeffding Adaptive Tree Classifier** (HATC)[55] is a single tree model that monitors branch performance to replace branches with decreasing performance by new and more accurate branches.
- **Adaptive Random Forest Classifier** (ARFC)[56] is an ensemble of tree models that induces diversity, using re-sampling and randomly selecting feature subsets for node splits and applies concept drift detectors per base tree. The classification outcome results from voting, where votes are weighted based on the test-then-train accuracy of the forest trees.

Based on sliding windows, the adopted feature engineering technique enables these models to create non-stationary knowledge spaces, where outdated information is forgotten, compared to existing methods (see Table 1). To optimize the results and save time, the original data set is downsampled by a factor of 500, and the resulting random subset is used to set the hyperparameters of the classifiers.

Algorithmic performance was evaluated using several standard classification metrics. First, accuracy was calculated, which reflects the proportion of correctly classified instances relative to the total number of samples analyzed. In addition, F-measure was employed in the macro-averaging and micro-averaging versions. While macro-averaging F-measure assigns equal weight to each class, allowing for the evaluation of model performance in situations of class imbalance, micro-averaging F-measure weights classes according to their frequency, thus providing a comprehensive view of performance based on the actual volume of each class. Finally, runtime data were included to quantify the computational efficiency of the models, a fundamental aspect in classification systems with continuous data flow. All these metrics were calculated using the prequential evaluation protocol[57], which involves evaluating the model sequentially as it receives new samples. This protocol first applies the

prediction to the current sample and subsequently uses that sample to update the model, thereby emulating the operation of an online learning system.

Online explainability

Natural language descriptions of the advent of a fault and its cause are highly relevant for the maintenance team. Thus, the information displayed has a dual purpose:

- Present the sensors that exhibit abnormal behavior textually and visually.
- Detail the most relevant features involved in the classification, including the most relevant FIR filters, using intelligible descriptions.

To extract this information, the decision path of the estimator (one estimator in the case of HTC and HATC and a configurable number of estimators in the case of the ARFC, see hyperparameter `model` in Listing 3) is traversed to gather the involved features that meet the *greater than* condition together with their frequency (number of occurrences). Then, they are ordered by decreasing frequency to extract the five most relevant. Furthermore, the ML model is also described through the selected relevant features: sliding windows, statistical parameters (average, standard deviation, Q_1, Q_2, Q_3), signal FFT and sensors (see Table 2). The generation of the sample classification and classification model descriptions are detailed in Algorithm 1 and Algorithm 2, respectively. Then, the most relevant features and parameters (windows, metrics, and sensors) are ordered by importance and introduced into templates (see Listing 4) to create natural language descriptions automatically. The latter data are also displayed on the user dashboard.

function FEATURES_INVOLVED_IN_ESTIMATOR_DECISION
  node_data= [{feature,threshold, left_branch, right_branch,type}] #List of nodes in the tree path.
  $sample\_data = [\{feature, feature(value)\}]$ #Input data sample.
  $feature\_relevance = []$ #Output feature relevance.
  $node = node\_data[0]$ #Starts with the root node.
  while $node[type] \neq leaf$ **do**
    $feature = node[feature]$
    $threshold = node[threshold]$
    $right\_branch = node[right\_branch]$
    $left\_branch = node[left\_branch]$
    if $individual\_data[feature(value)] \leq threshold$ **then**
      $node = left\_branch$
    else
      $feature\_relevance.append(feature])$
      $node = right\_branch$
    end if
  end while
  $feature\_relevance = feature\_relevance.sort\_by\_frequency()$ #Returns the features involved in the sample classification.
end function

Algorithm 1. Explanation of the sample classification.

Number	Type	Name	Description
1	Analog	DV pressure	Measures the compressor pressure when the air dryer towers discharge water.
2		Flowmeter	Measures the airflow circulating in the compressed air circuit.
3		H1	Detects when the pressure is above 10.2 bar.
4		MC	Measures the motor current.
5		Oil temperature	Measures the oil temperature in the compressor.
6		Reservoirs	Measures the pressure inside the compressed air tanks.
7		TP2	Measures the compressor pressure.
8		TP3	Measures the pneumatic panel pressure.
9	Digital	Flow rate	Activates when there is a change in the airflow rate.
10		COMP	Activates when there is no air admission in the compressor.
11		DV electric	Activates when the compressor is working under load.
12		LPS	Activates when the pressure is lower than 7 bar.
13		MGP	Activates when the pressure in the air-producing unit is below 8.2 bar.
14		Oil level	Activates when the oil level of the compressor is below the expected values.
15		Pressure switch	Activates when there is a discharge in the air drying towers.
16		Air dryer tower	Indicates the tower in operation (zero for tower one, one for tower two).

Table 2. Features in the transportation data set.

Class	Start	End
Air leak in the air dryer	28-02-22 21:53	01-03-22 02:00
Air leak in clients	23-03-22 14:54	23-03-22 15:24
Oil leak in the compressor	30-05-22 12:00	02-06-22 06:18

Table 3. Failure reports.

Class	Number of entries
Non-failure	372718
Oil leak compressor	202684
Air leak dryer	22021
Air leak client	9001
Total	606 424

Table 4. Distribution of classes in the experimental data set.

function DESCRIPTION_MOST_REPRESENTATIVE_PARAMETERS
$windows = windows.sort\_by\_frequency().get(5)$
$metrics = metrics.sort\_by\_frequency().get(5)$
$sensors = sensors.sort\_by\_frequency().get(5)$
end function

Algorithm 2. Explanation of the classification model.

Experimental results

This section describes the experimental data set (Section "Experimental data set") along with the feature engineering (Section "Feature engineering"), feature analysis and selection (Section "Feature analysis and selection"), classification (Section "Online classification") and explainability results (Section "Online explainability"). Ultimately, it discusses the results in comparison to those found in the literature (Section "Discussion").

For validation purposes, two experimental scenarios were designed. The first scenario exploits the average and standard deviation of the original features in Table 2, whereas the second also includes the quartile distribution and FFT of the original features obtained with the four FIR filters.

Experimental data set

The METROPT data set[58] was gathered at a frequency of 1 Hz from January to June 2022 by the metro operator in Porto, Portugal. The data comprise analog (*e.g.*, current, flow, pressure, and temperature) and digital (*e.g.*, control and status) signals related to the air-producing unit. Table 2 details the features in the METROPT data set. Each data sample comprises values for the 16 features listed. Table 3 lists the faults within the data set.

Only samples from the day before to the day after failures occur are used to ensure rapid evaluation. Finally, samples from the two hours preceding the failure were tagged as failures to test whether the proposed approach effectively predicts failures in advance. The resulting data set holds 606424 samples distributed as indicated in Table 4.

Online data pre-processing

This section details the implementations, design decisions, and results of the feature engineering, feature analysis, and selection stages.

Feature engineering

First, the minimal number of samples (sliding window length) of each FIR filter is determined using the first two days of data. The result was 1399, 116, 531 and 864 samples for the average, Q_1, Q_2 and Q_3 FIR filters, respectively. Next, the system generates, for each data sample, 24 new features corresponding to the average, standard deviation, Q_1, Q_2, Q_3 and FFT features of the four FIR filters. The total number of features amounts to 400: 16 original features listed in Table 2 plus 384 engineered features.

Feature analysis and selection

Feature selection evaluates the cumulative variance of each new feature sample using the `VarianceThreshold` technique (available at https://riverml.xyz/0.11.1/api/feature-selection/VarianceThreshold, June 2025) from `River` package (available at https://riverml.xyz/0.11.1, June 2025). The variance threshold was set to 0.5. The

selected features per scenario are listed below. For conciseness, the variance values of the selected features are not shown.

Scenario 1: the features engineered from original features 2 to 5, 7, and 8 of Table 2, corresponding to two new features (average and standard deviation) per selected original feature.
Scenario 2: the features engineered from original features 1 to 13, and 16 of Table 2, corresponding to 24 new features (average, standard deviation, Q_1, Q_2, Q_3 and FFT of the four FIR filters) per selected original feature.

In summary, feature selection in both scenarios was carried out using a variance threshold experimentally set at 0.5. This threshold was established after observing signal variations, concluding that a variation below this value can be considered attributable to noise or common sensor behavior, with greater variation being necessary to provide information to the classifier. While signals with lower variations could introduce bias into the prediction, increasing the risk of overfitting in the models, this approach eliminates variables whose variability is insufficient to contribute significantly to classification, reducing dimensionality and promoting model generalization and efficiency. As a result, 12 and 336 features were used in the first and second scenarios, respectively.

Online classification

The stream-based classification explores the following algorithmic implementations:

- GNB (available at https://riverml.xyz/dev/api/naive-bayes/GaussianNB, June 2025).
- HTC (available at https://riverml.xyz/dev/api/tree/HoeffdingTreeClassifier, June 2025).
- HATC (available at https://riverml.xyz/0.13.0/api/tree/HoeffdingAdaptiveTreeClassifier, June 2025).
- ARFC (available at https://riverml.xyz/dev/api/ensemble/AdaptiveRandomForestClassifier, June 2025).

Listings 1, 2, and 3 present the hyperparameter values explored for the HTC, HATC, and ARFC models, respectively. It should be noted that the GNB model does not require hyperparameter configuration since it assumes a Gaussian distribution for each feature and estimates the parameters directly from the observed data. The list of values was selected by searching for a parameter range that would guarantee stable system operation under controlled conditions without increasing server resource consumption. Thus, this exhaustive hyperparameter search process enables each model to be optimally adapted to the monitoring problem, thereby maximizing accuracy while maintaining stability and runtime efficiency. In each case, the values selected as optimal are highlighted in bold.

The applied hyperparameter optimization procedure constitutes an adaptation of the traditional *Grid Search* method (available at https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html, June 2025), widely used in offline supervised learning. However, since the problem addressed is framed in a streaming classification context, it is not feasible to employ conventional cross-validation since the samples arrive sequentially and cannot be considered independent or interchangeable in time. Therefore, the *Grid Search* scheme was adapted to a *prequential* evaluation framework. This approach enables the system to maintain its online nature, preserving both the temporal integrity of the data and the continuous learning dynamics.

In the case of HTC, the tree hyperparameters *depth*, *tie-threshold*, and *max-size* were explored. The best performance was obtained with a maximum depth of 50 nodes, which allows an adequate balance between representation capacity and the risk of overfitting in a continuous learning context, where the number of instances can grow indefinitely. A *tie-threshold* value of 0.5 imposes a relatively conservative margin for decision-making when increases in the gain metric are inconclusive, helping to stabilize tree growth in the advent of minor fluctuations in the data. Finally, a *max-size* of 50 controls the maximum number of memory buffers temporarily stored in each node before performing splits, favoring a sufficiently agile adaptation without resulting in high computational cost.

For HATC, the same ranges of *depth* and *tie-threshold* were used, selecting the optimal values of 50 and 0.5, respectively. However, the *max-size* parameter showed better performance with a value of 100, allowing the model to accumulate more information locally before deciding on structural modifications, which is consistent with the adaptive nature of this algorithm, which seeks to replace underperforming branches dynamically.

For its part, ARFC incorporates multiple hyperparameters related to both the ensemble size and the diversification of the base models. The optimal number of trees (*models*) was set to 50, providing a compromise between diversity and computational cost. The *features* parameter also achieved its best performance with 50 features, allowing for an adequate degree of randomization without sacrificing relevant information in each partition. Finally, the *lambda* parameter was set to 50, adjusting the bagging intensity to enhance the internal variability of the ensemble.

```
depth = [50, 100, 200]
tiethreshold = [0.5, 0.05, 0.005]
maxsize = [50, 100, 200]
```

Listing 1. HTC hyperparameter configuration (best values in bold).

```
depth = [50, 100, 200]
tiethreshold = [0.5, 0.05, 0.005]
maxsize = [50, 100, 200]
```

Listing 2. HATC hyperparameter configuration (best values in bold).

```
models = [50, 100, 200]
features = [50, 100, 200]
lambda = [50, 100, 200]
```

Listing 3. ARFC hyperparameter configuration (best values in bold).

Table 5 provides the classification results obtained in the two experimental scenarios (downsampled by a factor of 50 to reduce over-training with similar samples), with the best results highlighted in boldface. Both scenarios incorporate feature selection and hyperparameter optimization.

In the first scenario, the results reflect limitations in the classifiers' ability to capture the particularities of minority classes. As a consequence, models such as GNB, HTC, and HATC experience a notable drop in performance, as they are not able to characterize more complex aspects of the dynamic behavior of the signals captured by the sensors, such as changes in distribution, the presence of anomalous oscillations or frequency alterations typical of certain types of failures. Even so, the ARFC model manages to maintain the best relative performance within this scenario, reaching an accuracy of 97.53 %, although with room for improvement in class #4 (air leak client).

In scenario 2, the new features allow the system to capture more complex statistical patterns, anomalies, and frequency components that are not accessible through simple mean and standard deviation measures. Thanks to this richer and more descriptive representation, the models experience a substantial and homogeneous improvement in all the metrics evaluated, with the ARFC model reaching its maximum performance with an accuracy of 99.62 % and F-measure values greater than 98 % in all classes.

The better performance of ARFC is justified by the fact that it is an ensemble of multiple adaptive trees. The ARFC has a greater capacity to model nonlinear and complex relationships between features, something especially relevant in a sensor-based system where interference and operation are nonlinear. Furthermore, ARFC presents explainability capabilities that are especially valuable in PDM applications. Since it is based on decision trees, it is possible to inspect individual decision nodes and the features most frequently involved in classification paths. This traceability is much more limited in other models, such as GNB, where decisions are based on global probability distributions.

Moreover, ARFC classifies 58 samples per second. Given that the air production unit on board the Metro do Porto trains generates one sample per second, the online classifier performs more than adequately. Furthermore, given the need to sequentially test, train, and evaluate each sample received, the experiments were carried out on a single core of a 20-core platform. In a real deployment, where threads can be assigned to different ML models, the system throughput increases from 58 samples per second (one thread) to 1160 samples per second (20 threads). This scalability potential is essential for modeling complex processes involving multiple subsystems, such as railway networks. Decentralized performance can be addressed by distributing the workload across multiple computing nodes.

Online explainability

As previously mentioned, the method explains classification outcomes for both ML and maintenance experts. The ML expert explanation details the most relevant features involved in the prediction and the decision paths of each target class (local explanations) along with the overall decision path of the model (global explanations), using natural language (see Listing 4).

The maintenance expert appreciates a clear, natural language explanation and a visual dashboard (see Fig. 3) that highlights the most relevant features in predicting each target class. The dashboard displays the most relevant features' value, status, and normal and abnormal plots (visual and feature-relevance explanations). The

Scenario	Model	Accuracy	*F*-measure					Time (s)
			Macro	#1	#2	#3	#4	
1	GNB	92.19	86.16	93.41	91.67	85.01	**74.55**	4.95
	HTC	77.80	64.62	83.61	70.49	53.45	50.95	6.57
	HATC	94.80	77.72	95.79	96.90	73.33	44.86	5.16
	ARFC	**97.53**	**88.07**	**98.00**	**98.10**	**95.11**	61.07	135.97
2	GNB	76.15	62.62	82.54	60.90	82.33	24.69	15.36
	HTC	71.46	73.18	79.33	47.14	89.08	77.15	15.30
	HATC	95.89	90.90	96.59	96.08	88.46	82.48	21.12
	ARFC	**99.62**	**98.90**	**99.69**	**99.72**	**98.17**	**98.03**	209.65

Table 5. Online failure detection results (best values in bold, #1: non-failure, #2: oil leak, #3: air leak dryer, #4: air leak client).

Fig. 3. Explainability dashboard.

TEMPLATED (available at https://templated.co, June 2025) library was employed to display the most relevant features and the Highcharts (available at https://www.highcharts.com, June 2025) for the plots.

```
For sample 1340, the five most representative features are:
   1. FFT of Reservoirs from Q1-size sliding window
   2. FFT of Reservoirs from Q2-size sliding window.
   3. FFT TP3 from Q3-size sliding window.
   4. FFT of DV pressure from AVG-size sliding window.
   5. FFT of Flowmeter from Q1-sliding window.

The most representative parameters for the ML model are:
   Sliding windows: four sliding windows contribute equally.
   Signal Pre-Processing Technique: FFT.
   Sensors: Flowmeter, H1, Oil temperature, TP2, TP3.

Given sensors with
   abnormal patterns:
   - DV pressure sensor (> 30 minutes with significant change)
   - TP3 (> 15 minutes)
   - Reservoirs (> 10 minutes)
and anomalous values:
   - Flowmeter (> 2 minutes)
   - Reservoirs (> 2 minutes)
then the prediction is that
   there is an air leak in the air dryer.
```

Listing 4. Natural language explanation of a sample classification.

Discussion

It is crucial to elaborate on the practical implications of the results obtained in this study for implementing PDM solutions in real-world railway environments. Accordingly, its high performance ensures a robust ability to anticipate critical events, minimizing both false positives and undetected failures. This allows railway operators to significantly reduce unplanned downtime, optimize maintenance planning, and extend component lifespans, resulting in a direct improvement in operational efficiency and a decrease in associated costs. Similarly, the real-time processing of data streams ensures that the solution is compatible with the technical and logistical

Proposal	ML Technique	Accuracy	*F*-measure	Time
[21]	Autoencoder	NA	33.60	NA
[23]	XStream	97.49	83.05	NA
Current	ARFC	99.62	98.90	209.65

Table 6. Comparison of online failure detection results using MetroPT.

requirements of the railway environment. In this regard, consideration should be given to the system's ability to classify samples in real time (*i.e.*, 58 samples per second), which demonstrates its feasibility for deployment, even in contexts with multiple subsystems generating data simultaneously. Furthermore, the automatic generation of natural language and visual explanations provides a key differential value from an operational standpoint. This functionality enables maintenance operators to understand the reasons behind a failure prediction, facilitating informed decision-making and reducing diagnosis and resolution times. The latter also increases confidence in the system. Ultimately, the use of a real-world dataset, such as MetroPT, with verified operational failure labels, supports the relevance and applicability of the results. This demonstrates that the proposed solution is not limited to experimental conditions but has been validated in an industrial context with complex, heterogeneous data.

Since the surveyed fault detection/prediction methods use different datasets, comparing the results may not be straightforward. Fumeo et al.[40] employed an OSVR to estimate the remaining useful life of components exploring prognostic health monitoring challenge data sets. The solution presents a Mean Absolute Percentage Error (MAPE) of 2.57 %. Although decision-making is done online, the authors do not adopt classification techniques. Moreover, the fault detection solution by Ribeiro et al.[41] processes data online while the classification is performed offline with a Sparse Autoencoder. The evaluation metric used is Root Mean Squared Error (RMSE), and the best value obtained is 0.3048. Compared to Fumeo et al.[40] and Ribeiro et al.[41], the current solution attains better values (*i.e.*, 0.39 % MAPE and 0.0623 RMSE for scenario 2 with failure and non-failure categories). In contrast, Meira et al.[23] proposed a similar solution but without explainability capabilities and lower performance (*e.g.*, 16 % points lower for F-measure).

The only work that explores the MetroPT data set is that of Davari et al.[21]. The current solution increases the classification *F*-measure by 194 %. The performance of the proposed solution and the most closely related work from the literature are summarized in Table 6.

Regarding explainability, the current explanations can be compared with those obtained by processing the MetroPT data set with the neural symbolic explainer technique described by Ribeiro et al.[41]. Listing 5 holds the explanatory rules of the detected air leak on the train APU where B*i* indicates the binary representation of analog sensor values and RE the autoencoder reconstruction error value. While these explanatory rules constitute a post hoc technique that provides local textual feature-relevance explanations, they are difficult to understand. The current proposal explains fault predictions through global textual feature-relevance explanations and local textual, visual, and feature-relevance descriptions, providing meaningful insights to maintenance experts.

```
Rule 1: if (B1_TP3 > 7345 and B5_MC > 1925) then RE = 1.8116
Rule 2: if (Flowrate > 251) then RE = 2.3932
Rule 3: if (B6_TP3 < 5635) then RE = 2.4445
Rule 4: if (B2_H1 > 378) then RE = 1.8791
```

Listing 5. Rule-based explanations.

More specifically, our decision path traversing algorithm generates, utilizing natural language models and textual descriptions for the control panel, accompanied by visual elements. Moreover, it analyses the causal pattern of the failure considering the most relevant characteristics (*i.e.*, those whose value exceeds the bifurcation threshold for the failure category) in the decision tree path of the ARFC model. In this regard, the oil leak compressor failure is related to high values in the flowmeter (feature 2 in Table 2), the H1 (feature 3), and the broader wave of the MC (feature 4). In contrast, to detect the air leak dryer failure, high values in the oil temperature (feature 5) and the TP2 (feature 7), and the broader wave of the reservoirs (feature 6) must be considered. Regarding the air leak client failure causal pattern, high oil temperature, and broader wave values for this feature and the TP3 (feature 8) was detected.

Consequently, the explainability of the model enables effective information exchange between humans (*e.g.*, experts, and end-users) and complex, intelligent systems. The railway sector's harsh and dynamic operating conditions make fault prediction a crucial challenge. Specifically, it can significantly reduce the time needed for fault cause analysis by minimizing fault diagnosis tests. The adopted analysis of the most relevant features is well-suited for explaining and troubleshooting predictions. Furthermore, thanks to the generated descriptions, the experts can provide feedback to correct, refine, or even identify additional fault causes. Early failure prediction and explanation improve user experience by reducing service outages and delays.

Conclusion

This innovative real-time fault prediction provides natural language and visual explainability for ITS. The ultimate objective of this research is to mine the incoming train sensor data from a public railway operator to anticipate failures and support maintenance decisions.

Thus, the proposed solution detects and explains real-time failures based on the incoming data. The processing pipeline performs: (*i*) online data pre-processing (feature engineering, analysis, and selection), (*ii*) online sample classification, and (*iii*) online sample classification explanation. Consequently, this research advances the field by proposing a transparent and real-time PdM solution that incorporates novel features (statistical and frequency-related features) and adopts sliding-window-based processing to create non-stationary online ML models. Moreover, it advances explainable Artificial Intelligence research by presenting comprehensive natural language descriptions and visual indicators in a dashboard for maintenance experts and non-expert users, all in real time.

The experiments were performed with the MetroPT data set, which holds analog and digital data produced by the air-producing unit on board the trains of Metro do Porto. The best results, obtained with the Adaptive Random Forest Classifier, show accuracy as well as macro and micro *F*-measure above 98 %. Finally, the method explains classification outcomes to help decision-makers understand the underlying problem and trigger maintenance actions.

Therefore, early detection and explanation of potential failures contribute to enhancing the quality of passenger service and the operator's reputation. Notably, the proposed PdM pipeline accelerates fault detection, troubleshooting, and repair, thereby reducing costs, extending equipment life, and enhancing service quality. The primary weakness of the method lies in the configuration of the process parameters and model hyperparameters. The definition of the window length of the FIR filters and the sampling reduction factor used in the feature analysis and selection phase are determined dynamically through iterative adjustment and scoring methods. Due to the resource and time-consuming nature of online hyperparameter optimization, model hyperparameters are set at the beginning of the classification phase. These limitations can be overcome by updating process parameters (periodically) and model hyperparameters (whenever data drifts). Finally, the current fault detection, which works at the individual sensor level, can be more holistic, *e.g.*, taking into account the propagation of anomalies between neighboring sensors. To facilitate future research, additional datasets and language models will be considered to validate the method's applicability and generate more engaging natural language descriptions. Ultimately, in future experiments, we will gather quantitative data related to productivity improvements from the operator to validate our solution's positive impact further.

Data availability

The data is publicly available in the work by Veloso et al.[58].

Received: 16 May 2024; Accepted: 17 June 2025
Published online: 28 July 2025

References

1. Sony, S., Laventure, S. & Sadhu, A. A literature review of next-generation smart sensing technology in structural health monitoring. *Struct. Control. Health Monit.* **26**, 2321–2343. https://doi.org/10.1002/stc.2321 (2019).
2. Oztemel, E. & Gursev, S. Literature review of Industry 4.0 and related technologies. *J. Intell. Manuf.* **31**, 127–182. https://doi.org/10.1007/s10845-018-1433-8 (2020).
3. Jiang, Y., Yin, S., Dong, J. & Kaynak, O. A review on soft sensors for monitoring, control, and optimization of industrial processes. *IEEE Sens. J.* **21**, 12868–12881. https://doi.org/10.1109/JSEN.2020.3033153 (2021).
4. Li, P., Zhu, Y., Liu, Y. & Jin, C. Research on system architecture, basic platform, and development path of autonomous intelligent high-speed railway system. *IEEE Intell. Transp. Syst. Mag.* **1**, 2–13. https://doi.org/10.1109/MITS.2023.3305200 (2023).
5. Prakash, J., Murali, L., Manikandan, N., Nagaprasad, N. & Ramaswamy, K. A vehicular network based intelligent transport system for smart cities using machine learning algorithms. *Sci. Rep.* **14**, 468–483. https://doi.org/10.1038/s41598-023-50906-7 (2024).
6. Kolajo, T., Daramola, O. & Adebiyi, A. Big data stream analysis: A systematic literature review. *J. Big Data* **6**, 1–30. https://doi.org/10.1186/s40537-019-0210-7 (2019).
7. Vial, A., Daamen, W., Ding, A. Y., van Arem, B. & Hoogendoorn, S. AMSense: How mobile sensing platforms capture pedestrian/cyclist spatiotemporal properties in cities. *IEEE Intell. Transp. Syst. Mag.* **14**, 29–43. https://doi.org/10.1109/MITS.2019.2953509 (2022).
8. Wang, M., Zhou, D., Chen, M. & Wang, Y. Anomaly detection in the fan system of a thermal power plant monitored by continuous and two-valued variables. *Control. Eng. Pract.* **102**, 104522–104531. https://doi.org/10.1016/j.conengprac.2020.104522 (2020).
9. Zonta, T. et al. Predictive maintenance in the Industry 4.0: A systematic literature review. *Comput. Ind. Eng.* **150**, 106889–106906. https://doi.org/10.1016/j.cie.2020.106889 (2020).
10. Sahal, R., Breslin, J. G. & Ali, M. I. Big data and stream processing platforms for Industry 4.0 requirements mapping for a predictive maintenance use case. *J. Manuf. Syst.* **54**, 138–151. https://doi.org/10.1016/j.jmsy.2019.11.004 (2020).
11. Dalzochio, J. et al. Machine learning and reasoning for predictive maintenance in Industry 4.0: Current status and challenges. *Comput. Ind.* **123**, 103298–103313. https://doi.org/10.1016/j.compind.2020.103298 (2020).
12. Irannezhad, E. Is blockchain a solution for logistics and freight transportation problems?. *Transp. Res. Procedia* **48**, 290–306. https://doi.org/10.1016/j.trpro.2020.08.023 (2020).
13. Marzouk, M. & Elsayed, A. Framework for assessing bridges construction impact on work zone traffic using Br IM. *Sci. Rep.* **14**, 83–101. https://doi.org/10.1038/s41598-023-50404-w (2024).
14. Neilson, A., Indratmo, Daniel, B. & Tjandra, S. Systematic review of the literature on big data in the transportation domain: Concepts and applications. *Big Data Res.* **17**, 35–44. https://doi.org/10.1016/j.bdr.2019.03.001 (2019).
15. Zargayouna, M., Othman, A., Scemama, G. & Zeddini, B. Multiagent simulation of real-time passenger information on transit networks. *IEEE Intell. Transp. Syst. Mag.* **12**, 50–63. https://doi.org/10.1109/MITS.2018.2879166 (2020).
16. Burkart, N. & Huber, M. F. A survey on the explainability of supervised machine learning. *J. Artif. Intell. Res.* **70**, 245–317. https://doi.org/10.1613/jair.1.12228 (2021).
17. Vollert, S., Atzmueller, M. & Theissler, A. Interpretable machine learning: A brief survey from the predictive maintenance perspective. In *Proceedings of the International Conference on Emerging Technologies and Factory Automation*, 1–8. https://doi.org/10.1109/ETFA45728.2021.9613467 (IEEE, 2021).

18. Adadi, A. & Berrada, M. Peeking inside the black-box: A survey on explainable artificial intelligence (XAI). *IEEE Access* **6**, 52138–52160. https://doi.org/10.1109/ACCESS.2018.2870052 (2018).
19. Spadon, G., de Carvalho, A. C. P. L. F., Rodrigues-Jr, J. F. & Alves, L. G. A. Reconstructing commuters network using machine learning and urban indicators. *Sci. Rep.* **9**, 11801–11813. https://doi.org/10.1038/s41598-019-48295-x (2019).
20. Binder, M., Mezhuyev, V. & Tschandl, M. Predictive maintenance for railway domain: A systematic literature review. *IEEE Eng. Manag. Rev.* **51**, 120–140. https://doi.org/10.1109/EMR.2023.3262282 (2023).
21. Davari, N., Veloso, B., Ribeiro, R. P., Pereira, P. M. & Gama, J. Predictive maintenance based on anomaly detection using deep learning for air production unit in the railway industry. In *Proceedings of the IEEE International Conference on Data Science and Advanced Analytics*, 1–10 (IEEE, 2021). https://doi.org/10.1109/DSAA53316.2021.9564181.
22. Hnia, S., Flodin, J., Casanueva, C., Asplund, M. & Stichel, S. Predictive maintenance in railway systems: MBS-based wheel and rail life prediction exemplified for the Swedish Iron-Ore line. *Veh. Syst. Dyn.* **62**, 3–20. https://doi.org/10.1080/00423114.2022.2161920 (2024).
23. Meira, J. et al. Data-driven predictive maintenance framework for railway systems. *Intell. Data Anal.* **27**, 1087–1102. https://doi.org/10.3233/IDA-226811 (2023).
24. Le-Nguyen, M.-H., Turgis, F., Fayemi, P.-E. & Bifet, A. A complete streaming pipeline for real-time monitoring and predictive maintenance. In *Proceedings of the 31st European Safety and Reliability Conference*, 2112–2119, (Research Publishing Services, 2021). https://doi.org/10.3850/978-981-18-2016-8_400-cd.
25. Le-Nguyen, M.-H., Turgis, F., Fayemi, P.-E. & Bifet, A. Real-time learning for real-time data: Online machine learning for predictive maintenance of railway systems. *Transp. Res. Procedia* **72**, 171–178. https://doi.org/10.1016/j.trpro.2023.11.391 (2023).
26. Le-Nguyen, M.-H., Turgis, F., Fayemi, P.-E. & Bifet, A. Exploring the potentials of online machine learning for predictive maintenance: A case study in the railway industry. *Appl. Intell.* **53**, 29758–29780. https://doi.org/10.1007/s10489-023-05092-4 (2023).
27. Molnar, C., Casalicchio, G. & Bischl, B. Interpretable machine learning–A brief history, state-of-the-art and challenges. *Commun. Comput. Inf. Sci.* **1323**, 417–431. https://doi.org/10.1007/978-3-030-65965-3_28 (2020).
28. Pang, Z. et al. A survey of decision-making safety assessment methods for autonomous vehicles. *IEEE Intell. Transp. Syst. Mag.* https://doi.org/10.1109/MITS.2023.3292511 (2023).
29. Søgaard, A. Explainable natural language processing. In *Synthesis Lectures on Human Language Technologies*, 1–123, (Springer, 2021). https://doi.org/10.2200/S01118ED1V01Y202107HLT051.
30. Overview, An., Lampropoulos, G., Siakas, K. & Anastasiadis, T. Internet of things in the context of Industry 4.0. *Int. J. Entrep. Knowl.* **7**, 4–19. https://doi.org/10.2478/ijek-2019-0001 (2019).
31. Katsaros, K. V. et al. Connected and automated mobility services in 5G cross-border environments: Challenges and prospects. *IEEE Intell. Transp. Syst. Mag.* **15**, 145–157. https://doi.org/10.1109/MITS.2023.3237271 (2023).
32. Carvalho, T. P. et al. A systematic literature review of machine learning methods applied to predictive maintenance. *Comput. Ind. Eng.* **137**, 106024–106034. https://doi.org/10.1016/j.cie.2019.106024 (2019).
33. Proto, S. et al. REDTag: A predictive maintenance framework for parcel delivery services. *IEEE Access* **8**, 14953–14964. https://doi.org/10.1109/ACCESS.2020.2966568 (2020).
34. Bekar, E. T., Nyqvist, P. & Skoogh, A. An intelligent approach for data pre-processing and analysis in predictive maintenance with an industrial case study. *Adv. Mech. Eng.* **12**, 1–14. https://doi.org/10.1177/1687814020919207 (2020).
35. Xie, J., Huang, J., Zeng, C., Jiang, S.-H. & Podlich, N. Systematic literature review on data-driven models for predictive maintenance of railway track: Implications in geotechnical engineering. *Geosciences* **10**, 425–449. https://doi.org/10.3390/geosciences10110425 (2020).
36. Davari, N. et al. A survey on data-driven predictive maintenance for the railway industry. *Sensors* **21**, 5739–5751. https://doi.org/10.3390/s21175739 (2021).
37. Gama, J., Ribeiro, R. P. & Veloso, B. Data-driven predictive maintenance. *IEEE Intell. Syst.* **37**, 27–29. https://doi.org/10.1109/MIS.2022.3167561 (2022).
38. Ramírez-Gallego, S., Krawczyk, B., García, S., Woźniak, M. & Herrera, F. A survey on data preprocessing for data stream mining: Current status and future directions. *Neurocomputing* **239**, 39–57. https://doi.org/10.1016/j.neucom.2017.01.078 (2017).
39. Manco, G. et al. Fault detection and explanation through big data analysis on sensor streams. *Expert Syst. Appl.* **87**, 141–156. https://doi.org/10.1016/j.eswa.2017.05.079 (2017).
40. Fumeo, E., Oneto, L. & Anguita, D. Condition based maintenance in railway transportation systems based on big data streaming analysis. *Procedia Comput. Sci.* **53**, 437–446. https://doi.org/10.1016/j.procs.2015.07.321 (2015).
41. Ribeiro, R. P. et al. Online anomaly explanation: A case study on predictive maintenance. In *Proceedings of the Communications in Computer and Information Science*, Vol. 1753, 383–399 (Springer, 2023). https://doi.org/10.1007/978-3-031-23633-4_25.
42. Pazera, M., Mrugalski, M., Witczak, M. & Buciakowski, M. A Neural network-based approach to sensor and actuator fault-tolerant control. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Vol. 11507, 515–526 (Springer, 2019). https://doi.org/10.1007/978-3-030-20518-8_43.
43. Longo, L., Goebel, R., Lecue, F., Kieseberg, P. & Holzinger, A. Explainable artificial intelligence: Concepts, applications, research challenges and visions. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Vol. 12279, 1–16 (Springer, 2020). https://doi.org/10.1007/978-3-030-57321-8_1.
44. Ribeiro, M. T., Singh, S. & Guestrin, C. Why should I trust you explaining the predictions of any classifier. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1135–1144 (Association for Computing Machinery, 2016). https://doi.org/10.1145/2939672.2939778.
45. Lundberg, S. M. & Lee, S.-I. A unified approach to interpreting model predictions. In *Proceedings of the International Conference on Neural Information Processing Systems*, Vol. 30, 4768–4777 (Curran Associates Inc., 2017). https://doi.org/10.5555/3295222.3295230.
46. Allah Bukhsh, Z., Saeed, A., Stipanovic, I. & Doree, A. G. Predictive maintenance using tree-based classification techniques: A case of railway switches. *Transp. Res. Part C Emerg. Technol.* **101**, 35–54. https://doi.org/10.1016/j.trc.2019.02.001 (2019).
47. Niu, X., Jiang, C., Gao, J., Korniss, G. & Szymanski, B. K. From data to complex network control of airline flight delays. *Sci. Rep.* **11**, 18715. https://doi.org/10.1038/s41598-021-98112-7 (2021).
48. Plonka, G., Potts, D., Steidl, G. & Tasche, M. Fast Fourier transforms. In *Applied and Numerical Harmonic Analysis*, 231–303 (Springer, 2018). https://doi.org/10.1007/978-3-030-04306-3_5.
49. Segovia, P. et al. Sliding window assessment for sensor fault model-based diagnosis in inland waterways. *IFAC-PapersOnLine* **51**, 31–36. https://doi.org/10.1016/j.ifacol.2018.06.195 (2018).
50. Arumugam, N. & Paramasivan, B. A novel microprogrammed reconfigurable parallel VHBCSE based FIR filter for wireless sensor nodes. *Wirel. Pers. Commun.* **115**, 2197–2210. https://doi.org/10.1007/s11277-020-07677-5 (2020).
51. Cao, L., Yu, F., Yang, F., Cao, Y. & Gopaluni, R. B. Data-driven dynamic inferential sensors based on causality analysis. *Control. Eng. Pract.* **104**, 104626–104641. https://doi.org/10.1016/j.conengprac.2020.104626 (2020).
52. Treistman, A., Mughaz, D., Stulman, A. & Dvir, A. Word embedding dimensionality reduction using dynamic variance thresholding (DyVaT). *Expert Syst. Appl.* **208**, 118157–118171. https://doi.org/10.1016/j.eswa.2022.118157 (2022).
53. Xue, Q., Zhu, Y. & Wang, J. Joint distribution estimation and Naïve Bayes classification under local differential privacy. *IEEE Trans. Emerg. Top. Comput.* **9**, 2053–2063. https://doi.org/10.1109/TETC.2019.2959581 (2021).

54. Pham, X. C. et al. Learning from data stream based on random projection and Hoeffding tree classifier. In *Proceedings of the International Conference on Digital Image Computing: Techniques and Applications*, 1–8 (IEEE, 2017). https://doi.org/10.1109/DICTA.2017.8227456.
55. Stirling, M., Koh, Y. S., Fournier-Viger, P. & Ravana, S. D. Concept drift detector selection for Hoeffding adaptive trees. In *Proceedings of the Australasian Joint Conference on Artificial Intelligence*, Vol. 11320, 730–736 (Springer, 2018). https://doi.org/10.1007/978-3-030-03991-2_65.
56. Gomes, H. M. et al. Adaptive random forests for evolving data stream classification. *Mach. Learn.* **106**, 1469–1495. https://doi.org/10.1007/s10994-017-5642-8 (2017).
57. Gama, J., Sebastião, R. & Rodrigues, P. P. On evaluating stream learning algorithms. *Mach. Learn.* **90**, 317–346. https://doi.org/10.1007/s10994-012-5320-9 (2013).
58. Veloso, B., Ribeiro, R. P., Gama, J. & Pereira, P. M. The MetroPT dataset for predictive maintenance. *Sci. Data* **9**, 764–771. https://doi.org/10.1038/s41597-022-01877-3 (2022).

Acknowledgements

This work was partially supported by: (*i*) Xunta de Galicia grants ED481B-2022-093 and ED481D 2024/014, Spain; and (*ii*) Portuguese national funds through FCT—Fundação para a Ciência e a Tecnologia (Portuguese Foundation for Science and Technology)— as part of project UIDP/50014/2020 (DOI: 10.54499/UIDP/50014/2020).

Author contributions

S.G.M.: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Resources, Data Curation, Writing - Original Draft, Writing - Review & Editing, Visualization, Funding acquisition. F.A.P.: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Resources, Data Curation, Writing - Original Draft, Writing - Review & Editing, Visualization, Funding acquisition. F.L.: Conceptualization, Writing - Original Draft, Writing - Review & Editing. B.B.: Resources, Writing - Review & Editing. B.M.: Conceptualizations, Writing - Review & Editing, Supervision, Funding acquisition. J.C.B.R: Conceptualizations, Writing - Review & Editing.

Declarations

Competing interests

The authors declare no competing interests.

Additional information

Correspondence and requests for materials should be addressed to S.G.-M.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.