Decision Analytics Journal 16 (2025) 100607

Contents lists available at ScienceDirect

Decision Analytics Journal

journal homepage: www.elsevier.com/locate/dajour

A predictive analytics framework for early detection of production halts and quality issues

Arthur Matta [a,b,*], Luís Miguel Matos [b], Jorge Miguel Silva [c], Miguel Bastos Gomes [c], André Pilastri [a], Paulo Cortez [a,b,*]

[a] CCG/ZGDV ICT Innovation Institute, Universidade do Minho, Campus de Azurém, ed. 14, Guimarães, 4800-058, Braga, Portugal
[b] ALGORITMI/LASI Center – Dep. Information Systems, Escola de Engenharia, Universidade do Minho, Campus de Azurém, Guimarães, 4800-058, Braga, Portugal
[c] SONAE ARAUCO Portugal, S.A., Quinta da Paça – S. Paio de Gramaços, Apartado 73, Oliveira do Hospital, 3400-691, Coimbra, Portugal

ARTICLE INFO

Keywords:
Explainable Artificial Intelligence
Machine Learning
Quality control
Industry 4.0
Predictive modeling
Process optimization

ABSTRACT

This study presents a Machine Learning (ML) framework for an Ahead-of-Time (AoT) prediction of production halts and defects in particleboard manufacturing that uses only pre-production input variables. The proposed approach incorporates both Single-Task Learning (STL) and Multi-Task Learning (MTL) paradigms, which are evaluated across three production lines under two modeling strategies: Line-Specific Modeling (LSM) and Line-Agnostic Modeling (LAM). The experimental evaluation benchmarks a lightweight Logistic Regression (LogR) model against three Automated Machine Learning (AutoML) techniques: H2O AutoML, Ludwig, and a Bayesian-optimized Deep Feedforward Network (DFFN). Results show that the STL-LSM combination using LogR achieves the highest overall predictive performance. To enhance model interpretability, we apply two model-agnostic eXplainable Artificial Intelligence (XAI) techniques: SHapley Additive exPlanations (SHAP) and One-Dimensional Sensitivity Analysis (1DSA). These methods generate feature importance rankings across targets and production lines, which are evaluated using quantitative (normalized distance metrics) and qualitative measures (alignment with domain expert insights). The XAI findings reveal a strong consistency between SHAP and 1DSA, with 1DSA requiring a substantially lower computational cost. Moreover, the convergence between model-derived interpretations and expert feedback highlights the practical relevance of the proposed ML framework in supporting data-driven decision-making for particleboard production planning.

1. Introduction

The Industry 4.0 concept represents the digital transformation of manufacturing through the seamless flow of data across supply chains, integrating technologies such as the Internet of Things (IoT)[1] and Machine Learning (ML) to create intelligent, customizable and efficient production systems [1]. This work details an Industry 4.0 Research and Development (R&D) collaboration project with a Portuguese company specialized in the production of decorated particleboards that are widely used in furniture and construction. The production process spans three production lines, where the particleboards are assembled between decorative paper layers, pressed, inspected for defects and finally packaged. To maintain product consistency and quality, each production line operates based on specific product recipes that establish the equipment and machinery settings. In terms of operational requirements, the company must design production plans for the upcoming days (e.g., up to two weeks), determining the sequence of orders and allocating recipes and resources to meet customer demand. However, inaccuracies in recipe definitions or resource allocations often lead to production halts and defects, wasting time, materials, energy, and labor. To address these inefficiencies, the company identified the potential of using ML algorithms to predict Ahead-of-Time (AoT) production halts and defects.

There are several recent related works that have approached production halts (e.g., [2,3]) or defects (e.g., [4,5]) but not both tasks. Moreover, as detailed in Section 2, the state-of-the-art studies focus only on the execution or inspection production stages, using data records retrieved once the production control elements (e.g., product recipe) are fixed. In contrast, in this paper, we assume an ML AoT prediction of both production halts and defects that uses only pre-production data, thereby enabling the predictive models to act as surrogate tools for production planning. Moreover, by adopting eXplainable Artificial Intelligence (XAI) methods, the predictive ML

* Corresponding author at: CCG/ZGDV ICT Institute, Universidade do Minho, Campus de Azurém, ed. 14, 4800-058, Guimarães, Portugal.
E-mail addresses: id10138@alunos.uminho.pt (A. Matta), pcortez@dsi.uminho.pt (P. Cortez).
[1] The full list of the acronyms used in this manuscript is presented in Table A.1.

https://doi.org/10.1016/j.dajour.2025.100607
Received 15 April 2025; Received in revised form 25 June 2025; Accepted 7 July 2025
Available online 9 July 2025

models can also be used to estimate the impact of adjustments to product recipes, paving the way for recipe optimization and reduced halts and defects.

The main contributions of this research are:

1. We simultaneously address quality control (production defects) and production uptime (halts) across multiple production lines.
2. Our framework evaluates multiple ML algorithms, namely a Logistic Regression [6] and three Automated Machine Learning (AutoML) tools: H2O AutoML [7], Ludwig [8], and a Bayesian-optimized Deep FeedForward Network [9,10]. These ML algorithms are explored using both a Single-Task Learning (STL) approach, where each target is predicted independently, and a Multi-Task Learning (MTL) approach, which predicts both targets simultaneously from a single ML model. Additionally, we compare a Line-Specific Modeling (LSM), where separate models are trained for each production line, with a Line-Agnostic Modeling (LAM), which treats all lines as a single unified dataset.
3. Unlike most studies that focus on the execution or inspection phases (e.g., [3,4]), we emphasize the planning phase of production. Such a goal is challenging since we can only use pre-production input features, i.e., that are known prior to the execution of the particleboard production.
4. To improve model transparency and trustworthiness, we apply two XAI methods, a SHapley Additive exPlanations (SHAP) [11,12] and One-Dimensional Sensitivity Analysis (1DSA) [13,14], to generate global input feature importance rankings. We assess the consistency of these rankings across tasks and production lines using the Normalized Kendall Tau Distance [15]. In addition, we compare model-derived rankings with those provided by domain experts, offering empirical evidence of alignment between model reasoning and expert knowledge. This alignment reinforces the potential for a reliable and actionable deployment of the models in industrial environments.

The proposed ML framework presents immediate and practical implications for industrial particleboard production settings. By facilitating AoT predictions of halts and defects utilizing solely pre-production data, manufacturers can modify production plans and adjust recipes prior to execution, thereby mitigating the risk of disruptions and defects. This predictive capability transforms conventional reactive quality control into a preventive approach, which can potentially reduce material, energy, time and labor waste. Furthermore, the application of XAI methods offers transparent and interpretable insights regarding how specific recipe configurations and input settings affect the production outcomes. This empowers process engineers and planners with the capacity to adjust product recipes in a more informed and targeted manner. Ultimately, the ML approach promotes more efficient, reliable, and sustainable operations and can be generalized to other manufacturing scenarios that involve recipe-based production planning and exhibit high variability in operational outcomes.

This paper is organized as follows: Section 2 reviews the related work and highlights its connections to our work. Section 3 describes the industrial data analyzed, the proposed ML methods, and the evaluation procedure. The results are presented and analyzed in Section 4, in terms of predictive performance (Section 4.1) and XAI analysis (Section 4.2). Finally, Section 5 summarizes the main conclusions and outlines the directions for future research.

2. Related work

The related works are summarized in Table 1 and characterized in terms of the **Year** of publication, the prediction **Goal** (**Q**uality **C**ontrol or **P**roduction **U**ptime), the **Production Phase** (**Plan**ning, **Exec**ution, or **Insp**ection), the type of **XAI Analysis** (**Quan**titative or **Qual**itative), the **XAI Method**, the **ML Method**, and the **Industry** type. Table 1 lists the relevant studies in chronological order, with our previous work [16] appearing second to last and the current study last. The distinction between our two works is detailed at the end of this section.

In the domain of quality control (e.g., product defect detection), distinct approaches have been explored. Image-based methods, often based on Convolutional Neural Networks (CNNs), are prevalent [4,20,23,25,27–30,32,33]. In addition, tabular data methods, using operational or sensory data from production, have gained traction. These methods employ supervised [5,19,22,24,26,31] and unsupervised [17,18,21] ML algorithms. Most studies focus on data from the inspection phase [5,24–32], allowing the identification of defects in finished products. Some studies also utilize data from the execution phase [17–23], which supports real-time monitoring of production processes. In terms of production uptime (e.g., predicting production halts), tabular data approaches dominate, analyzing operational or sensory data created during production. These studies mainly depend on supervised ML techniques [2,3,34–44]. As shown by the last row of Table 1, the novelty of our research is that it approaches both quality control (production defects) and production uptime (production halts) using tabular input features that are known prior to production execution.

Regarding the explainability of the model and the adoption of XAI methods, only a limited number of studies address this critical aspect [3,5,26,36,38,39,41,42,44]. These studies typically use the SHAP method [5,26,36,39,41,42,44], while some integrate additional techniques, such as Local Interpretable Model-agnostic Explanations (LIME) [3,5,41,44], 1DSA [38], Partial Dependence Plots (PDP) [41], and Individual Conditional Expectation (ICE) [41]. However, these studies generally only perform quantitative and technical analysis of the explanations, lacking qualitative evaluations that compare the relevance and precision of the explanations against the knowledge of experts in the domain. In contrast with the previous works, this paper performs both a quantitative and a qualitative analysis of two XAI methods (SHAP and 1DSA), bridging the gap between the computational execution of the XAI methods and their practical relevance.

Our previous work [16] also focused on predicting production halts and defects, although it was limited to a single particleboard production line. Moreover, in [16] we have only applied one XAI method (1DSA) as a demonstration example. In contrast, this study significantly broadens the scope by incorporating multiple production lines. This expanded scenario introduced an additional layer of experimentation: a comparison between LSM and LAM, akin to the STL versus MTL approach, enabling us to assess whether individual models per production line outperform a unified model for all lines. Furthermore, we extend the range of ML models to include LogR, a lightweight computationally efficient model, and Ludwig, a low-code AutoML framework that works with both STL and MTL tasks. In addition, XAI analysis was also improved by comparing SHAP, a widely used method, with 1DSA. This comparison includes both quantitative (e.g. computational effort, input relevance alignment) and qualitative analysis, in which the domain expert knowledge was compared with the XAI results.

3. Materials and methods

3.1. Industrial data

The production of decorative particleboards involves a complex material flow, transforming raw materials into semi-finished and finished products through a series of processes guided by the production plan. This planning process comprises two key phases. In the first phase, a macro-plan is created to define potential working days, shifts, production capacity, and production blocks. In the second phase, specific production orders are formalized to execute the macro-plan. These orders address customer demands, interfactory transfers, stock replenishment, and line-specific production needs (which are addressed in this research).

The particleboard production company is currently implementing an Industry 4.0 environment, with an increasing digitalization of the

Table 1
Related works for ML production uptime and quality prediction.

Ref.	Year	Goal		Production phase			XAI analysis		XAI method[a]	ML method[b]	Industry
		Q.C.	P.U.	Plan.	Exec.	Insp.	Quan.	Qual.			
[17]	2021	✓	–	–	✓	–	–	–	–	IF	Welding
[18]	2021	✓	–	–	✓	–	–	–	–	K-Means, AE, ANN, GMM	Metallurgy
[19]	2021	✓	–	–	✓	–	–	–	–	RF, GBT, SVM, AE	Particleboard
[20]	2022	✓	–	–	✓	–	–	–	–	CNN	Printed Circuit Boards
[21]	2023	✓	–	–	✓	–	–	–	–	AE, KMeans	Laminates
[22]	2023	✓	–	–	✓	–	–	–	–	RF, DT, GBT, AdaBoost, XGBoost	Manufacturing
[23]	2024	✓	–	–	✓	–	–	–	–	CNN	Textile
[24]	2021	✓	–	–	–	✓	–	–	–	SVM, AdaBoost	Particleboard
[25]	2022	✓	–	–	–	✓	–	–	–	CNN	Composite Panels
[26]	2022	✓	–	–	–	✓	✓	–	SHAP	GBT	Metallurgy
[27]	2022	✓	–	–	–	✓	–	–	–	CNN, RNN, LSTM	Laminates
[28]	2023	✓	–	–	–	✓	–	–	–	CNN	Semiconductors
[29]	2023	✓	–	–	–	✓	–	–	–	CNN	Additive
[30]	2023	✓	–	–	–	✓	–	–	–	CNN	Laminates
[31]	2023	✓	–	–	–	✓	–	–	–	GBR	Metallurgy
[5]	2024	✓	–	–	–	✓	✓	–	SHAP, LIME	RF, GBT, SVM, ANN, LGBM	Tire
[32]	2024	✓	–	–	–	✓	–	–	–	SVM, LogR, RF, ANN, CNN	Metallurgy
[33]	2024	✓	–	–	–	✓	–	–	–	CNN	Metallurgy
[4]	2024	✓	–	–	–	✓	–	–	–	CNN	Metallurgy
[34]	2022	–	✓	–	✓	–	–	–	–	RNN, LSTM, AE	Paper, Milling
[35]	2022	–	✓	–	✓	–	–	–	–	XGBoost	Metallurgy
[36]	2022	–	✓	–	✓	–	✓	–	SHAP	XGBoost, RF	3-D Printer
[37]	2023	–	✓	–	✓	–	–	–	–	LSTM	manufacturing
[2]	2023	–	✓	–	✓	–	–	–	–	RF, GBT, ANN	Manufacturing
[38]	2023	–	✓	–	✓	–	✓	–	1DSA	XGBoost, DFFN, RF, AE, IF	Particleboard
[39]	2023	–	✓	–	✓	–	✓	–	SHAP	ANN, SVM, RF	Manufacturing
[40]	2023	–	✓	–	✓	–	–	–	–	GBT, DT, ANN, CNN, LSTM, XGBoost	Manufacturing
[41]	2024	–	✓	–	✓	–	✓	–	LIME, SHAP, PDP, ICE	SVM, KNN, DT, RF	Manufacturing
[3]	2024	–	✓	–	✓	–	✓	–	LIME	SVM, RF, DT, KNN, LogR, NB	Manufacturing
[42]	2024	–	✓	–	✓	–	✓	–	SHAP	KNN, SVM, LinR, ANN, LGBM, XGBoost, GBT, …	Automotive
[43]	2024	–	✓	–	✓	–	–	–	–	RF, SVM, KNN, XGBoost, LogR	Subsea cables
[44]	2024	–	✓	–	✓	–	✓	–	SHAP, LIME	LogR, XGBoost, SVM, NB, RF	Manufacturing
[16]	2024	✓	✓	✓	–	–	✓	–	1DSA	DT, AutoML (LSM)	Particleboard
This work	–	✓	✓	✓	–	–	✓	✓	SHAP, 1DSA	LogR, AutoML (LSM, LAM)	Particleboard

[a] **XAI Method**: **SHAP** - SHapley Additive exPlanations; **1DSA** - 1-Dimensional Sensitivity Analysis; **LIME** - Local Interpretable Model-agnostic Explanations; **PDP** - Partial Dependence Plot; **ICE** - Individual Conditional Expectation;

[b] **ML Method**: **AE** - AutoEncoder; **ANN** - Artificial Neural Network; **AutoML** - Automated Machine Learning; **CNN** - Convolutional Neural Network; **DFFN** - Deep Feed Forward Network; **DT** - Decision Tree; **GBR** - Gradient Boosting Regressor; **GBT** - Gradient Boosting Tree; **GMM** - Gaussian Mixture Model; **IF** - Isolation Forest; **KNN** - K Nearest Neighbors; **LGBM** - Light Gradient Boosting Machine; **LinR** - Linear Regression; **LogR** - Logistic Regression; **LSTM** - Long Short Term Memory; **NB** - Naive Bayes; **RF** - Random Forest; **RNN** - Recurrent Neural Networks; **SVM** - Support Vector Machine; **XGBoost** - eXtreme Gradient Boosting (see also Table A.1);

planning and production processes. Once the manually designed production plan is finalized, it is integrated into the Manufacturing Execution System (MES), transitioning to a digital version. This digital plan provides a real-time visual representation of the current planning status and analytics for future orders. It includes alarms and statistical insights on potential halts, setup times, and other critical measures, utilizing data from more than 700 IoT sensors distributed across production lines.

In this study, industrial data were retrieved from the MES. It consists of historical production plans from January 1, 2023, to June 30, 2024, including product recipes and the characteristics of raw materials and finished products from three production lines (PL1, PL2, PL3). The product recipes specify machinery configuration parameters such as temperature, pressure, and cycle times, while the material characteristics cover aspects like composition, dimensions, and intrinsic properties (e.g., resin and volatile content).

The preprocessing began with a data cleaning step to eliminate records affected by unexpected environmental factors (e.g., fires, power outages). After this cleaning, the raw data still contained missing values due to two main reasons: (i) the layering process, which can involve decorative paper on the top, bottom, or both layers, leaving blank fields when no paper was used; and (ii) incomplete manual entries for certain product recipe attributes. To handle the first issue, missing values were replaced with a new "N/A" category for categorical attributes or with 0 for numerical attributes. For the second issue, we applied a hot-deck imputation [45] using a K-Nearest Neighbors (KNN) algorithm, specifically a 5-NN model as implemented by the `scikit-learn` Python package [6], trained on the available data.

Furthermore, since several ML algorithms (e.g., LogR and DFFN) require numeric inputs, all categorical variables were first converted into a numeric format by using the Inverse Document Frequency (IDF) transform [46] from the *cane*[2] Python package [47]. The transform is applied to the training data, where each level $l \in x_i$ is numerically coded as: $IDF(l) = \ln(\frac{N}{f_l})$, where x_i denotes the categorical input analyzed, N is the total number of instances and f_l is the number of occurrences of level l. The rationale of this transform is to put more frequent levels closer to the left numerical scale (0) and more widely spread, while infrequent levels are numerically closer to each other, tending to a maximum value of $ln(N)$ when $f_l = 1$ (right side of the

[2] Available at https://pypi.org/project/cane/.

numerical scale). Compared to the traditional One-Hot Encoding (OHE) approach, the IDF transformation provides a more compact representation by encoding each categorical variable as a single numeric input. For example, the IDF transformation produced a dataset with 26 input features, while OHE resulted in 115 features. This substantial reduction in dimensionality suggests potential improvements in computational efficiency and model interpretability. To assess this, we conducted a preliminary experiment comparing both encoding techniques using the same LogR model. The results did not indicate differences in predictive performance; however, the average training time increased nearly tenfold with OHE, from 0.067 s (IDF) to 0.618 s (OHE).

We also performed another preliminary experiment using two ML algorithms (LogR and DFFN) to test the application of a z-score transform to all numerical inputs, including those derived from IDF-encoded categorical variables. The experiment showed that there was no improvement in terms of predictive performance, and therefore we opted to discard this z-score transform. Finally, the dataset was also organized chronologically to enable temporal rolling window evaluation experiments, as detailed in Section 3.3.

Table 2 outlines the data attributes analyzed, their types, and the methods used to address missing values. After preprocessing, the final dataset contains 6478 rows (each corresponding to a particleboard production process), 26 input attributes, and two target variables: defects (with 51.71% positive cases) and halts (with 59.73% positive cases). We emphasize that the 26 input attributes refer to data values that can be known or set before the execution of the particleboard production process. Thus, an ML prediction model that uses these inputs can be used for planning purposes.

In the same preliminary computational experiments, we explored the use of an additional input attribute indicating the specific production line associated with each entry for the LAM approach. However, this feature had no noticeable impact on predictive performance across all models. As a result, it was excluded from the experiments reported in this work.

3.2. ML and XAI methods

In this work, we used a LogR model and three AutoML methods to address two binary classification tasks: predicting production halts and defects in decorated particleboard manufacturing. For the STL approach, where the model predicts only one target (thus two distinct models are trained), we used the LogR algorithm from the `scikit-learn` `Python` `module` [6] (with default values), the popular H2O AutoML tool [7], a DFFN architecture [9] optimized through Bayesian search [10], and the Ludwig AutoML framework [8]. Since LogR and H2O do not support MTL, we relied solely on DFFN and Ludwig for the MTL models, which can naturally handle multiple tasks by incorporating separate output nodes for each classification target. All ML implementations were performed in Python, using `scikit-learn` [6] (for LogR), `h2o`, `ludwig`, `tensorflow` (for DFFN) and `optuna` (for Bayesian search) libraries.

To reduce computational effort, the selection of the ML model (H2O, Ludwig) and the hyperparameters (H2O, Ludwig, DFFN) was limited to the first iteration of the Rolling Window (RW) evaluation procedure (Section 3.3). After selecting the model and fixing its hyperparameters, the selected ML algorithms were trained in all RW iterations, thus updating over time. To perform the ML model and hyperparameter selection (H2O, Ludwig, DFFN) an internal 5-fold cross-validation was applied to the oldest training data (initial W data records of the RW procedure, see Section 3.3). The model selection criterion was the best Area Under the Curve (AUC) of the Receiver Operating Characteristic (ROC) curve [48], calculated over the validation data.

The H2O AutoML tool was configured to explore four algorithms and their hyperparameters (ρ representing the number of hyperparameters tuned): Distributed Random Forest (DRF) ($\rho = 0$), Generalized Linear Model with Regularization (GLM) ($\rho = 1$), Extreme Gradient Boosting (XGBoost) ($\rho = 9$), and Gradient Boosted Machines (GBM) ($\rho = 8$). These algorithms were chosen as default options in H2O, excluding deep learning (already covered by the DFFN approach due to its computational demand) and stacked ensembles (to simplify the explainability analysis). For each AutoML run, the tool was configured to stop after 10 min of execution (details of the computational environment are provided in Section 4.1). The selected models and hyperparameter values are detailed in Tables B.1–B.7 in Appendix B.

A DFFN is a type of deep multilayer perceptron that uses only forward connections between layers. By stacking multiple hidden layers, DFFNs can progressively learn more abstract and powerful representations of input data, which improves their classification performance [9]. Given the large number of hyperparameters to tune for DFFNs, we used an automated Bayesian optimization approach, which is particularly useful when the objective function is computationally expensive and the search space is large [49].

The adopted DFFN assumes a maximum number of units of 1024 for the first hidden layer, with each subsequent layer restricted to a maximum size that is 1.2 times greater than the preceding one, forming a trapezoidal structure that has been shown to perform well for classification tasks [50]. Additionally, the Bayesian search was designed to prevent two consecutive dropout layers. DFFNs were trained using the Adam optimizer with a binary cross-entropy loss function, and training was halted either when the early stopping AUC value (calculated on 10% of the training data) no longer improved or after 200 epochs. Bayesian optimization ran for 100 trials, testing 100 different DFFN architectures. The details of the searched DFFN hyperparameters and the Bayesian optimized values are shown in Table B.8.

Ludwig is an open-source framework designed to simplify the creation of deep learning models, making it accessible to users with minimal programming experience. At the heart of Ludwig's design is its core modeling architecture, known as ECD (Encoder–Combiner–Decoder). Input features are processed by specialized encoders that transform raw data into latent representations. These representations are then passed through a Combiner, which integrates encoded inputs into a unified latent space. On the output side, the combiner's output is routed to decoders for each defined output feature, facilitating predictions and post-processing. This modular and flexible architecture supports a wide variety of input types, such as text, images, and timeseries, and enables Ludwig to handle multimodal data seamlessly.

For our experiments, Ludwig used an ADAM optimizer [51], which is an adaptive learning rate optimization algorithm that efficiently and effectively updates model parameters during training, and a TabNet combiner [52], which uses attention and sparsity to achieve high performance on tabular data. Similarly to H2O execution, each Ludwig run was set to stop after 10 min. We note that in some cases the full Ludwig execution time may exceed the limit, as the tool continues training the current model even after the time cap is reached. The trainer and combiner hyperparameters and respective searchable and selected values are described in Tables B.9–B.12 in Appendix B.

In this study, we applied two model-agnostic XAI methods (SHAP and 1DSA), aiming to examine the overall contributions of the various input features (described in Table 2) to the models' predictions.

SHAP [11,12] is an XAI method that assigns to each input feature a contribution value that indicates its impact on the prediction of a model. Based on cooperative game theory, SHAP computes Shapley values to fairly distribute the difference between a model's actual output and its average prediction among all input features. It can explain which features influenced a prediction, as well as the direction and magnitude of their effects. However, because it approximates the Shapley values by evaluating many combinations of feature subsets, SHAP is computationally intensive.

On the other hand, 1DSA is a lightweight method that examines how variations in input features influence the model output [13,14]. By perturbing one feature at a time (hence the term one-dimensional) while holding others constant (e.g., to their average values), 1DSA

Table 2
Description of the dataset input features and targets.

Attribute	Data type [categorical levels/range]: examples	Null handling (% missing)
Quantity of Decorated Particleboard to be Produced (QTY)	Integer [2 to 5100]: 30, 264, 1440, …	–
Decorated Particleboard Characteristic Code (DPCC)	String [17 levels]: BATB0, BATB7, BATF0, …	–
Decorated Particleboard Length (DPL)	Integer [2440 to 5700]: 2440, 2750, 3660, …	–
Decorated Particleboard Width (DPW)	Integer [1220 to 2120]: 1220, 1830, 2070, …	–
Decorated Particleboard Thickness (DPT)	Integer [3 to 40]: 3, 10, 16, …	–
Decorated Particleboard Top Finishing (DPTF)	String [16 levels]: ASM, CMS, IW1, …	–
Decorated Particleboard Bottom Finishing (DPBF)	String [17 levels]: ASM, CMS, IW1, …	–
Particleboard Characteristic Code (PCC)	String [16 levels]: AB0B1, AB0F1, AD2B2, …	–
Particleboard Finishing (PF)	String [4 levels]: NLX21, LIXAG25, LIX150, LIX	–
Top Paper Characteristic code (TPCC)	String [8 levels]: N/A, PBA02, PBA71, PBA75, …	Replaced by “N/A” (4.13%)
Top Paper Resin Content (TPRC)	Double [0 to 72.6]: 0, 58.2, 59.7, 63.5, …	Replaced by 0 (4.13%)
Top Paper Volatile Content (TPVC)	Double [0 to 9.9]: 0, 4.7, 5.6, 6.4, …	Replaced by 0 (4.13%)
Top Paper Manufacturing Process (TPMP)	String [6 levels]: N/A, BPDHI, BPDUP, BPH2O, …	Replaced by “N/A” (4.13%)
Top Paper Resin Weight (TPRW)	Double [0 to 284]: 0, 108, 162, 177, …	Replaced by 0 (4.13%)
Top Paper Humidity Weight (TPHW)	Double [0 to 284]: 0, 108, 162, 177, …	Replaced by 0 (4.13%)
Bottom Paper Characteristic Code (BPCC)	String [8 levels]: N/A, PBA02, PBA71, PBA75, …	Replaced by “N/A” (5.96%)
Bottom Paper Resin Content (BPRC)	Double [0 to 69.5]: 0, 58.2, 59.7, 63.5, …	Replaced by 0 (5.96%)
Bottom Paper Volatile Content (BPVC)	Double [0 to 11.3]: 0, 4.7, 5.6, 6.4, …	Replaced by 0 (5.96%)
Bottom Paper Manufacturing Process (BPMP)	String [6 levels]: N/A, BPDHI, BPDUP, BPH2O, …	Replaced by “N/A” (5.96%)
Bottom Paper Resin Weight (BPRW)	Double [0 to 284]: 0, 108, 162, 177, …	Replaced by 0 (5.96%)
Bottom Paper Humidity Weight (BPHW)	Double [0 to 284]: 0, 108, 162, 177, …	Replaced by 0 (5.96%)
Recipe Value for the Press’ Top Plate Temperature (TEMPUP)	Double [165 to 210]: 175.0, 195.0, 200.0, …	KNN (19.13%)
Recipe Value for the Press’ Bottom Plate Temperature (TEMPDW)	Double [165 to 210]: 175.0, 195.0, 200.0, …	KNN (19.13%)
Recipe Value for the Mechanical Cycle (MECHCT)	Double [7.7 to 18]: 7.9, 10.0, 14.0, …	KNN (19.13%)
Recipe Value for the Thermal Cycle (THERMCT)	Double [12 to 50]: 12.0, 19.0, 26.0, …	KNN (19.26%)
Recipe Value for the Number of Press Cycles per Hour (CYCHOUR)	Double [27.07 to 180]: 28.125, 61.02, 81.45, …	KNN (19.12%)
Flag Indicating the Occurrence of Defects	Bool: 0 (48.29%), 1 (51.71%)	–
Flag Indicating the Occurrence of Halts	Bool: 0 (40.27%), 1 (59.73%)	–

identifies the magnitude of change in the predictions. The 1DSA method was implemented using the `rminer`[3] R package, while SHAP was applied using the `shap`[4] Python package.

3.3. Evaluation

In this study, we evaluate the performance of the ML models using a robust and realistic Rolling Window (RW) scheme [53], which simulates how classifiers are used over time with periodic updates to training and test sets (Fig. 1). In the first iteration ($u = 1$), the model is trained using a dataset window consisting of the oldest W instances, after which the model predicts T future observations. In each subsequent iteration ($u = 2$, $u = 3$, etc.), the training set is updated by replacing the oldest S records with the next S observations (the update step). Then a new model is trained to predict the next T observations. This process is repeated iteratively until all data are processed, yielding a total of $U = \frac{D-(W+T)}{S}$ updates, rounded down to the nearest integer, where D is the total number of examples in the dataset.

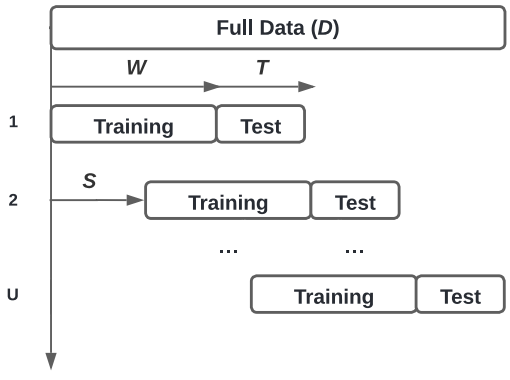


Fig. 1. Schematic of the Rolling Window (RW) evaluation.

We adopted a temporal RW evaluation strategy, where each window is defined by a fixed time duration rather than a fixed number of observations. Based solely on consultations with domain experts and to align with operational practices, we set the training window to $W = 1$ year, which corresponds to approximately 70% of the available data, ensuring that each model is trained on a sufficiently large and representative sample. The test window of $T = 2$ weeks was chosen to reflect the typical duration of the production planning cycle, allowing the evaluation to be performed on realistic predictive horizons. The step size S was set equal to T, resulting in $U = 10$ non-overlapping and consecutive RW test sets per target variable. As described in Section 3.2, model selection and hyperparameter tuning were performed only during the initial RW iteration ($u = 1$) to mimic a realistic deployment scenario and reduce computational overhead.

The performance of the model on the RW test data was evaluated using the popular Area Under the Curve (AUC) measure of the Receiver Operating Characteristic (ROC) curve [48]. For each predicted class probability $p \in [0, 1]$, the predicted label was classified as positive if $p_k > K$, where K is the decision threshold. We selected this measure as the primary predictive performance evaluation metric due to its threshold-independent nature, providing a comprehensive view of the model discrimination performance across all classification thresholds. This makes AUC particularly suitable for comparing different models in scenarios where the optimal decision threshold is not known in advance. The ROC curve plots 1−Specificity, or False Positive Rate (FPR), on the x axis against Sensitivity, or True Positive Rate (TPR), on the y axis, across all possible threshold values ($K \in [0, 1]$). The AUC is calculated as the area under the ROC curve, that is, $AUC = \int_0^1 ROC(K)dK$. AUC values can be interpreted as follows [46]: 0.5 indicates a random classification, 0.7 is considered reasonable, 0.8 is very good, 0.9 is excellent and 1.0 corresponds to perfect class discrimination. To aggregate results across RW iterations ($u \in 1, \ldots, U$), the median AUC value was used, as it is less sensitive to outliers compared to the average statistic. Furthermore, a paired Wilcoxon signed rank test [54] was conducted to determine the statistical significance of the differences between the median AUC values obtained by two given ML approaches. In addition, we also store the computational effort, measured in terms of the time elapsed for model selection, training, and inference (i.e., the time required to predict a single example).

[3] Available at https://CRAN.R-project.org/package=rminer.
[4] Available at https://shap.readthedocs.io/en/latest/.

To ensure a fair performance evaluation of the LSM and LAM approaches, we computed both the production line-specific and global AUC scores. For the LSM approach, in each rolling window (RW) iteration, the test data from the three production lines were combined into a single dataset, and a global AUC score was calculated. The final evaluation was based on the median global AUC in all iterations, providing a comprehensive view of the overall performance of the model (Fig. 2(a)). In contrast, for the LAM approach, the test data in each RW iteration were filtered by the production line, allowing the computation of the specific AUC scores. The final evaluation relied on the median AUC for each line (Fig. 2(b)).

Once the best-performing ML approach and model were identified, we applied two XAI methods (1DSA and SHAP) to each trained model from the RW iterations, resulting in 10 distinct models. This analysis pursued two primary objectives: first, to leverage XAI methods to uncover how the models predict defects or halts; and second, to validate these insights using the practical expertise of three particleboard production company planners and operators. By comparing model reasoning with expert knowledge, we aimed to identify similarities and discrepancies, improving the reliability and transparency of predictive analytics for industrial use.

For the first objective, we performed a comparative analysis of the feature importance rankings of SHAP and 1DSA to identify areas of agreement and divergence. Both methods were applied independently, using the median importance values of features derived from each RW training iteration. To quantitatively assess the similarity between the SHAP and 1DSA feature importance rankings, we used the Normalized Kendall Tau Distance (NKTD) ranking metric [15]. NKTD measures rank dissimilarity by counting pairwise disagreements (inversions) between two rankings. An NKTD of 0 indicates complete alignment, while a value of 1 (or −1 for reversed rankings) signifies total disagreement. This NKTD analysis is executed in three stages:

1. **SHAP versus 1DSA alignment**: we computed the NKTD between SHAP and 1DSA rankings for each production line (PL1, PL2, PL3), separately for defects and halts, to evaluate their agreement.
2. **Production-Line comparison**: the NKTD between the importance rank of the features in the production lines was calculated for each XAI method, again separately for defects and halts, to examine the consistency between the production lines.
3. **Task comparison**: we computed the NKTD between rankings for defects and halts for each production line and XAI method to assess alignment between the two predictive tasks.

For the second objective, to assess the alignment between XAI knowledge and operator expertise, we presented the feature importance rankings obtained from the SHAP and 1DSA methods to the company's operators and collected their feedback. In addition, operators were asked to classify the features according to their expected impact levels ("Low", "Medium", "High", or "Very High") for both production defects and halts. To facilitate a direct comparison, we applied a quantile binarization process to the XAI-derived feature importance values, aligning them with the operators' categorical classifications. To quantify this alignment, we assign a numerical scale to categories (1 – "Low", 2 – "Medium"; 3 – "High"; and 4 – "Very High") and calculate the Mean Absolute Error (MAE) between the qualitative expert rankings and XAI-derived ones.

4. Results

4.1. Predictive results

The RW experiments were conducted on an Ubuntu 20.04 virtual machine equipped with 32 GB of memory and an Intel(R) Xeon(R) E5-2650 v4 @ 2.20 GHz CPU with 16 cores. Table 3 shows the average computational effort (for one RW iteration, in seconds) required by the different ML algorithms. The AutoML model selection process took approximately 8 min for the H2O tool, approximately 16 min for Ludwig, and 34 min for DFFN. Regarding the training and inference effort, Ludwig is the most computationally demanding tool, requiring around 3 min of training time and almost 1 s to predict one instance. In comparison, the other AutoML tools (H2O and DFFN) are substantially faster, requiring a training time of 15 to 16 s and an inference time of a few milliseconds. As expected, the LogR method (which does not require any model selection) is the fastest, requiring just around 5 centiseconds of training time and 0.3 ms to predict one example.

Table 3
Average computational effort (time elapsed, in seconds; best values in **bold**) for ML model selection, training and inference (one prediction).

Model	Time elapsed (s)		
	Selection	Training	Inference
LogR	–	**0.07**	**0.0003**
H2O	**509**	14.87	0.0111
Ludwig	951	162.41	0.7117
DFFN	2,064	16.14	0.0617

Table 4 presents the predictive performance of the RW evaluation, reporting the median AUC along with the interquartile range (IQR) in all $U = 10$ iterations for each ML algorithm under the LSM approach. The row labeled *ALL* reflects the aggregated performance across all production lines.

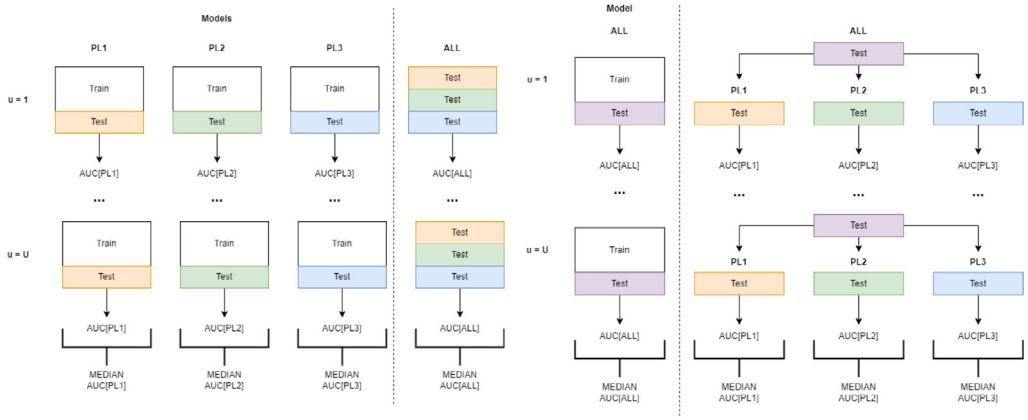
In the STL approach for defect prediction, the LogR, H2O AutoML and DFFN models exhibited similar performance, with no model distinctly outperforming the others, whereas Ludwig consistently underperformed across all production lines. In contrast, for halt prediction, LogR demonstrated superior performance in the majority of cases, only marginally lagging behind H2O in PL1.

In contrast, under the MTL setup, the DFFN model consistently outperformed Ludwig by a small margin in most scenarios, with the exception of the prediction of defects in PL3 and the corresponding global metric. However, all MTL configurations showed poorer performance relative to their STL counterparts, which ultimately motivated the selection of STL over MTL.

While MTL theoretically offers the advantage of leveraging shared representations to capture task correlations, its practical success hinges on the presence of strong and consistent interdependencies between targets. In our case, the XAI analysis performed in Section 4.2 (e.g., Table 9) revealed that defects and halts are influenced by different rankings of relevant input features. Consequently, the shared modeling constraints imposed by MTL tended to degrade the individual task performance. STL, on the contrary, enable ML models to independently specialize in the unique data distributions and predictive patterns of each target. This property resulted in more accurate and robust task-specific predictions.

For demonstration purposes, the left of Fig. 3 illustrates the ROC curve that was obtained by LogR (in blue, with an AUC of 0.81) for the $u = 7$th RW iteration to predict defects. For all threshold values, LogR outperforms Ludwig's STL predictive performance (in orange, with an AUC of 0.68). For the LogR curve, we selected two different thresholds: a more sensitive ($K = 0.21$, in red) and a more specific ($K = 0.69$, in green). The respective confusion matrices are shown to the right of Fig. 3. For each matrix, we present the False Negative Rate (FNR) and False Positive Rate (FPR) values. Within the Industry 4.0 context, it is often relevant to present low FNR values (e.g., low chance of missing a halt or defect), thus the $K = 0.21$ threshold is considered more relevant. In this case, an interesting FNR value of just 4% was obtained.

Taking into account both predictive performance and computational effort, the LogR algorithm emerges as a highly favorable ML option. In effect, LogR offers much faster training and comparable (or slightly higher) predictive performance when compared with the more complex algorithms. Consequently, the LogR model was selected for the LSM versus LAM experiments, with the results outlined in Table 5.



(a) Line-Specific Modeling (LSM) (b) Line-Agnostic Modeling (LAM)

Fig. 2. Schematic of the AUC calculation for the LSM (a) and LAM (b) approaches.

Table 4
Median AUC ± IQR results for the LSM approach (best results in **bold** for each ML predictive task, target and production line).

Task	Target	Prod. line	Algorithm			
			LogR	H2O	DFFN	LUDWIG
STL	Defects	PL1	0.655 ± 0.082	0.667 ± 0.044	**0.721 ± 0.047**	0.629 ± 0.046
		PL2	°0.694 ± 0.046	**0.724 ± 0.059**	0.679 ± 0.069	0.628 ± 0.057
		PL3	°0.819 ± 0.052	0.808 ± 0.057	0.800 ± 0.049	0.760 ± 0.067
		ALL†	°0.712 ± 0.041	**0.722 ± 0.053**	0.716 ± 0.040	0.657 ± 0.039
	Halts	PL1	°0.785 ± 0.023	**0.790 ± 0.064**	0.778 ± 0.028	0.698 ± 0.045
		PL2	°0.716 ± 0.054	0.704 ± 0.046	0.713 ± 0.048	0.572 ± 0.034
		PL3	° ⋆0.818 ± 0.031	0.780 ± 0.041	0.784 ± 0.076	0.698 ± 0.040
		ALL†	°0.756 ± 0.045	0.735 ± 0.043	0.744 ± 0.043	0.632 ± 0.032
MTL	Defects	PL1	–	–	**0.660 ± 0.067**	0.622 ± 0.083
		PL2	–	–	**0.694 ± 0.052**	0.679 ± 0.030
		PL3	–	–	0.758 ± 0.047	**0.784 ± 0.041**
		ALL†	–	–	0.686 ± 0.040	**0.696 ± 0.027**
	Halts	PL1	–	–	**0.791 ± 0.071**	0.743 ± 0.042
		PL2	–	–	**0.695 ± 0.034**	0.691 ± 0.063
		PL3	–	–	**0.813 ± 0.058**	0.805 ± 0.037
		ALL†	–	–	**0.726 ± 0.036**	0.707 ± 0.066

⋆Statistically significant compared to DFFN.
°Statistically significant compared to LUDWIG.
†Computed value (not an actual ML model).

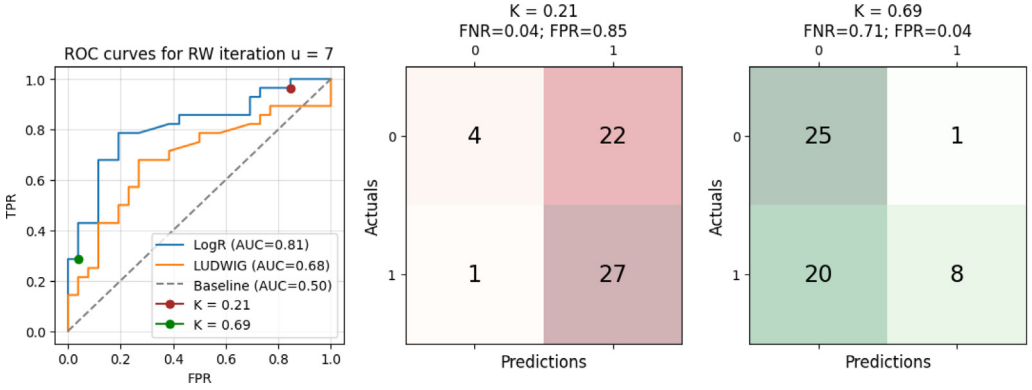


Fig. 3. Examples of two defects prediction ROC curves for the $u = 7$th RW iteration (left) and confusion matrices for two LogR selected threshold (K) values (right).

Table 5 compares the LSM and LAM approaches when adopting the LogR algorithm. The results reveal a slight preference for the LAM approach in the defect prediction task and a more pronounced preference for the LSM approach in the halt prediction task. However, the performance differences between the two approaches are minimal, particularly in the defect prediction case, with a mean difference of 0.036 across the three production lines and 0.009 in the global scenario. For the halt prediction, the differences are slightly larger, with a 0.048 mean difference between the production lines and a 0.052 difference in the global scenario. Given that most experiments favored the LSM approach, it was chosen as the best overall strategy. As a result, six separate LogR models were developed (one for each production line and predictive task).

Table 5
Comparison of LogR median AUC ± IQR results for LSM and LAM approaches (best results in **bold**).

Production line	Defects		Halts	
	LSM	LAM	LSM	LAM
PL1	0.655 ± 0.082	**0.697 ± 0.139**[a]	**0.785 ± 0.023**	0.737 ± 0.025[a]
PL2	0.694 ± 0.046	**0.728 ± 0.131**[a]	**0.716 ± 0.054**	0.664 ± 0.152[a]
PL3	**0.819 ± 0.052**	0.786 ± 0.099[a]	**0.818 ± 0.031**	0.773 ± 0.045[a]
ALL	0.712 ± 0.041[a]	**0.721 ± 0.038**	**0.756 ± 0.045**[a]	0.704 ± 0.054

[a] Computed value (not an actual ML model).

Although only pre-production inputs were used by the selected predictive models, the discrimination levels obtained are of quality, ranging from 70% (reasonable) to 82% (very good). In fact, the predictive results were shown to the domain experts, who considered them very interesting and useful for planning purposes.

4.2. Explainable AI analysis

In this study, we applied the SHAP and 1DSA XAI techniques to the selected LogR models, allowing us to examine the global contributions of the various input features (described in Table 2) to the models' predictions. Although LogR is a more interpretable model when compared with other tested models (e.g. XGBoost, DFFN), it still makes sense to apply XAI methods for this model. Both SHAP and 1DSA can provide visual graphs of the extracted knowledge that are easier to analyze by domain experts. Also, some of the explored XAI graphs, such as Variable Effect Characteristic (VEC) curves, are not trivial to retrieve directly from the LogR model. Similarly to our approach, we note that a recent related study also applied XAI methods (SHAP and LIME) after a LogR modeling [44]. Secondly, the explored SHAP and 1DSA methods are model agnostic, and thus the proposed XAI evaluation procedure can still be adopted if in the future other complex ML algorithms outperform the LogR model (e.g., new AutoML methods).

The SHAP and 1DSA methods were applied to all trained models (total of $U = 10$ distinct models) of the RW procedure. In preliminary computational experiments, we applied SHAP and 1DSA to the first logR models in the rolling window, using all training data. However, the computational time required by SHAP was considerably high compared to 1DSA for the same task. Thus, to reduce computational effort, we opted to use a sample of $R_s = 500$ entries (randomly extracted from the training set), using the same sample for both SHAP and 1DSA methods for a particular uth RW iteration. Using this smaller R_s sample, the processing time for both algorithms was significantly reduced. The execution times for SHAP and 1DSA for both sample sizes are presented in Table 6.

Table 6
Comparison of SHAP and 1DSA execution times per RW iteration, in seconds (best values in **bold**).

Sample size (R_s)	Algorithms	
	SHAP	1DSA
6,478	20,634.8	**12.8**
500	1,973.9	**7.2**

To visualize the rankings and variability of the feature importance scores, the outputs of both XAI methods were presented as boxplots of the $U = 10$ RW iterations (Figs. 4 and 5). In general, SHAP boxplots exhibit more narrow boxes and whiskers compared to 1DSA. The boxplots are colored and grouped according to a four-level scale quantile binarization (explained in the qualitative XAI analysis). Regarding the input rankings, the QTY feature was consistently positioned as the most important input. This was an expected result, as higher production volumes increase the likelihood of problems arising. Furthermore, TPRW, TPHW, and BPRC features consistently rank among the most important on all production lines. The features related to the actionable recipe values (TEMPUP, TEMPDW, THERMCT, MECHCT, and CYCHOUR) also exhibit medium to very high importance. On the other extreme, the DPBF and DPTF features consistently rank among the least important inputs.

For demonstration purposes, Fig. 6 illustrates the influence of variables QTY, TPRW, TPHW, and BPRC during the fifth iteration of the RW process, presented as Variable Effect Characteristic (VEC) curves. The VEC plots (SHAP – red curve; 1DSA – blue curve) show the average impact of each input variable on the model predictions [13,14]. The plots also include the input histograms on the training data (shown in terms of the gray bars). The visual analysis of Fig. 6 confirms a strong agreement between both XAI methods, which produce very similar VEC curves. Regarding the input influence, the VEC graphs confirm a stronger and positive non-linear effect of QTY on the probability of defects anomaly (for all production lines). As for the other inputs, there is in general a moderate negative and more linear effect in the predicted output response.

To quantitatively compare the importance of input features, we calculated the NKTD value between SHAP and 1DSA rankings for each production line (PL1, PL2, PL3), analyzing the defects and halts separately to evaluate the alignment between the two methods (Table 7). Next, we computed the NKTD between production lines for each XAI method, again separating defects and halts, to assess the alignment between the importance ranking of features between production lines (Table 8). Finally, we calculated the NKTD between defects and halts for each production line and XAI method to evaluate the alignment of the rankings between these predictive tasks (Table 9).

Table 7
Normalized Kendall Tau Distance (NKTD) between 1DSA and SHAP methods for each predictive task and production line.

Production line	Predictive task	
	Defects	Halts
PL1	0.08	0.08
PL2	0.08	0.10
PL3	0.10	0.17

Table 8
Normalized Kendall Tau Distance (NKTD) between production lines for each predictive task and XAI method.

Production line	Predictive task			
	Defects		Halts	
	1DSA	SHAP	1DSA	SHAP
PL1 vs. PL2	0.21	0.20	0.19	0.18
PL1 vs. PL3	0.23	0.24	0.29	0.37
PL2 vs. PL3	0.21	0.23	0.32	0.34

Table 9
Normalized Kendall Tau Distance (NKTD) between Defects and Halts for each production line and XAI method.

Production line	XAI method	
	1DSA	SHAP
PL1	0.23	0.26
PL2	0.21	0.20
PL3	0.25	0.27

The NKTD values that compare SHAP and 1DSA rankings (Table 7) remain consistently low across all production lines (below 0.17), indicating a strong alignment between both methods in the overall ranking of features importance. However, halts consistently exhibit slightly higher NKTD values compared to defects, reflecting a greater divergence between SHAP and 1DSA in capturing feature importance

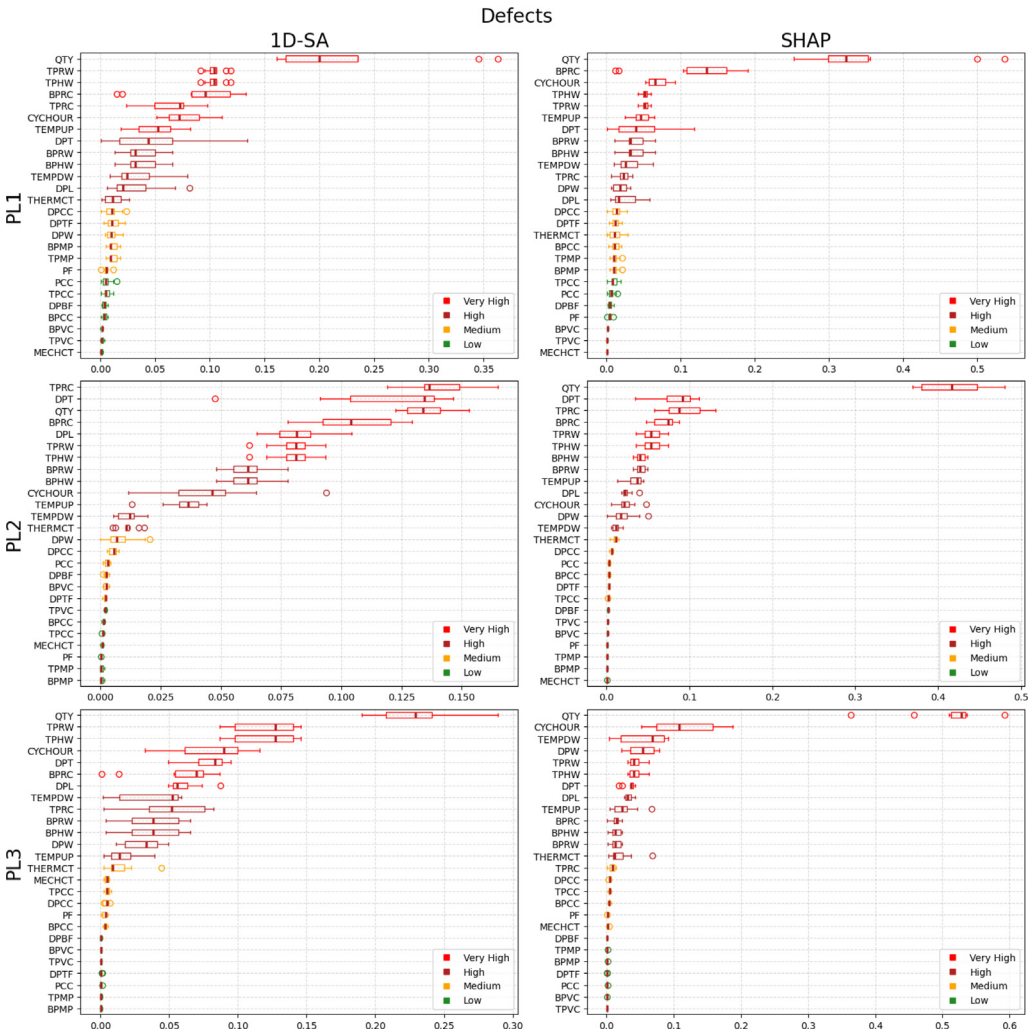


Fig. 4. Comparison of SHAP and 1DSA XAI methods for predicting defects on the three production lines (qualitative levels are signaled by using different colors).

for halt-related predictions. In particular, the NKTD values increase from 0.08 in PL1 to 0.17 in PL3, suggesting that PL3 halts are influenced by factors interpreted somewhat differently by SHAP and 1DSA. Regarding NKTD values between production lines (Table 8), these are more pronounced, ranging from 0.18 to 0.37. For defects, both SHAP and 1DSA show relatively stable NKTD values across production line comparisons, generally between 0.20 and 0.25. In contrast, halts exhibit higher NKTD values, particularly in comparisons involving PL1 vs. PL3 and PL2 vs. PL3. For example, 1DSA reports an NKTD of 0.29 for PL1 vs. PL3, while SHAP reaches 0.37 for the same comparison. These results denote some variability in the feature importance rankings between production lines, with SHAP demonstrating a greater sensitivity to production line differences than 1DSA. Lastly, a moderate disagreement is observed between the ranking of defects and halts for each production line and XAI method (Table 9), with NKTD values ranging from 0.20 to 0.27. This finding indicates that the feature importance rankings are somewhat task-dependent, varying between defects and halts for both XAI methods, which could explain why STL obtained better predictive results (see Section 4.1) when compared to MTL.

When presented with these findings, the operators' responses were moderately positive, with mixed perspectives on specific features. The identification of QTY as the most influential feature was received with some skepticism. Although its importance is operationally intuitive, since higher production volumes can correlate with a higher likelihood of halts or defects, experts noted that QTY is largely externally dictated by customer demand. As such, its predictive value offers limited utility for direct control or intervention. In contrast, the features related to the decorated particleboard, the top and bottom papers, and the raw board received favorable feedback. These features were highlighted as valuable because their impacts were less understood by the operators before the analysis. However, recipe-related features, particularly the top and bottom temperatures and the thermal cycle, were criticized for their moderate importance rankings, as operators viewed these parameters as critical to production outcomes.

Based on this feedback, the operators categorized the features into four qualitative levels ("Low", "Medium", "High", and "Very High") for their perceived impact on production halts and defects, as detailed in Table 10. The table also describes the input attributes (9 out of 26) that are actionable, i.e., the features that can be changed freely and are not externally driven.

To obtain the qualitative level classifications directly from the XAI methods and avoid the need for manual threshold selection, we applied a quantile binarization to the XAI feature importance values, as illustrated in the boxplot color scheme presented in Figs. 4 and 5. The resulting classifications of the predictive models were then compared with those provided by the operators, as shown in Table 11.

The comparison revealed several similarities and some differences between the feature importance rankings provided by the operators and those derived from the XAI methods. For example, features such as QTY, TPRC, BPRC, and DPCC exhibited strong agreement between qualitative assessments and model-derived rankings. Other input features, such as PCC, PF, and MECHCT, also showed a close alignment,

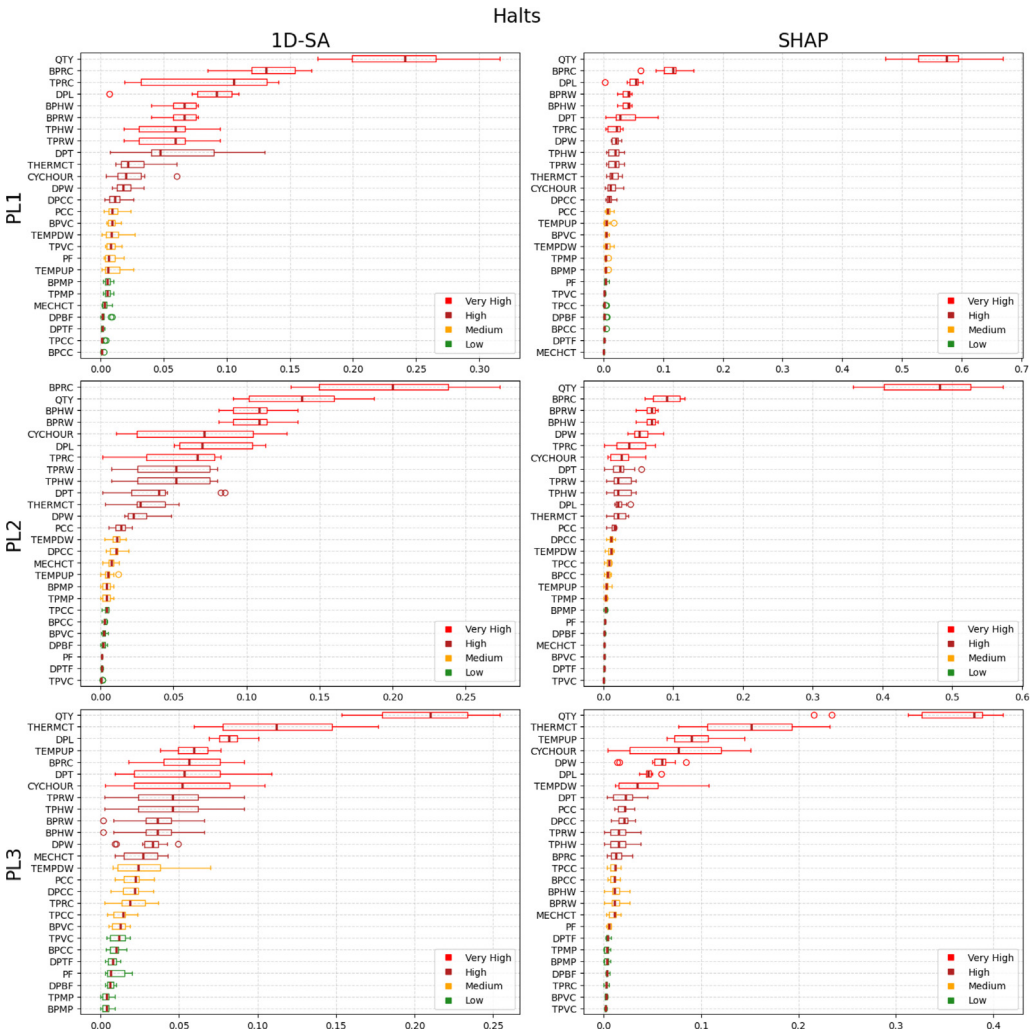
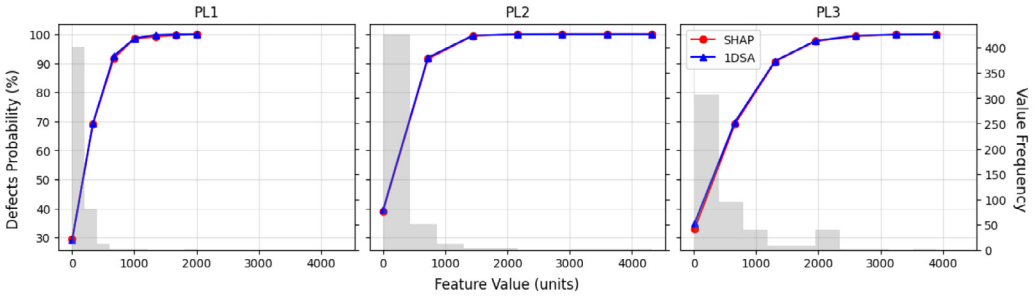


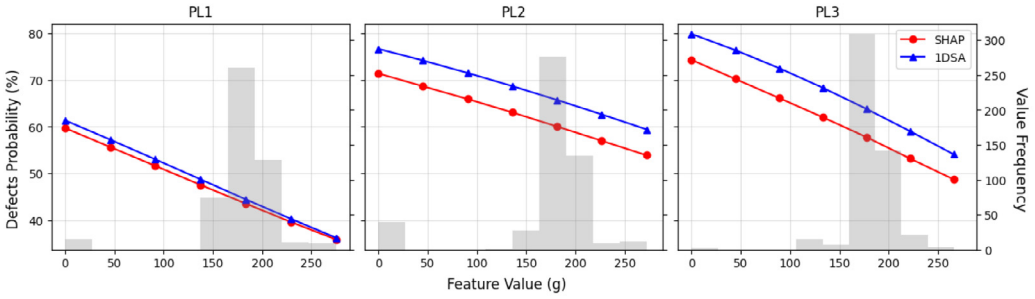
Fig. 5. Comparison of SHAP and 1DSA XAI methods for predicting halts on the three production lines (qualitative levels are signaled by using different colors).

Table 10
Qualitative importance assigned to the inputs by the domain experts (operators).

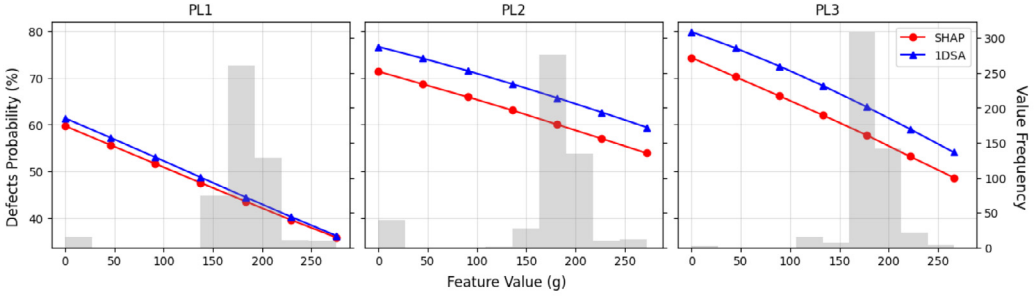
Attribute	Perceived impact	Actionable
Quantity of Decorated Particleboard to be Produced (QTY)	Very High	–
Decorated Particleboard Characteristic Code (DPCC)	Medium	–
Decorated Particleboard Length (DPL)	Low	–
Decorated Particleboard Width (DPW)	Low	–
Decorated Particleboard Thickness (DPT)	Medium	–
Decorated Particleboard Top Finishing (DPTF)	High	–
Decorated Particleboard Bottom Finishing (DPBF)	High	–
Particleboard Characteristic Code (PCC)	Medium	–
Particleboard Finishing (PF)	Medium	–
Top Paper Characteristic code (TPCC)	High	–
Top Paper Resin Content (TPRC)	Very High	✓
Top Paper Volatile Content (TPVC)	Very High	✓
Top Paper Manufacturing Process (TPMP)	High	–
Top Paper Resin Weight (TPRW)	Medium	–
Top Paper Humidity Weight (TPHW)	Medium	–
Bottom Paper Characteristic Code (BPCC)	High	–
Bottom Paper Resin Content (BPRC)	Very High	✓
Bottom Paper Volatile Content (BPVC)	Very High	✓
Bottom Paper Manufacturing Process (BPMP)	High	–
Bottom Paper Resin Weight (BPRW)	Medium	–
Bottom Paper Humidity Weight (BPHW)	Medium	–
Recipe Value for the Press' Top Plate Temperature (TEMPUP)	Very High	✓
Recipe Value for the Press' Bottom Plate Temperature (TEMPDW)	Very High	✓
Recipe Value for the Mechanical Cycle (MECHCT)	Medium	✓
Recipe Value for the Thermal Cycle (THERMCT)	Very High	✓
Recipe Value for the Number of Press Cycles per Hour (CYCHOUR)	Medium	✓



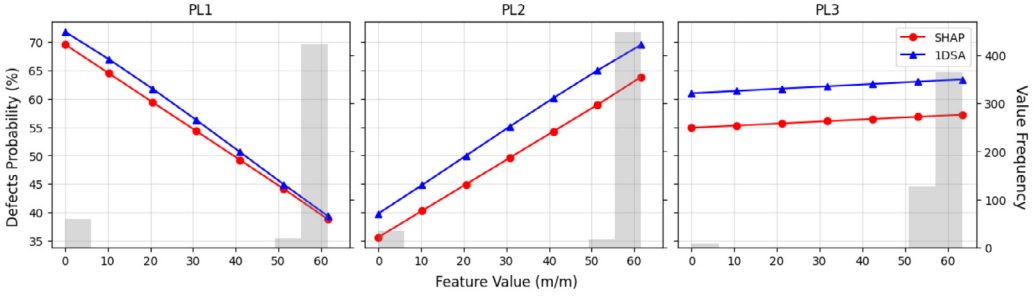
(a) Influence of Quantity of Decorated Particleboard to be Produced (QTY)



(b) Influence of Top Paper Resin Weight (TPRW)



(c) Influence of Top Paper Humidity Weight (TPHW)



(d) Influence of Bottom Paper Resin Content (BPRC)

Fig. 6. Influence of features (a) QTY, (b) TPRW, (c) TPHW and (d) BPRC for Production Lines (PL) 1, 2 and 3 for predicting defects.

with discrepancies limited to one classification level. In contrast, the features TPVC and BPVC, deemed "Very High" (VH) by the operators, were frequently categorized as "Low" (L) by the predictive models.

The qualitative analysis presented in Table 11 indicates that most actionable features (6 out of 9) exert a high to very high impact on predictive results, namely TPRC, BPRC, TEMPUP, TEMPDW, THERMCT and CYCHOUR. This implies that operators have some ability to influence production outcomes by modulating these input features to obtain a desired predictive outcome. For instance, as shown in left of Fig. 6(d), increasing the value of BPRC for PL1 decreases the probability of defects.

Finally, regarding the MAE values for the qualitative XAI comparison (last row from Table 11), we note a close alignment between XAI-derived and operators-derived rankings, ranging from 1.15 to 1.47 (average difference of approximately one level). When comparing the two XAI methods, SHAP demonstrated slightly better alignment for halts, particularly for PL3 (MAE = 1.15), while 1DSA showed slightly improved alignment for defects (e.g. MAE = 1.19 for PL1).

5. Conclusions

This paper presents a Machine Learning (ML) approach for predicting particleboard production halts and defects within a multi-line Industry 4.0 environment. Our contributions are threefold: first, we focus on the planning phase of production, using only pre-production input features, diverging from the predominant related work that emphasize

Table 11
Comparison of LogR feature importance with qualitative feature importances for each predictive task, XAI method and production line (**bold** denotes the best MAE values per predictive task, XAI method and production line).

Attribute	Qualitative feature importance	LogR feature importance											
		Defects						Halts					
		1DSA			SHAP			1DSA			SHAP		
		PL1	PL2	PL3	PL1	PL2	PL3	PL1	PL2	PL3	PL1	PL2	PL3
QTY	VH	VH	VH	VH	VH	VH	VH	VH	VH	VH	VH	VH	VH
DPCC	M	M	M	M	M	M	M	H	M	M	H	M	H
DPL	L	H	VH	VH	H	H	H	VH	VH	VH	VH	H	VH
DPW	L	M	M	H	H	H	VH	H	H	H	H	VH	VH
DPT	M	H	VH	VH	VH	VH	VH	H	H	VH	VH	H	H
DPTF	H	M	M	L	M	M	L	L	L	L	L	L	L
DPBF	H	L	M	L	L	L	L	L	L	L	L	L	L
PCC	M	L	M	L	L	M	L	M	H	M	M	H	H
PF	M	M	L	M	L	L	M	M	L	L	L	L	M
TPCC	H	L	L	M	L	M	M	L	L	M	L	M	M
TPRC	VH	VH	VH	H	H	VH	M	VH	VH	M	VH	VH	VH
TPVC	VH	L	L	L	L	L	L	M	L	L	L	L	L
TPMP	H	M	L	L	M	L	L	L	M	L	M	M	L
TPRW	M	VH	VH	VH	VH	VH	VH	VH	H	H	H	H	H
TPHW	M	VH	VH	VH	VH	VH	VH	VH	H	H	H	H	H
BPCC	H	L	L	M	M	M	M	L	L	L	L	M	M
BPRC	VH	VH	VH	VH	VH	VH	H	VH	VH	VH	VH	VH	H
BPVC	VH	L	M	L	L	L	L	M	L	M	M	L	L
BPMP	H	M	L	L	M	L	L	L	M	L	M	L	L
BPRW	M	H	H	H	H	H	H	VH	VH	H	VH	VH	M
BPHW	M	H	H	H	H	VH	H	VH	VH	H	VH	VH	M
TEMPUP	VH	VH	H	H	VH	H	H	M	M	VH	M	M	VH
TEMPDW	VH	H	H	H	H	H	VH	M	M	M	M	M	VH
MECHCT	M	L	L	M	L	L	M	L	M	H	L	L	M
THERMCT	VH	H	H	M	M	M	H	H	H	VH	H	H	VH
CYCHOUR	M	VH	H	VH	VH	H	VH	H	VH	VH	H	VH	VH
MAE		**1.19**	**1.27**	**1.42**	1.35	1.35	**1.42**	1.46	**1.42**	1.35	**1.42**	**1.42**	**1.15**

L - Low; M - Medium; H - High; VH - Very High.

the execution and inspection phases. Second, we develop a comprehensive empirical ML framework that evaluates the performance of Single-Task Learning (STL) versus Multi-Task Learning (MTL) and Line-Specific Modeling (LSM) versus Line-Agnostic Modeling (LAM). Third, we enhanced model transparency and trustworthiness by incorporating eXplainable Artificial Intelligence (XAI) methods with both quantitative and qualitative analyses.

The ML experiments compared Logistic Regression (LogR) and three Automated Machine Learning (AutoML) algorithms: a Bayesian-optimized Deep Feedforward Network (DFFN) and the H2O and Ludwig AutoML tools. For the prediction of defects in the STL approach, LogR, H2O and DFFN demonstrated similar performance, while Ludwig performed poorly in all production lines. For halts, LogR performed strongly overall, surpassed only by H2O in PL1. In MTL experiments, DFFN slightly outperformed Ludwig in most scenarios. However, STL consistently achieved better results than MTL, probably because of the influence of distinct input feature rankings for defects and halts, making it the preferred approach as it enables models to specialize in target-specific patterns.

Among the ML algorithms tested, LogR demonstrated strong predictive performance and computational efficiency, making it the most suitable choice for further experimentation. In the LSM versus LAM experiments, the results showed a slight preference for LAM for defects and a stronger preference for LSM for halts. The overall effectiveness of LSM led to its selection, resulting in six LogR models tailored to each production line and predictive task.

To enhance transparency and operational trust, we applied SHapley Additive exPlanations (SHAP) and One-Dimensional Sensitivity Analysis (1DSA) for feature importance ranking and compared their output using the Normalized Kendall Tau Distance (NKTD) ranking metric. Both methods produced consistent results, and their rankings aligned well with qualitative insights from plant operators, differing, on average, by just one level on a four-tier relevance scale. Given its lower computational cost, 1DSA is recommended as a lightweight but effective alternative to SHAP in similar production planning scenarios typical of Industry 4.0 environments.

In terms of research and practical implications, this work demonstrates that the proposed predictive models (LogR algorithm and LSM approach) are computationally fast and potentially valuable for particleboard production planning, given that only pre-production inputs were used and that the models achieved reasonable (70%) to very good (82%) predictive performance. Furthermore, the proposed quantitative and qualitative XAI analysis (e.g. NKTD measures) can be easily adapted to other studies in the Industry 4.0 domain (e.g. textile sector). Regarding the XAI findings, we confirmed that several of the most influential features align with the particleboard production domain knowledge and are actionable (e.g., Top Paper Resin Content - TPRC, Top Plate Temperature - TEMPUP).

In terms of limitations, although our evaluation followed a realistic simulation setup, using a Rolling Window (RW) strategy to mimic the deployment of ML in a real-world setting, all experiments were conducted offline after collecting the full set of particleboard production data. Consequently, while the empirical results demonstrate strong predictive potential, the proposed models and their associated explainability outputs have not yet been tested for their practical impact on actual production planning decisions. Another limitation is that we primarily relied on a single criterion to measure predictive performance (AUC of the ROC curve). Furthermore, while qualitative XAI validation with domain experts supports the relevance of the identified input features, it is important to recognize that expert assessments may be influenced by subjective perceptions, biases, or incomplete knowledge.

In terms of future work, a real-time alert system is currently being designed to notify operators of imminent defects or halt-prone conditions, allowing proactive interventions. Moreover, in conjunction with the domain experts, we are currently assessing the benefits and costs associated with correct predictions and misclassifications to

automatically select the best decision thresholds (e.g., using validation data). Also, we are implementing a multi-objective optimization framework based on evolutionary metaheuristics that leverages the predictive models as surrogates for the physical process to recommend optimized particleboard recipes (i.e., actionable pre-production inputs) for upcoming batches. Both developments are being integrated into an Intelligent Decision Support System (IDSS) tailored for particleboard manufacturing, allowing real-time interaction with production planners and supporting continuous feedback and model refinement. Furthermore, we are exploring the use of the IDSS in a dynamic scheduling application that adapts resource allocation based on predictive insights, in order to minimize downtime and improve operational efficiency.

CRediT authorship contribution statement

Arthur Matta: Conceptualization, Formal analysis, Investigation, Methodology, Project administration, Software, Validation, Visualization, Writing – original draft, Writing – review & editing. **Luís Miguel Matos:** Conceptualization, Formal analysis, Investigation, Software, Validation, Writing – original draft, Writing – review & editing. **Jorge Miguel Silva:** Conceptualization, Data curation, Resources, Supervision, Validation. **Miguel Bastos Gomes:** Conceptualization, Data curation, Resources, Supervision, Validation. **André Pilastri:** Conceptualization, Supervision, Validation, Writing – review & editing. **Paulo Cortez:** Conceptualization, Formal analysis, Funding acquisition, Investigation, Methodology, Supervision, Validation, Visualization, Writing – original draft, Writing – review & editing.

Funding sources

This work has been supported by the European Union under the Next Generation EU, through a grant of the Portuguese Republic's Recovery and Resilience Plan (RRP) Partnership Agreement, within the scope of the project PRODUTECH R3 – "Agenda Mobilizadora da Fileira das Tecnologias de Produção para a Reindustrialização", Total project investment: 166,988,013.71 Euros; Total Grant : 97,111,730.27 Euros.

Declaration of competing interest

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

Acknowledgments

The authors wish to thank the planners and operators of the particleboard production company for their helpful feedback on the ML and XAI results. Finally, we wish to thank the anonymous reviewers for their helpful comments.

Appendix A. List of acronyms

See Table A.1.

Appendix B. Automl selected models and hyperparameters

See Tables B.1–B.12.

Data availability

The authors do not have permission to share data.

Table A.1
List of Acronyms used in this manuscript.

Acronym	Description
1DSA	One-Dimensional Sensitivity Analysis
AE	AutoEncoder
ANN	Artificial Neural Network
AoT	Ahead-of-Time
AUC	Area Under the Curve
AutoML	Automated Machine Learning
BPCC	Bottom Paper Characteristic Code
BPHW	Bottom Paper Humidity Weight
BPMP	Bottom Paper Manufacturing Process
BPRC	Bottom Paper Resin Content
BPRW	Bottom Paper Resin Weight
BPVC	Bottom Paper Volatile Content
CNN	Convolutional Neural Network
CYCHOUR	Recipe Value for the Number of Press Cycles per Hour
DFFN	Deep FeedForward Network
DPBF	Decorated Particleboard Bottom Finishing
DPCC	Decorated Particleboard Characteristic Code
DPL	Decorated Particleboard Width
DPT	Decorated Particleboard Thickness
DPTF	Decorated Particleboard Top Finishing
DPW	Decorated Particleboard Width
DRF	Distributed Random Forest
DT	Decision Tree
ECD	Encoder-Combiner-Decoder
FNR	False Negative Rate
FPR	False Positive Rate
GBM	Gradient Boosted Machines
GLM	Generalized Linear Model
ICE	Individual Conditional Expectation
IDF	Inverse Document Frequency
IDSS	Intelligent Decision Support System
IF	Isolation Forest
IoT	Internet of Things
IQR	InterQuartile Range
KNN	K-Nearest Neighbors
LAM	Line-Agnostic Modeling
LIME	Local Interpretable Model-agnostic Explanations
LinR	Linear Regression
LGBM	Light Gradient Boosting Machine
LogR	Logistic Regression
LSM	Line-Specific Modeling
LSTM	Long Short Term Memory
MAE	Mean Absolute Error
MECHCT	Recipe Value for the Mechanical Cycle
MES	Manufacturing Execution System
ML	Machine Learning
MTL	Multi-Task Learning
NB	Naive Bayes
NKTD	Normalized Kendall Tau Distance
OHE	One-Hot Encoding
PCC	Particleboard Characteristic Code
PDP	Partial Dependence Plots
PF	Particleboard Finishing
PL{1,2,3}	Production Line {1, 2, 3}
QTY	Quantity of Decorated Particleboard to be Produced
R&D	Research and Development
RF	Random Forest
RNN	Recurrent Neural Network
ROC	Receiver Operating Characteristic
RW	Rolling Window
SHAP	SHapley Additive exPlanations
STL	Single-Task Learning
SVM	Support Vector Machine
TEMPDW	Recipe Value for the Press' Bottom Plate Temperature
TEMPUP	Recipe Value for the Press' Top Plate Temperature
THERMCT	Recipe Value for the Thermal Cycle
TPCC	Top Paper Characteristic Code
TPHW	Top Paper Humidity Weight
TPMP	Top Paper Manufacturing Process
TPR	True Positive Rate
TPRC	Top Paper Resin Content
TPRW	Top Paper Resin Weight

(*continued on next page*)

Table A.1 (*continued*).

Acronym	Description
TPVC	Top Paper Volatile Content
VEC	Variable Effect Characteristic
XAI	eXplainable Artificial Intelligence
XGBoost	Extreme Gradient Boosting

Table B.1
H2O selected models for each modeling approach and predictive task.

Modeling approach	Production line	Predictive task	
		Defects	Halts
LSM	PL1	XGBoost	XGBoost
	PL2	GBM	GLM
	PL3	GBM	GBM
LAM	–	GBM	XGBoost

Table B.2
H2O hyperparameters and searchable values for the GLM model.

ρ	Description	Searchable values
alpha	Regularization distribution between L1 and L2	0.0 to, 1.0

Table B.3
H2O hyperparameters and searchable values for the GBM model.

ρ	Description	Searchable values
col_sample_rate	Column sample rate	0.4, 0.7, 1.0
col_sample_rate_per_tree	Column sample rate per tree	0.4, 0.7, 1.0
learn_rate	Learning rate	Hard coded: 0.1
max_depth	Maximum tree depth	3 to 17
min_rows	Minimum number of observations for a leaf	1, 5, 10, 15, 30, 100
min_split_improvement	Minimum relative improvement in squared error reduction in order for a split to happen	$1e^{-4}$, $1e^{-5}$
ntrees	Number of trees	Hard coded: 10,000[a]
sample_rate	Row sampling ratio of the training instance	0.5 to 1.0

[a] True value found by early stopping.

Table B.4
H2O hyperparameters and searchable values for the XGBoost model.

ρ	Description	Searchable values
booster	Booster type	'gbtree', 'dart'
col_sample_rate	Column sample rate	0.6, 0.8, 1.0
col_sample_rate_per_tree	Column sample rate per tree	0.7, 0.8, 0.9, 1.0
max_depth	Maximum tree depth	5, 10, 15, 20
min_rows	Minimum number of observations for a leaf	0.01, 0.1, 1.0, 3.0, 5.0, 10.0, 15.0, 20.0
ntrees	Number of trees	Hard coded: 10,000[a]
reg_alpha	Value for L1 regularization	0.001, 0.01, 0.1, 1, 10, 100
reg_lambda	value for L2 regularization	0.001, 0.01, 0.1, 0.5, 1
sample_rate	Row sampling ratio of the training instance	0.6, 0.8, 1.0

[a] True value found by early stopping.

Table B.5
H2O XGBoost hyperparameters values for PL1.

Hyperparameter (ρ)	Modeling task	
	Defects	Halts
booster	dart	gbtree
col_sample_rate	1.0	0.6
col_sample_rate_per_tree	0.8	1.0
max_depth	12	9
min_rows	20	3
ntrees	300	35
reg_alpha	0.01	0.5
reg_lambda	100	100
sample_rate	0.8	0.8

Table B.6
H2O GBM and GLM hyperparameters values for PL2.

Modeling task	Model	Hyperparameter (ρ)	Value
Defects	GBM	col_sample_rate	0.4
		col_sample_rate_per_tree	1.0
		learn_rate	0.1
		max_depth	12
		min_rows	100.0
		min_split_improvement	$1e^{-4}$
		ntrees	34
		sample_rate	0.6
Halts	GLM	alpha	0.0

Table B.7
H2O GBM hyperparameters values for PL3.

Hyperparameter (ρ)	Modeling task	
	Defects	Halts
col_sample_rate	1.0	0.7
col_sample_rate_per_tree	1.0	1.0
learn_rate	0.1	0.1
max_depth	16	3
min_rows	100	10
min_split_improvement	1e−4	1e−4
ntrees	35	31
sample_rate	0.5	0.5

Table B.8
Deep Feedforward Neural Network (DFFN) hyperparameters.

ρ	Description	Searchable values	Selected values
N_{layers}	Number of hidden layers in the network	1 to 7	4
HU_1	Number of units of the first hidden layer	1 to 1024	$HU_1 = 727$
HU_i	Number of units of the ith subsequent hidden layers	1 to $1.2 \cdot HU_{i-1}$	$HU_2 = 747$ $HU_3 = 511$ $HU_4 = 256$
HA_i	Activation Function in the ith subsequent hidden layers	'linear', 'ReLu', 'tanh'	$HA_1 = ReLu$ $HA_2 = ReLu$ $HA_3 = linear$ $HA_4 = ReLu$
D_i	Whether to include a dropout layer after the ith subsequent hidden layers	False, True	$D_1 = False$ $D_2 = False$ $D_3 = False$ $D_4 = True$
DV_i	Dropout rate to use in the ith dropout layer	0.1 to 0.5	$DV_1 = 0$ $DV_2 = 0$ $DV_3 = 0$ $DV_4 = 0.22$
BN_i	Whether to include a Batch Normalization layer after the ith subsequent hidden layers	False, True	$BN_1 = False$ $BN_2 = True$ $BN_3 = True$ $BN_4 = False$

Table B.9
Ludwig hyperparameters.

ρ	Description	Searchable values
size	Size of the hidden layers	8, 16, 24, 32, 64
output_size	Output size of a fully connected layer	8, 16, 24, 32, 64, 128
num_steps	Number of steps/repetitions of the attentive transformer and feature transformer computations	3, 4, 5, 6, 7, 8, 9, 10
bn_momentum	Momentum of the batch norm	0.4, 0.3, 0.2, 0.1, 0.05, 0.02
bn_virtual_bs	Size of the virtual batch size used by ghost batch norm	256, 512, 1024, 2048, 4096
sparsity	Multiplier of the sparsity inducing loss	0.0, 1e−06, 0.0001, 0.001, 0.01, 0.1
relaxation_factor	Factor that influences how many times a feature should be used across the steps of computation	1.0, 1.2, 1.5, 2.0
learning_rate	Controls how much to change the model in response to the estimated error each time the model weights are updated	2e-05 to 0.001 (loguniform)
decay_rate	Decay per epoch (%)	0.8, 0.9, 0.95
decay_steps	The number of steps to take in the exponential learning rate decay	500, 2000, 8000, 10000, 20000

Table B.10
Ludwig hyperparameter values for PL1.

Hyperparameter (ρ)	Modeling task		
	Defects	Halts	MTL
size	32	32	32
output_size	8	128	8
num_steps	4	9	4
bn_momentum	0.3	0.2	0.3
bn_virtual_bs	2048	256	2048
sparsity	0.0001	0	0.0001
relaxation_factor	1	1	1
learning_rate	7.997e−04	1.745e−04	7.997e−04
decay_rate	0.8	0.95	0.8
decay_steps	20 000	500	20 000

Table B.11
Ludwig hyperparameter values for PL2.

Hyperparameter (ρ)	Modeling task		
	Defects	Halts	MTL
size	32	8	32
output_size	8	32	8
num_steps	4	5	4
bn_momentum	0.3	0.4	0.3
bn_virtual_bs	2048	512	2048
sparsity	0.0001	0	0.0001
relaxation_factor	1	2	1
learning_rate	7.997e−04	1.445e−04	7.997e−04
decay_rate	0.8	0.9	0.8
decay_steps	20 000	10 000	20 000

Table B.12
Ludwig hyperparameter values for PL3.

Hyperparameter (ρ)	Modeling task		
	Defects	Halts	MTL
size	32	64	32
output_size	8	16	8
num_steps	4	3	4
bn_momentum	0.3	0.2	0.3
bn_virtual_bs	2048	256	2048
sparsity	0.0001	0.001	0.0001
relaxation_factor	1	1.2	1
learning_rate	7.997e−04	3.595e−04	7.997e−04
decay_rate	0.8	0.8	0.8
decay_steps	20 000	500	20 000

References

[1] X. Xu, Y. Lu, B. Vogel-Heuser, L. Wang, Industry 4.0 and industry 5.0—Inception, conception and perception, J. Manuf. Syst. 61 (2021) 530–535, http://dx.doi.org/10.1016/j.jmsy.2021.10.006.

[2] D.S. Satwaliya, H.P. Thethi, A. Dhyani, G.R. Kiran, M. Al-Taee, M.B. Alazzam, Predictive maintenance using machine learning: A case study in manufacturing management, in: Proceedings of the 3rd International Conference on Advance Computing and Innovative Technologies in Engineering, ICACITE, IEEE, 2023, pp. 872–876, http://dx.doi.org/10.1109/ICACITE57410.2023.10183012.

[3] S. Gawde, S. Patil, S. Kumar, P. Kamat, K. Kotecha, An explainable predictive maintenance strategy for multi-fault diagnosis of rotating machines using multi-sensor data fusion, Decis. Anal. J. 10 (2024) http://dx.doi.org/10.1016/j.dajour.2024.100425.

[4] J. Yang, Z. Liu, A novel real-time steel surface defect detection method with enhanced feature extraction and adaptive fusion, Eng. Appl. Artif. Intell. 138 (A) (2024) http://dx.doi.org/10.1016/j.engappai.2024.109289.

[5] R.M.A. Oliveira, Â.M.O. Sant'Anna, P.H.F. da Silva, Explainable machine learning models for defects detection in industrial processes, Comput. Ind. Eng. 192 (2024) 110214, http://dx.doi.org/10.1016/j.cie.2024.110214.

[6] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay, Scikit-learn: Machine learning in python, J. Mach. Learn. Res. 12 (2011) 2825–2830.

[7] D. Cook, Practical machine learning with H2O: powerful, scalable techniques for deep learning and AI, O'Reilly Media, Inc., 2016.

[8] P. Molino, Y. Dudin, S.S. Miryala, Ludwig: a type-based declarative deep learning toolbox, CoRR (2019) http://dx.doi.org/10.48550/arXiv.1909.07930, arXiv:1909.07930.

[9] I. Goodfellow, Y. Bengio, A. Courville, Deep learning, Adaptive Computation and Machine Learning series, MIT Press, 2016.

[10] T. Akiba, S. Sano, T. Yanase, T. Ohta, M. Koyama, Optuna: A next-generation hyperparameter optimization framework, in: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD '19, Association for Computing Machinery, 2019, pp. 2623–2631, http://dx.doi.org/10.1145/3292500.3330701.

[11] S.M. Lundberg, S.-I. Lee, A unified approach to interpreting model predictions, in: Advances in Neural Information Processing Systems, 30, 2017, pp. 4765–4774.

[12] E. Štrumbelj, I. Kononenko, Explaining prediction models and individual predictions with feature contributions, Knowl. Inf. Syst. 41 (3) (2014) 647–665, http://dx.doi.org/10.1007/s10115-013-0679-x.

[13] P. Cortez, M.J. Embrechts, Using sensitivity analysis and visualization techniques to open black box data mining models, Inform. Sci. 225 (2013) 1–17, http://dx.doi.org/10.1016/j.ins.2012.10.039.

[14] B. Iooss, A. Saltelli, Introduction to sensitivity analysis, Handbook of Uncertainty Quantification, Springer, 2015, http://dx.doi.org/10.1007/978-3-319-11259-6_31-1.

[15] M.G. Kendall, A new measure of rank correlation, Biometrika 30 (1/2) (1938) 81–93, http://dx.doi.org/10.1093/biomet/30.1-2.81.

[16] A. Matta, L.M. Matos, A. Pilastri, J.M. Silva, M.B. Gomes, P. Cortez, Ahead of time prediction of decorated particleboard production disruptions and defects using single and multi-target AutoML, Procedia Comput. Sci. 246 (2024) 2110–2119, http://dx.doi.org/10.1016/j.procs.2024.09.633.

[17] A. Baghbanpourasl, D. Kirchberger, C. Eitzinger, Failure prediction through a model-driven machine learning method, in: Proceedings of the 2021 IEEE International Workshop on Metrology for Industry 4.0 & IoT (MetroInd4.0&IoT), IEEE, 2021, pp. 527–531, http://dx.doi.org/10.1109/MetroInd4.0IoT51437.2021.9488550.

[18] M.G. Mohammadi, D. Mahmoud, M. Elbestawi, On the application of machine learning for defect detection in L-PBF additive manufacturing, Opt. Laser Technol. 143 (2021) 107338, http://dx.doi.org/10.1016/j.optlastec.2021.107338.

[19] R.C. Dias, P.P. Senna, A.F. Gonçalves, J. Reis, N. Michalaros, K. Alexopoulos, M. Gomes, PREFAB framework - product quality towards zero defects for melamine surface boards industry, IFAC- Pap. 54 (1) (2021) 570–575, http://dx.doi.org/10.1016/j.ifacol.2021.08.065.

[20] A. Saadallah, J. Büscher, O. Abdulaaty, T. Panusch, J. Deuse, K. Morik, Explainable predictive quality inspection using deep learning in electronics manufacturing, Procedia CIRP 107 (2022) 594–599, http://dx.doi.org/10.1016/j.procir.2022.05.031.

[21] J. Wang, W. Zhou, X. ying Ren, M. ming Su, J. Liu, A waveform-based clustering and machine learning method for damage mode identification in CFRP laminates, Compos. Struct. 312 (2023) 116875, http://dx.doi.org/10.1016/j.compstruct.2023.116875.

[22] D. Nikolova, P. Lameski, I.M. Pires, E. Zdravevski, Towards industry 4.0: Machine malfunction prediction based on iIoT streaming data, in: Proceedings of the 18th Conference on Computer Science and Intelligence Systems, in: Annals of Computer Science and Information Systems (ACSIS), 35, Polish Information Processing Society, 2023, pp. 291–296, http://dx.doi.org/10.15439/2023F677.

[23] H. V., S. K., Machine learning techniques for quality control in textile fabric manufacturing, in: International Conference on Sustainable Computing and Smart Systems, ICSCSS, 2024, pp. 902–907, http://dx.doi.org/10.1109/ICSCSS60660.2024.10625436.

[24] H. Zhang, Y. Wang, C. Yu, Research on key technology of online detection for particleboard, in: Proceedings of the 2021 IEEE International Conference on Electronic Technology, Communication and Information, ICETCI, IEEE, 2021, pp. 512–515, http://dx.doi.org/10.1109/ICETCI53161.2021.9563486.

[25] Y.M. Tang, A.W.H. Ip, W. Li, Artificial intelligence approach for aerospace defect detection using single-shot multibox detector network in phased array ultrasonic, in: IoT and Spacecraft Informatics, Elsevier, 2022, pp. 1–27, http://dx.doi.org/10.1016/B978-0-12-821051-2.00008-8.

[26] J. Takalo-Mattila, M. Heiskanen, V. Kyllönen, L. Määttä, A. Bogdanoff, Explainable steel quality prediction system based on gradient boosting decision trees, IEEE Access 10 (2022) 68099–68110, http://dx.doi.org/10.1109/ACCESS.2022.3185607.

[27] B. Yang, Y. Zhang, S. Wang, W. Xu, M. Xiao, Y. He, F. Mo, A global interactive attention-based lightweight denoising network for locating internal defects of CFRP laminates, Eng. Appl. Artif. Intell. 116 (2022) 105436, http://dx.doi.org/10.1016/j.engappai.2022.105436.

[28] N. Bharot, M. Soderi, P. Verma, J.G. Breslin, Improving product quality control in smart manufacturing through transfer learning-based fault detection, in: Proceedings of the 2023 IEEE International Conference on Smart Computing, SMARTCOMP, IEEE, 2023, pp. 213–215, http://dx.doi.org/10.1109/SMARTCOMP58114.2023.00051.

[29] T. Özel, D. Malekar, S. Aniyambeth, P. Li, Physics-informed machine learning for defect identification in fused filament fabrication additive manufacturing, Procedia CIRP 118 (2023) 723–728, http://dx.doi.org/10.1016/j.procir.2023.06.124.

[30] B. Yang, W. Xu, F. Bi, Y. Zhang, L. Kang, L. Yi, Multi-scale neighborhood query graph convolutional network for multi-defect location in CFRP laminates, Comput. Ind. 153 (2023) 104015, http://dx.doi.org/10.1016/j.compind.2023.104015.

[31] J. Horňas, J. Běhal, P. Homola, R. Doubrava, M. Holzleitner, S. Senck, A machine learning based approach with an augmented dataset for fatigue life prediction of additively manufactured Ti-6Al-4V samples, Eng. Fract. Mech. 293 (2023) 109709, http://dx.doi.org/10.1016/j.engfracmech.2023.109709.

[32] A.M. Dharwa, A.K. Yadav, Dhiraj, V. Maan, K.S. Sangwan, Steel surface defect detection using machine learning techniques, in: 2024 First International Conference on Electronics, Communication and Signal Processing, ICECSP, 2024, pp. 1–6, http://dx.doi.org/10.1109/ICECSP61809.2024.10698084.

[33] S. Chen, S. Jiang, X. Wang, P. Sun, C. Hua, J. Sun, An efficient detector for detecting surface defects on cold-rolled steel strips, Eng. Appl. Artif. Intell. 138 (A) (2024) http://dx.doi.org/10.1016/j.engappai.2024.109325.

[34] D. Xu, Z. Zhang, J. Shi, Failure prediction using gated recurrent unit and autoencoder in complex manufacturing process, in: Proceedings of the 4th World Symposium on Artificial Intelligence, WSAI, IEEE, 2022, pp. 68–73, http://dx.doi.org/10.1109/WSAI55384.2022.9836412.

[35] L. Wang, W. Zhang, Q. Bao, Q. Wang, XGBoost-based failure prediction method for metal additive manufacturing equipment, in: Proceedings of the 34th Chinese Control and Decision Conference, CCDC, IEEE, 2022, pp. 3059–3064, http://dx.doi.org/10.1109/CCDC55256.2022.10034052.

[36] D. Chowdhury, A. Sinha, D. Das, XAI-3DP: Diagnosis and understanding faults of 3-D printer with explainable ensemble AI, IEEE Sensors Lett. 7 (1) (2023) 1–4, http://dx.doi.org/10.1109/LSENS.2022.3228327.

[37] D. Pagano, A predictive maintenance model using long short-term memory neural networks and Bayesian inference, Decis. Anal. J. 6 (2023) http://dx.doi.org/10.1016/j.dajour.2023.100174.

[38] C. Afonso, A. Matta, L.M. Matos, M.B. Gomes, A. Santos, A. Pilastri, P. Cortez, Machine learning for predicting production disruptions in the wood-based panels industry: A demonstration case, Artif. Intell. Appl. Innov. (2023) http://dx.doi.org/10.1007/978-3-031-34107-6_27.

[39] Y. Wang, P. Wang, Explainable machine learning for motor fault diagnosis, in: 2023 IEEE International Instrumentation and Measurement Technology Conference (I2MTC), 2023, pp. 1–6, http://dx.doi.org/10.1109/I2MTC53148.2023.10175895.

[40] A. Hosseinzadeh, F.F. Chen, M. Shahin, H. Bouzary, A predictive maintenance approach in manufacturing systems via AI-based early failure detection, Manuf. Lett. 35 (2023) 1179–1186, http://dx.doi.org/10.1016/j.mfglet.2023.08.125.

[41] S. Gawde, S. Patil, S. Kumar, P. Kamat, K. Kotecha, S. Alfarhood, Explainable predictive maintenance of rotating machines using LIME, SHAP, PDP, ICE, IEEE Access 12 (2024) 29345–29361, http://dx.doi.org/10.1109/ACCESS.2024.3367110.

[42] K. Jovin, B.C. Ben, E. Lule, J. Kizito, N. Hellen, G. Marvin, Explainable machine learning techniques for predictive car engine health monitoring, in: 2nd International Conference on Sustainable Computing and Smart Systems, ICSCSS 2024, 2024, pp. 1209–1218, http://dx.doi.org/10.1109/ICSCSS60660.2024.10624883.

[43] A. Shaala, D. Baglee, D. Dixon, Machine learning model for predictive maintenance of modern manufacturing assets, in: International Conference on Automation and Computing, ICAC, 2024, pp. 1–6, http://dx.doi.org/10.1109/ICAC61394.2024.10718768.

[44] A.N. Zereen, A. Das, J. Uddin, Machine fault diagnosis using audio sensors data and explainable ai techniques-LIME and SHAP, Comput. Mater. Contin. 80 (3) (2024) 3463–3484, http://dx.doi.org/10.32604/cmc.2024.054886.

[45] M. Brown, J. Kros, Data mining and the impact of missing data, Ind. Manag. & Data Syst. 103 (8) (2003) 611–621.

[46] P.J. Pereira, P. Cortez, R. Mendes, Multi-objective grammatical evolution of decision trees for mobile marketing user conversion prediction, Expert Syst. Appl. 168 (2021) 114287, http://dx.doi.org/10.1016/J.ESWA.2020.114287.

[47] L.M. Matos, J. Azevedo, A. Matta, A.L. Pilastri, P. Cortez, R. Mendes, Categorical attribute transformation environment (CANE): A python module for categorical to numeric data preprocessing, Softw. Impacts 13 (2022) 100359, http://dx.doi.org/10.1016/J.SIMPA.2022.100359.

[48] T. Fawcett, An introduction to ROC analysis, Pattern Recognit. Lett. 27 (8) (2006) 861–874, http://dx.doi.org/10.1016/j.patrec.2005.10.010.

[49] J. Mockus, Bayesian approach to global optimization: Theory and applications, first ed., Mathematics and its Applications, vol. 37, Springer Dordrecht, 1989, http://dx.doi.org/10.1007/978-94-009-0909-0.

[50] L.M. Matos, P. Cortez, R. Mendes, A. Moreau, A deep learning-based decision support system for mobile performance marketing, Int. J. Inf. Technol. Decis. Mak. 22 (2) (2023) 679–703, http://dx.doi.org/10.1142/S021962202250047X.

[51] D.P. Kingma, J. Ba, Adam: A method for stochastic optimization, 2017, http://dx.doi.org/10.48550/arXiv.1412.6980, arXiv:1412.6980.

[52] S.O. Arik, T. Pfister, TabNet: Attentive interpretable tabular learning, 2020, http://dx.doi.org/10.48550/arXiv.1908.07442, arXiv:1908.07442.

[53] L.J. Tashman, Out-of-sample tests of forecasting accuracy: an analysis and review, Int. J. Forecast. 16 (4) (2000) 437–450, http://dx.doi.org/10.1016/S0169-2070(00)00065-0.

[54] M. Hollander, D.A. Wolfe, E. Chicken, Nonparametric statistical methods, third ed., Wiley Series in Probability and Statistics, John Wiley & Sons, Inc., 2013, http://dx.doi.org/10.1002/9781119196037.