Machine Learning for Predicting Production Disruptions in the Wood-Based Panels Industry: A Demonstration Case

Cláudia Afonso[1,2][0009−0002−4138−4541], Arthur Matta[1,2][0000−0002−4902−9483], Luís Miguel Matos[2][0000−0001−5827−9129], Miguel Bastos Gomes[3], Antonina Santos[3][0009−0005−7615−7025], André Pilastri[1][0000−0002−4380−3220], and Paulo Cortez[2][0000−0002−7991−2090]

[1] EPMQ, CCG/ZGDV Institute, Guimarães, Portugal
claudia.afonso@ccg.pt, arthur.matta@ccg.pt, andre.pilastri@ccg.pt
[2] ALGORITMI/LASI, Dep. Information Systems, University of Minho, Guimarães, Portugal
luis.matos@dsi.uminho.pt, pcortez@dsi.uminho.pt
[3] SONAE Arauco Portugal, S.A., Oliveira do Hospital, Portugal
miguel.gomes@sonaearauco.com, antonina.santos@sonaearauco.com

Abstract. In this paper, we study the application of Machine Learning (ML) in detecting and predicting Ahead-of-Time (AoT) production disruptions in a Portuguese Wood-Based Panels Industry. Assuming an Industry 4.0 concept, the analyzed ML classification task presents several challenges, such as a high number of Internet of Things (IoT) sensors, high-velocity data generation and extremely imbalanced data. To solve these issues, we adapt and compare five state-of-the-art ML algorithms for anomaly detection. Moreover, we pre-process the big data and employ a Selective Sampling (SS) technique to train and test computationally efficient ML models. Overall, high-quality results were obtained by an eXtreme Gradient Boosting (XGBoost) model, both in terms of detection and AoT prediction of production stoppages. Finally, we applied an eXplainable AI (XAI) technique based on sensitivity analysis to the XGBoost model, enabling the understanding of the impact of the sensor inputs on the disruption condition.

Keywords: Anomaly Detection · Industry 4.0 · Machine Learning · Ahead-of-Time Prediction.

1 Introduction

The Industry 4.0 paradigm is causing a transformation in diverse industries. Advanced Information Technologies (IT), such as the Internet of Things (IoT), Big Data, Artificial Intelligence (AI) and Machine Learning (ML), can enable more efficient and intelligent manufacturing [7, 15]. In particular, ML has emerged as a powerful tool in several Industry 4.0 applications, such as Product Quality Assessment, Predictive Maintenance (PdM) and Detection of Production Anomalies [14].

In this paper, we demonstrate the usefulness of an ML approach to detect and even predict Ahead-of-Time (AoT) Production Disruptions related to a Portuguese Wood-Based Panels Industry. The final goal is to create a digital twin to simulate the stoppages conditions and ultimately prevent their occurrence by employing a prior intelligent control of some relevant production variables (e.g., top and bottom plate temperatures of the press operation). The analyzed AoT binary classification task ("normal production", "stoppage") corresponds to an instance of the Machine Learning based Early Decision Making (ML-EDM) general problem, which is considered challenging [1]. Indeed, the Wood-Based Panels AoT production disruption task is nontrivial due to three main reasons. Firstly, it involves a large number of IoT sensors, resulting in hundreds of potential inputs. Second, the sensors generate data at a high velocity, creating big data that requires a prohibitive ML computational effort. For instance, the analyzed period of nine months corresponds to around 360 Gb of raw production data. Thirdly, the data is extremely imbalanced, with only a small fraction of the data records corresponding to stoppages (around 2%).

To handle the AoT stoppage detection task, we adapt and compare five state-of-the-art anomaly detection ML algorithms: unsupervised – Isolation Forest (IF) and deep Autoencoder (AE); and supervised – Random Forest (RF), eXtreme Gradient Boosting (XGBoost), and a Deep FeedForward Neural network (DFFN). The five algorithms are compared in terms of their computational effort and AoT predictive performance. For the ML experimentation, we create two production stoppage datasets: **Full2d** – corresponding to two days of full data and used in preliminary ML experiments; and **SS9m** – which assumes a novel Selective Sampling (SS) that is executed over the full nine-month big data, resulting in a much smaller dataset that allows to adequately train and test the ML algorithms while making a reasonable usage of computational resources. Within our knowledge, our approach is novel when compared with state-of-the-art works. For instance, some of the explored ML algorithms have been applied to predict production machine failures in the wood industry, such as executed in [3] (XGBoost and RF) and [16] (IF and AE). However, none of these works studied AoT stoppage prediction or employed a SS to handle big data. Moreover, we employ an eXplainable AI (XAI) technique, based on a sensitivity analysis [6], to the best AoT predictive model, which allows for demonstrating the impact of the adopted sensor inputs in the stoppage condition.

2 Materials and Methods

2.1 Industrial Data

The collected raw data assumed 221 IoT sensors installed on distinct machines of the wood-based panels' production line, recorded from May 2021 to February 2022. These sensors provide a continuous data stream on diverse aspects of the production line, including machine conditions, the goods being produced and the materials being used. To facilitate the analysis of the sensor data, we categorized them into 7 distinct groups based on their similarities, as shown in Table 1.

Table 1: Groups of IoT sensors with similar characteristics

Group	**Description**	**# sensors**	**# records**
Infeed	Board and paper infeed area (e.g., active station)	46	4,514,999
Press	Press area (e.g., temperature)	53	65,392,678
Cut	Cut area (e.g., intensity)	13	7,653,753
Edge Cleaner	Edge cleaner area (e.g., trimmer speed)	25	22,739,428
Stacking	Stacking area (e.g., elevation speed)	36	2,777,763
General	General information of the production order (e.g., shift)	48	15,510,539
Materials	Raw materials used (e.g., melamine code)	42	28,322,721

To differentiate between sensor readings captured during regular production and those captured during disruptions, we relied on the start and end timestamps of previous disruptions in the production line. The database accessed contained also a distinctive identifier for the specific area of the production line where the reading took place. It should be noted that distinct sensors have different data generation frequencies. For instance, some sensors produce multiple readings per minute while others only produce one value every 10 minutes. Thus, the raw data records are not produced at regular time intervals but only when there is at least one sensor reading. When such a reading occurs with a timestamp t, all previous unchanged sensor values are repeated, thus each data record contains all received sensor values at time t. On average, 3 records are produced for each second.

For ML experimentation purposes, we created two datasets: Full2d and SS9m. The first Full2d was aimed at a preliminary ML proof-of-concept, to test if the ML stoppage detection was possible and a computationally feasible task. Given the sheer volume of data, it only includes the first two days of raw production data, which corresponds to 448,058 records related to 5 production stoppages. In the preprocessing stage, all empty, constant, or categorical input variables that directly signaled a stoppage condition were removed, resulting in a total of 95 variables. Then, the remaining categorical inputs were transformed into numeric values by assuming a simple label encoding.

As for the second dataset (SS9m), it assumes a Selective Sampling (SS) that covers with more detail all stoppages and normal conditions rear these stoppages while retaining a realistic imbalanced disruption ratio of examples (Fig. 1). The rationale is to allow an AoT analysis closer to and during stoppage events rather than prolonged normal production stages. The SS method is applied for all

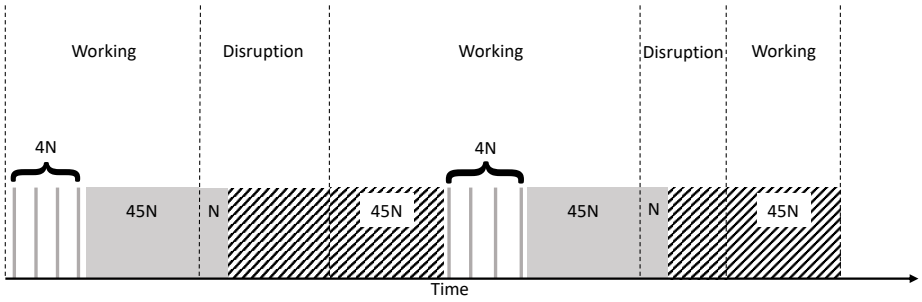


Fig. 1: Schematic of the proposed Selective Sampling (SS) approach.

production segments, where each segment is made of a normal working cycle that is ended by a disruption. Let $N = 5$ denote the number of consecutive records that occur at the beginning of a stoppage. Since the ratio of non-disruption to disruption records is R=49:1, we sample a total of $49 \times N = 245$ normal production examples that occur before a stoppage. Around $45 \times N = 225$ of these examples occur sequentially before the disruption event, thus resulting in a denser sampling that allows an AoT prediction analysis (shown as a gray area in Fig. 1). The other $4 \times N = 20$ normal instances (area with vertical gray lines) are sampled at larger and equally spaced intervals within the previous normal working zone of the production segment, with exception of the initial segment records (stripped area) that are ignored (total of $45 \times N$ instances). The excluded area (stripped zone in Fig. 1) includes a stoppage staled status (uninteresting data), a manual production reboot (which is not predictable), and its subsequent initial machine rebooting readings.

After consulting the wood-based panel production experts, the SS9m dataset was filtered to only include disruptions up to the press stage, since these correspond to the stoppages that have more impact on the well-functioning of the production process and some disruptions after the press stage are not even predictable. Thus, all data features related to sensors after the press area were excluded, which includes those in the "edge cleaner", "cut", and "stacking" groups. Also, all data records that occur after the press area were removed. Furthermore, a more in-depth data quality feature selection preprocessing was conducted by employing an exploratory data analysis. As the result of this analysis, features that met at least one of the following criteria were removed: contain reading interruptions over time (missing data); represent a theoretical or a predefined value, not a real measured one; describe another feature; remain nearly constant throughout the entire period; has no relevance to the study; have timestamps that do not accurately reflect the actual time of the readings; Or are highly correlated with another feature (correlation coefficient higher than 0.95). The final set of input variables included a total of 68 distinct sensor readings. Then, the known One-Hot (OH) encoding technique was applied to the categorical attributes, transforming each categorical level into a numeric input that represents

a boolean value (0 or 1) by using the Python package `CANE`[4] [12]. After preprocessing the data, the SS9m dataset includes a total of 984 numeric inputs and 649,161 records that are related to 68 IoT sensors and 1,107 production segments (sequences of sampled normal and stoppage conditions).

2.2 Machine Learning Methods

All ML algorithms were implemented by using the Python language and the following modules: `scikit-learn`[5] – for IF and RF; `TensorFlow`[6] – for AE and DFFN; and `Xgboost`[7] – for XGBoost classifier model. Unless stated otherwise, we adopt the default Python implementation values for the ML hyperparameters.

RF is a supervised ML algorithm that utilizes an ensemble of individual decision trees to form a prediction. Each decision tree is built using a random selection of input variables and a subset of training samples, known as bagging [2]. The RF algorithm is used to predict anomaly class probabilities, where the resulting values ($d_i \in [0.0, 1.0]$) represent the degree of anomaly present in the input data instance.

XGBoost is another ML tree ensemble that employs gradient boosting, assuming a loss function to evaluate the accuracy of the predictions made by each tree base learner. XGBoost has been demonstrated to outperform several other ML algorithms in diverse predictive modeling tasks [4].

The DFFN model is a deep learning architecture that includes several hyperparameters that are often set using heuristics and trial-and-error experiments [10]. In this paper, we adopt the dense network structure presented [13], which assumes a triangular-shaped multilayer perceptron, in which each subsequent layer size is smaller. The DFFN hyperparameters were tunned using preliminary experiments that were conducted using only the training data, from which 30% of the most recent values were used as the validation set. For both datasets, the DFFN includes a fixed structure with 8 hidden layers, with the following number of nodes: (I, 1024, 512, 256, 128, 64, 32, 16, 8, 1), where I denotes the number of inputs. The hidden nodes use the linear (Full2d) and Leaky ReLu (SS9m) activation functions, while the output node of both DFFNs computes the logistic function, to output the abnormal class probability. The popular Adam optimizer, with a batch size of 1024 and a binary cross-entropy loss function, was used to train the DFFNs. The training algorithm was stopped when the validation error does not improve or after a maximum of 100 epochs.

The Isolation Forest (IF) is a one-class ML algorithm, that utilizes the concept of isolation, where an anomaly is expected to be more isolated compared to normal data points [11]. The algorithm constructs a forest of decision trees. Each tree is grown by recursively partitioning the dataset into two random parts until the anomalies are isolated in their tree branches. The anomalies are isolated due to their inherent rarity and a high degree of difference from standard

[4] https://pypi.org/project/cane/
[5] https://scikit-learn.org/stable/
[6] https://www.tensorflow.org/
[7] https://xgboost.readthedocs.io/

data points. The `scikit-learn` IF implementation provides a decision score that ranges from $\hat{y}_i$ =-1 (highest abnormal score) to $\hat{y}_i$ =1 (highest normal score). This score was normalized such that the final disruption probability is set within the [0,1] range.

Autoencoders (AEs) use an unsupervised learning method that can efficiently compress data into a lower-dimensional representation [10]. In this paper, we assume the AE proposed in [9], which uses a deep dense multilayer perceptron with a bottleneck layer of L_b hidden nodes and that includes two components. The first component is a triangular-shaped encoder that starts with I inputs. Then, the number of hidden layer units decreases by half in each subsequent hidden layer, until three hidden layers are defined. For instance, for the SS9m dataset, the encoder includes the following number of layer nodes: (I =984, 492, 246, L_b =123). The second decoder component shape is symmetric to the encoder, ending up with I output nodes. When adapted for anomaly detection, the AE training algorithm is only fed with normal instances and the goal is to generate output values identical to the inputs. The same Adam optimizer used to train the supervised DFFN model is adopted to train the AE, with the exception that the Mean Absolute Error (MAE) is used as the loss function. The same MAE value is used as the reconstruction error. The higher the reconstruction score, the higher the anomaly class probability, thus the predictive MAE error for a new instance was normalized (using training data) within the [0,1] range. In preliminary experiments (using only training data), we compared two AE hidden function activations (ReLU and linear). The best results were obtained by the linear activation AEs, which were used in the ML comparison experiments.

To extract XAI knowledge from a trained ML model we adopt the computationally efficient one-Dimensional Sensitivity Analysis (1D-SA) method proposed in [6]. The method involves fixing all ML inputs at their average values, except for a target input that is ranged with L=7 distinct levels. The ML responses are stored and then the overall input relevance is computed based on the Average Absolute Deviation (AAD) measure applied to the sensitivity responses. The Variable Effect Characteristic (VEC) curves can also be produced, plotting the overall sensitivity analysis effect of an input response on the target output. This XAI method was implemented by using the `rminer` package of the R tool[8][5].

2.3 Evaluation

The predictive anomaly detection performance is based on the Receiver Operating Characteristic curve [8]. When a classifier outputs a decision score d_t at time t, the class can be interpreted as positive if $d_t > K$, where K is a fixed decision threshold, otherwise it is considered negative. With the class predictions, there will be True Positives (TP), True Negatives (TP), False Positives (FP) and False Negatives (FN). The ROC curve shows the performance of a two-class classifier across all $K \in [0, 1]$ values, plotting one minus the specificity (x-axis), or False Positive Rate (FPR), versus the sensitivity (y-axis), or True Positive

[8] https://CRAN.R-project.org/package=rminer

Rate (TPR). The discrimination performance is given by the Area Under the Curve (AUC): $AUC = \int_0^1 ROCdK$. It is common to interpret the quality of the AUC values as: 0.5 – equal to a random classifier; 0.6 – reasonable; 0.7 – good; 0.8 – very good; 0.9 – excellent; and 1 – perfect. We also record the computational effort, in terms of the total training time (in s) and prediction response time for all test instances (in s) when using a 2.4 GHz i9 Intel processor. Given that both datasets (Full2d and SS9m) are quite large, in all ML experiments we assume one execution of a time-ordered holdout split, where the oldest 70% records were used as the training set and the more recent 30% of the examples were used as the test set.

For the 9th month data (SS9m), we also perform an AoT analysis, where a distinct ROC curve is computed for different AoT prediction time values ($A \in \{0, 1, ..., A_{\max}\}$). Let y_t denote the target output values at time t, where t is a time-ordered example from the analyzed data (e.g., test set). The AoT analysis is performed for each production segment, by considering the target output values from the first normal event (y_{t_n}=0 at time t_n) of the segment until the last stoppage event (y_{t_s}=1 at time t_s). The goal is to only analyze sensor-based predictable normal or stoppage events since a production reboot (the return of a normal condition) is executed manually. When $A = 0$, a strict stoppage detection is performed, by comparing all predicted anomaly scores d_t from the production segment with the target output values (y_t). An AoT prediction occurs when $A > 0$, where the ROC curve is built by matching the d_{t-A} predictions with the same target output values (y_t). In this work, the AoT analysis is achieved by computing a proposed AoT Stoppage Prediction Graph (ASG). The ASG plot shows the evolution of the AUC values (y-axis) for the ROC curves associated with an increasing A value (x-axis). As mentioned in Section 2.1, on average there are around 3 data records per second. For the AoT analysis, we set $A_{\max}$=15, which corresponds to an average maximum AoT prediction of 5 s.

3 Results

Table 2 summarizes the results obtained by five ML algorithms on the two analyzed datasets, namely Full2d and SS9m. The stoppage detection (thus $A = 0$) metric is the AUC of the ROC curve, while the full training and test computational effort times are shown in seconds.

In terms of the Full2d data, the best detection was obtained by the supervised deep learning model (DFFN, with an AUC of 99%), followed by the unsupervised IF (94%) and then RF and AE (91%). However, these preliminary results should be analyzed with some caution, since the Full2d dataset only covers two days of data, with the training set containing only 3 stoppages and the test set 2 production disruption events. Thus, rather than ranking the ML algorithm results for this dataset, the obtained AUC values do show that there is a potential to detect wood-based panel production disruptions since they were substantially higher than a random classifier (AUC of 50%). As for the computational effort, the lighter training algorithm was the RF (2 s), while the faster prediction time

was obtained by XGBoost (0.1 s). And as expected, the deep learning algorithms (DFFN and AE) required substantially higher training times.

After obtaining the promising but initial Full2d results, we designed the SS and created the more representative SS9m, which covers all 1,107 stoppages that occur during nine months. When using this dataset, the AUC results favor the supervised learning methods, with the best result achieved by XGBoost (97%), followed by DFFN (94%) and RF (92%). In contrast, the unsupervised methods produce much worse results, closer to the random classifier performance (AE – 59%; IF – 48%). Regarding the computational effort, the ML training times are very reasonable. For instance, XGBoost requires only five minutes to fit a model when using around 454 thousand training samples that cover around 6 months of production time. As for the test time, it is much faster, only requiring around 2.5 s to produce around 195,000 predictions. This clearly attests to the usefulness of the proposed SS, which allows training some lightweight ML models that achieve a high-quality detection performance, with the test set including a realistic sample with around 330 stoppages.

Table 2: Stoppage detection results (best values in **bold**).

Dataset	Method	Model	Train Time (s)	Test Time (s)	AUC
Full2d	Supervised Learning	XGBoost	18.19	**0.10**	0.72
		RF	**2.02**	0.35	0.91
		DFFN	406.57	16.02	**0.99**
	Unsupervised Learning	AE	43.65	**4.46**	0.91
		IF	**5.04**	6.22	**0.94**
SS9m	Supervised Learning	XGBoost	315.55	**2.58**	**0.97**
		RF	**61.56**	3.28	0.92
		DFFN	266.55	26.71	0.94
	Unsupervised Learning	AE	**238.78**	**45.29**	**0.59**
		IF	355.63	55.31	0.48

Given the SS9m AUC results, an AoT analysis was further executed using only the supervised ML algorithms, which is shown by the ASG plotted in Table 2. For the full AoT time (around 5 s on average), all three ML algorithms are capable of producing interesting AoT AUC values. In particular, the XGBoost results are highlighted (blue curve), since it always produces the highest AUC values when compared with DFFN (second best model) and RF. Moreover, the XGBoost AUC values are of excellent quality, higher than 90% for most of the A range of values (e.g., up to A=10).

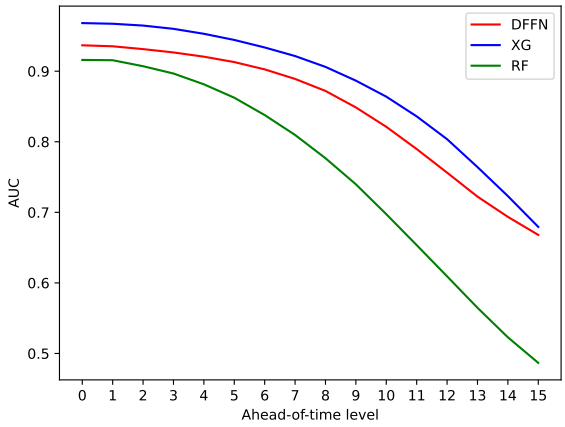
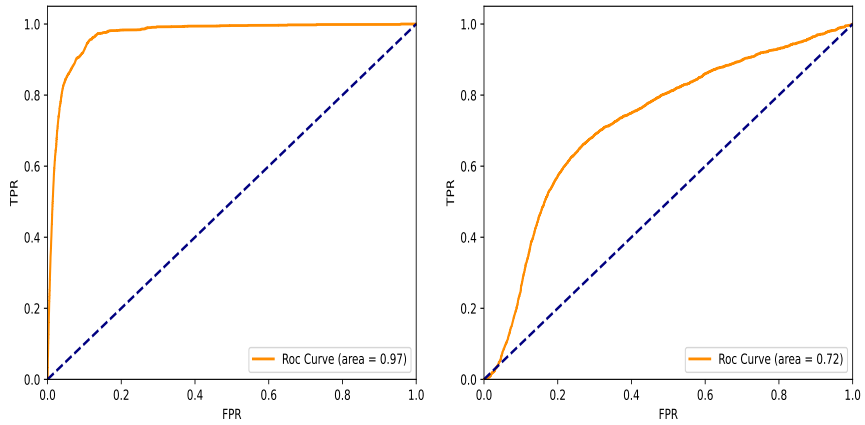


Fig. 2: AoT Stoppage Prediction Graph (ASG) for the supervised ML models.

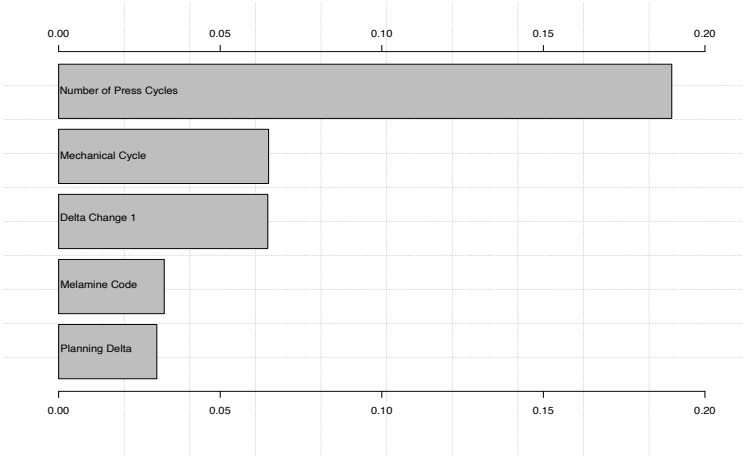
For demonstration purposes, Fig. 3 shows the individual ROC curves for the selected XGBoost model that were computed for pure detection ($A = 0$) and a large AoT prediction ($A = 15$).



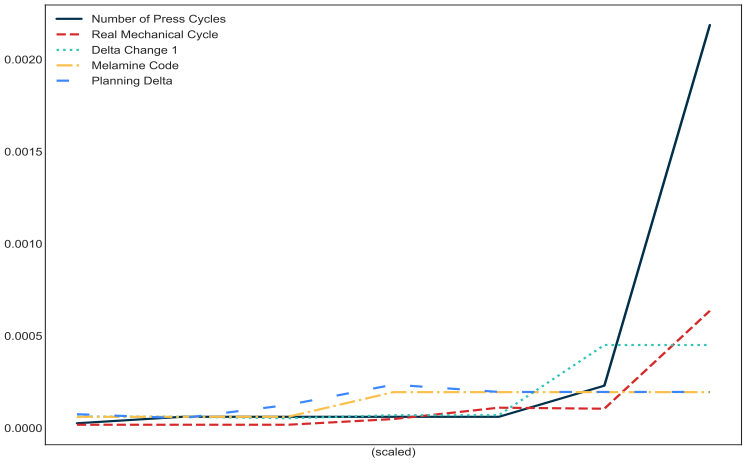
(a) $A = 0$. (b) $A = 15$.

Fig. 3: ROC curves for the XGBoost model and the two ASG extreme A values.

To further demonstrate the application domain value of the XGBoost AoT prediction model, we applied the SA XAI approach described in Section 2.2. Fig. 4(a) plots the obtained input importance for the five most relevant input features. For instance, the most influential input is related to the number of press cycles (total relevance of 19%). As for Fig. 4(b), it shows the Variable Effect Characteristic (VEC) curves for the same five top relevant inputs (the x-



(a) Importance values (in %) for the top five relevant inputs.



(b) Top 5 input VEC curves.

Fig. 4: Extracted XAI knowledge from the XGBoost model.

axis denotes the full range of input domain values, while the y-axis denotes the overall obtained prediction output). The plot reveals that the most influential input (Number of Press Cycles) produces the largest XGBoost output response change (thus impacting more on the model). It also confirms that some types of materials (melamine code) are more prone to produce stoppages. In general, an increase in the numeric input also produces an increase in the production line stoppage probability. The obtained XAI knowledge was also provided to the production experts, which confirmed that both input influence and input effects were interesting.

4 Conclusions

In this paper, we demonstrate the usefulness of using a Machine Learning (ML) approach to detect and predict Ahead-of-Time (AoT) production disruptions related to a Portuguese Wood-Based Panels Industry. We adapt and compare five state-of-the-art anomaly detection ML algorithms: unsupervised – Isolation Forest and deep Autoencoder; and supervised – Random Forest, XGBoost, and a Deep FeedForward Neural network. From the collected raw big data, we create and preprocess two production stoppage datasets: a shorter dataset including two days of full data (Full2d) and a Selective Sampling (SS) dataset comprising all interesting events during a nine months (SS9m).

First, preliminary experiments were executed using the Full2d data, showing that the proposed ML algorithms could provide interesting stoppage detection results. Then, more robust experimentation was performed by considering the SS9m dataset, which includes 1,107 production stoppages. The best results were obtained by the XGBoost, which produces a high-quality AoT stoppage prediction performance (ranging from 97% for a pure detection to 72% for an AoT of 5 s) under a reasonable computational effort usage. Finally, an eXplainable AI (XAI) approach based on a sensitivity analysis was applied to the XGBoost model, presenting the influence of the inputs in the disruption condition. The obtained results were shown to the Wood-Based Panels production managers, which provided positive feedback. Indeed, in future work, we plan to deploy the proposed ML model as a digital twin, to better monitor and even prevent production stoppages.

Acknowledgements

This work was supported by the European Structural and Investment Funds in the FEDER Component through the Operational Competitiveness and Internationalization Programme (COMPETE 2020) under Advanced Decision Making in productive systems through Intelligent Networks (ADM.IN) Project 055087 (POCI-01-0247-FEDER-055087).

References

1. Bondu, A., Achenchabe, Y., Bifet, A., Clérot, F., Cornuéjols, A., Gama, J., Hébrail, G., Lemaire, V., Marteau, P.: Open challenges for Machine Learning based Early Decision-Making research. SIGKDD Explorations **24**(2), 12–31 (2022). https://doi.org/10.1145/3575637.3575643
2. Breiman, L.: Random Forests. Mach. Learn. **45**(1), 5–32 (2001). https://doi.org/10.1023/A:1010933404324
3. Calabrese, M., Cimmino, M., Fiume, F., Manfrin, M., Romeo, L., Ceccacci, S., Paolanti, M., Toscano, G., Ciandrini, G., Carrotta, A., Mengoni, M., Frontoni, E., Kapetis, D.: SOPHIA: An Event-Based IoT and Machine Learning Architecture for Predictive Maintenance in Industry 4.0. Inf. **11**(4), 202 (2020). https://doi.org/10.3390/info11040202

4. Chen, T., Guestrin, C.: XGBoost: A scalable tree boosting system. Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining pp. 785–794 (2016). https://doi.org/10.1145/2939672.2939785
5. Cortez, P.: Data Mining with Neural Networks and Support Vector Machines Using the R/rminer Tool. In: Perner, P. (ed.) Advances in Data Mining. Applications and Theoretical Aspects, 10th Industrial Conference, ICDM 2010, Berlin, Germany, July 12-14, 2010. Proceedings. Lecture Notes in Computer Science, vol. 6171, pp. 572–583. Springer (2010). https://doi.org/10.1007/978-3-642-14400-4_44
6. Cortez, P., Embrechts, M.J.: Using sensitivity analysis and visualization techniques to open black box data mining models. Inf. Sci. **225**, 1–17 (2013). https://doi.org/10.1016/j.ins.2012.10.039
7. Dalzochio, J., Kunst, R., Pignaton, E., Binotto, A., Sanyal, S., Favilla, J., Barbosa, J.: Machine learning and reasoning for predictive maintenance in industry 4.0: Current status and challenges. Computers in Industry **123**, 103298 (2020). https://doi.org/10.1016/j.compind.2020.103298
8. Fawcett, T.: An introduction to ROC analysis. Pattern Recognition Letters **27**, 861–874 (2006). https://doi.org/10.1016/j.patrec.2005.10.010
9. Fontes, G., Matos, L.M., Matta, A., Pilastri, A.L., Cortez, P.: An Empirical Study on Anomaly Detection Algorithms for Extremely Imbalanced Datasets. In: Artificial Intelligence Applications and Innovations - 18th IFIP WG 12.5 International Conference, AIAI 2022, Hersonissos, Crete, Greece, June 17-20, 2022, Proceedings, Part I. IFIP Advances in Information and Communication Technology, vol. 646, pp. 85–95. Springer (2022). https://doi.org/10.1007/978-3-031-08333-4_7
10. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press (2016), http://www.deeplearningbook.org
11. Liu, F.T., Ting, K.M., Zhou, Z.: Isolation Forest. In: Proc. of the 8th IEEE Int. Conf. on Data Mining (ICDM), Pisa, Italy. pp. 413–422. IEEE (2008). https://doi.org/10.1109/ICDM.2008.17
12. Matos, L.M., Azevedo, J., Matta, A., Pilastri, A., Cortez, P., Mendes, R.: Categorical Attribute traNsformation Environment (CANE): A python module for categorical to numeric data preprocessing. Software Impacts p. 100359 (2022). https://doi.org/10.1016/j.simpa.2022.100359
13. Matos, L.M., Cortez, P., Mendes, R., Moreau, A.: A Deep Learning-Based Decision Support System for Mobile Performance Marketing. International Journal of Information Technology & Decision Making (IJITDM) **22**(02), 679–703 (2023). https://doi.org/10.1142/S021962202250047X
14. Silva, A.J., Cortez, P., Pereira, C., Pilastri, A.L.: Business analytics in /Industry 4.0: A systematic review. Expert Systems **38**(7) (2021). https://doi.org/10.1111/exsy.12741
15. Singh, H.: Big data, industry 4.0 and cyber-physical systems integration: A smart industry context. Materials Today: Proceedings **46**, 157–162 (2021). https://doi.org/10.1016/j.matpr.2020.07.170
16. Özgün, K., Aklan, S., Tekin, A., Cebi, F.: Malfunction Detection on Production Line Using Machine Learning: Case Study in Wood Industry, pp. 1116–1124 (2021). https://doi.org/10.1007/978-3-030-51156-2_130