Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 00 (2019) 000–000

Procedia Computer Science

www.elsevier.com/locate/procedia

International Conference on Industry 4.0 and Smart Manufacturing

Predictive maintenance on sensorized stamping presses by time series segmentation, anomaly detection, and classification algorithms

Daniel Coelho[a], Diogo Costa[a,*], Eugénio M. Rocha[b,c], Duarte Almeida[d], José P. Santos[a]

[a]*Department of Mechanical Engineering, University of Aveiro, 3810-193, Portugal*
[b]*Department of Mathematics, University of Aveiro, 3810-193, Portugal*
[c]*Center for Research and Development in Mathematics and Applications (CIDMA), Aveiro, 3810-193, Portugal*
[d]*Bosch Termotecnologia S.A., 3810-193, Portugal*

Abstract

Sheet metal forming tools, like stamping presses, play an ubiquitous role in the manufacture of several products. With increasing requirements of quality and efficiency, ensuring maximum uptime of these tools is fundamental to marketplace competitiveness. Using anomaly detection and predictive maintenance techniques, it is possible to develop lower risk and more intelligent approaches to maintenance scheduling, however, industrial implementations of these methods remain scarce due to the difficulties of obtaining acceptable results in real-world scenarios, making applications of such techniques in stamping processes seldom found. In this work, we propose a combination of two distinct approaches: (a) time segmentation together with feature dimension reduction and anomaly detection; and (b) machine learning classification algorithms, for effective downtime prediction. The approach (a)+(b) allows for an improvement rate up to 22.971% of the macro F1-score, when compared to sole approach (b). A ROC AUC index of 96% is attained by using Randomized Decision Trees, being the best classifier of twelve tested. An use case with a decentralized predictive maintenance architecture for the downtime forecasting of a stamping press, which is a critical machine in the manufacturing facilities of Bosch ThermoTechnology, is discussed.

Keywords: Predictive Maintenance; Anomaly Detection; Machine Learning; Time Segmentation

1. Introduction

Heavy digitization of industrial practices is leading to an enormous collection of manufacturing data, gathered across all stages of the productive process through a wide range of sensing and Internet of Things (IoT) devices. Access to this information can be utilized towards improving business decisions and production planning, with particular

* Corresponding author. Tel.: +351-234-370-359 ; fax: +351-234-382-014.
E-mail address: d.costa@ua.pt

emphasis on enabling predictive maintenance (PdM) and early anomaly detection (AD) practices, due to the tremendous role maintenance strategies play at determining the success of manufacturing operations. In fact, maintenance is estimated to amount to 15-60% of all operational costs in manufacturing [1]. Conversely, incorrect or inadequate maintenance plans compromise a corporation's entire sustainability model [2], having negative effects such as the reduction by 5% to 20% of the overall machine productive capacity [3]. PdM aims to address this, as maintenance operations are planned well in advance based on predictions of future equipment health, leading to estimations regarding the system's remaining useful life (RUL), thus increasing equipment uptime and availability [3]. Due to the exorbitant human effort required to create knowledge-driven models for AD, data-driven techniques such as machine learning (ML) are seeing increasing use in PdM applications [4, 5].

Anomalies are patterns in data that do not conform to expected behavior, taking form of point, contextual, or collective anomalies [6], commonly being the sole manifestation of performance deficiencies perceivable in data [7]. When dealing with IoT data, additional challenges are imposed, due to the heterogeneous, large-scale data streams generated, commonly leading to highly correlated and high noise data. These attributes influence the proper choice of AD techniques [8]. Data-driven techniques, such as ML, are frequently used in PdM applications due to their capability of extracting relationships in data, otherwise concealed to users, even in high-dimensional and multivariate data common in industrial scenarios [9]. Yet, choosing a ML methodology that yields satisfactory results for a specific application is a daunting task, with performance dependent not only on choice of algorithms, but also on the initial understanding and analysis of the characteristics of datasets, amount of data available, and on the optimization of the initial stages of the ML pipeline [10, 11]. Stages of data driven PdM are analogous to typical stages of a ML pipeline, mainly consisting of two phases [12]: the learning process, where the model is trained using historical data; and inference, where trained models will predict target values based on new data. For each phase, data must be collected and pre-processed. Moreover, feature engineering techniques are required, extracting new features that will then be selected based on relevancy. After completing the training process, the goal is to generate a model with the optimal parameters. ML algorithms are usually classified into three categories [10, 13]: supervised learning, unsupervised learning, and reinforcement learning. In supervised learning, labeled input datasets exist to train models, consisting of examples with defined target outputs (labels), that can be taken as an absolute truth. During the training process, algorithms make predictions on the input data based on known labels, until an acceptable level of accuracy is reached. Supervised models should be preferred when desired outcomes of the model are known. Typically, they are used for *Classification* or *Regression* problems [10, 14], with both approaches being of value for PdM applications. For instance, classification problems can determine the probability of a failure within n cycles, while regression problems predict RUL of components or equipment, until next failure [2, 15].

Due to computational and time constraints, the training process is typically delegated to powerful compute environments (cloud-based servers). The inference phase, much lighter on resources, can be deployed within constrained devices in the edge layer [12, 13]. Large computational capabilities available in cloud computing make it suited for analysis of collected data through ML and deep learning (DL) algorithms, becoming the current standard for PdM and AD. However, we face a shift towards more decentralized computing platforms, into distributed systems where part of the data processing is also transferred to edge devices [13], which is of particular importance in time critical applications (e.g., PdM), where processing raw data into actionable information should happen in real-time. Cloud computing paradigms have shown limitations due to the inefficient utilization of network resources, with high bandwidth requirements and often leading to redundant data transmission between devices, whilst relying on a single point of failure [16]. In edge computing (EC), data is directly processed on-board the gathering sensors or on a gateway physically close to these devices [17]. As processing is done locally, it is more suitable for real-time applications or when fast decision is needed. The shift of computation towards the network's edge using typical EC hardware such as IoT has been discussed, with some advantages including: an increased energy efficiency; lower implementation cost; higher system reliability; and a decrease in overall system latency [18]. Furthermore, it addresses security and privacy concerns, usually associated with Cloud platforms, as data does not need to travel to centralized locations [19].

Leveraging IoT technologies, even legacy devices can be integrated into PdM frameworks. This is beneficial to equipment that plays a pivotal role in manufacturing industries, but is expected to operate for large periods of time, therefore lacking in modern sensitization technologies. One such example are sheet metal forming (SMF) tools and presses, that play ubiquitous roles in the manufacturing of several products [20]. Processes like progressive stamping are widely used in forming industries as they achieve high yield rates with high precision [21]. Keeping this in mind, the sensorized stamping press use-case is presented, where viability of a PdM system is asserted for use in a progressive die press at a real manufacturing facility. This mechanical press suffers from high number of breakdowns leading to a common state of unreliable or unavailable machine, creating significant financial impact, as it hampers the overall efficiency of several of the manufacturer's assembly lines. Difficulties are further exacerbated by the age of the equipment, showing low sensitization and restricted networking capabilities. To the best of author's knowledge, predictive maintenance applications for industrial metal stamping remain scarce. However, Zehetner et al. [22] discuss the creation of a digital twin for high-quality sheet metal production, yet, model-driven methods require high-effort and advanced procedural knowledge. Alternatively, Zhou et al. [21] proposed a method to automatically monitor missing part faults in progressive stamping processes, using Support Vector Machines (SVM) to identify critical faulty conditions. Likewise, in [23] Long Short-Term Memory (LSTM) networks were employed for the early detection of machine failures in a real-world metal stamping equipment use-case, achieving a 99% accuracy in failure classification, within a 5-minute time-window. Nevertheless, precision alone is an insufficient metric to evaluate a predictive maintenance platform. For instance, in critical applications, the number of false positives should be lower than 70% [5], while the number of false negatives is usually much more penalized, and thus, to fully validate implemented solutions, measures such as ROC AUC or F1-score should be calculated. In other industrial applications, we find a larger number of examples of successful PdM implementations. For instance, Ahmad et al. [24] measured rotational speed and vibration in rotating machines to classify faulty bearings using SVM, Naïve Bays, and Random Forests (RF), achieving accuracy values of 78% with SVM. Related work was done by Cakir et al. [25], although additional features of sound levels and temperature were added, concluding that Decision Trees (DT) frequently represent the best compromise between inference speed and classification scores. Tree-based algorithms are typically the best compromise between efficiency and predictive performance. For instance, in [26] RUL was estimated for real-world use cases, with bagging and boosting algorithms out-performing individual algorithms. Calabrese et al. [4] predicted failures in woodworking machines using tree-based classification methods, achieving 99.6% recall using Gradient Boosted Trees (GB). Faults can also be identified using time series forecasting models such as Adaptive Auto Regressive Integrated Moving Average (ARIMA). Adaptive ARIMA was proposed in [27], a varying window technique, used to predict the oil contamination value in hydraulic sand molding machinery. Glock [28] also discussed the value of combining ARIMA with RF techniques in order to increase explainability in fault prediction models.

Novelty of this paper is two-fold: we first propose a decentralized predictive maintenance architecture, compatible with the use-case manufacturing facility, contemplating all stages of data manipulation, from collection to prediction; secondly, we introduce a novel approach for ML classification based on feature extraction from preliminary AD techniques using segmented timeseries. We further demonstrate that using this approach we can greatly increase the predictive performance of the ML architecture, creating more reliable downtime forecasts for stamping presses, even when using real-world datasets.

2. Industrial Data Pipeline

Operational equipment data from the press is sent to the *Nexeed* Manufacturing Execution System (MES), originating from one of two sources: either by directly monitoring variables within the equipment's Programmable Logic Controller (PLC); or through external sensors retrofitted to the press. As such, it is possible for only a small group of fields to contain missing data in case of miscommunication. Measurements are done immediately after a part is produced, and over 25 variables are commonly monitored by technical experts. However, from this set, only the most relevant for determination of mechanical faults are selected. As typical techniques for monitoring mechanical systems include vibration analysis, oil analysis, and, for rotational components, rotational speed [24, 25, 27], metrics of oil temperature, oil level, press shaft vibration, and the engine's actual speed are retrieved and used for examination. Preliminary features also include the pressure on the left and right side connecting rods. As several distinct tools are used during an ordinary production day, with each one influencing the selected features differently, tool height is tracked as this is the differential characteristic between tools. A short description of these variables is presented in Table 1.

Further enriching gathered datasets, additional metadata is used, including comprehensive production logs which detail each shift's productivity, and, more importantly, manage machine downtimes and planned stops, with unplanned stops being classified and categorized according to probable cause. For the purposes of predictive maintenance, only downtimes derived from technical issues are of interest. Technical downtimes are recorded whenever machine or

Table 1. Description of relevant variables measured during sensorized stamping press operation.

Sensor Name	Description	Unit
OilTemperature	Oil temperature in the hydraulic tank	ºC
OilLevel	Oil level in the hydraulic tank	%
EngineActualSpeed	Electric motor speed	rpm
ShaftVibration	Average vibration of crankshaft bearings and connecting rod bearings	mg
PressureLeft	Hydraulic pressure on the left connecting rod supercharge relief system	bar
PressureRight	Hydraulic pressure on the right connecting rod supercharge relief system	bar
ToolHeight	Height of current tool in press	mm

tool issues arise that require either intervention from the maintenance team or from the operator. Seeing that this information is already registered by the technical staff and is properly structured, this can be leveraged by the ML pipeline creating a dataset labelling channel for use with supervised learning algorithms.

2.1. *Implemented Architecture*

The data pipeline for the predictive maintenance system is shown in Figure 1. This pipeline receives data in real-time from the edge device installed on the press (PLC), and, through several data transformations, produces forecasts identifying whether or not there will be a failure. The differentiating element of the proposed approach to more conventional ones is the segmentation of time series into two segments, *Working* and *Stopped*. Through an analysis of the press data, it is possible to identify three different operating patterns: there are periods where the press is working; there are periods where the press is stopped; and there are periods where, although the press is working, it is in a test cycle. The frequency with which data is sent by the edge device allows to clearly identify these three periods. Whenever the frequency is close to 1 Hz, the press is working; whenever the frequency is 0 Hz, the press is stopped; and whenever the frequency is between 0 Hz and 0.5 Hz, the press is in testing periods. The segmentation groups the testing and downtime periods into the *Stopped* segment, leaving only data related to the working period in the *Working* segment. This segmentation, therefore, can be seen as a filter, once it separates the useful data (*Working* segment) from the noisy data (*Stopped* segment), and in the future, when detecting anomalies and extracting data patterns, only data belonging to the *Working* segment will be used, reducing complexity of employed algorithms. Although in this work, implemented segmentation has focused on the division of the time series into *Working* and *Stopped* states, the concept of segmentation can be further explored. Based on the OilLevel time series, Fig. 3, it is possible to identify two new states inside the *Working* state (green background): *Warming* and *Steady*. The first state corresponds to the period when oil starts rising, with the second corresponding to the period when the oil level finally stabilizes. This second segmentation would distribute the data into more similar segments, further aiding AD. Moreover, the present work also proposes a synergy between an unsupervised AD algorithm and a supervised ML algorithm, in which, through

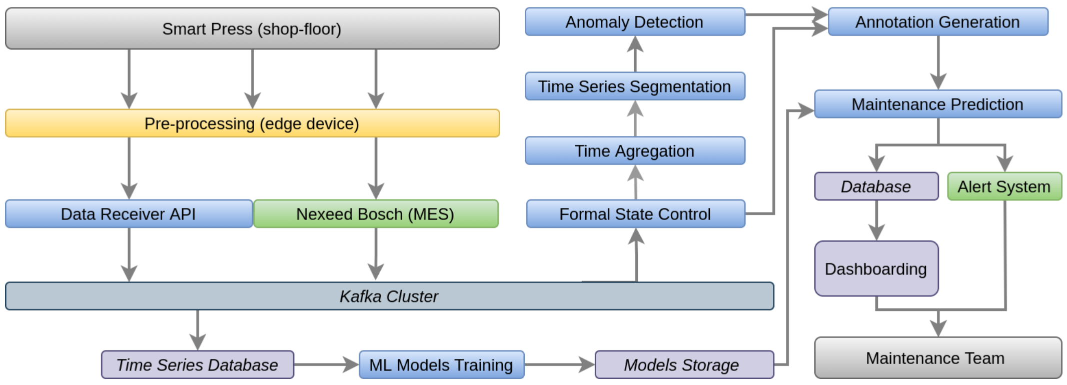

Fig. 1. Data pipeline for the predictive maintenance, where blue boxes are micro-services developed for this project.

the analysis of the data belonging to the *Working* segments, the observations that least resemble the neighborhood are identified using an unsupervised AD algorithm, and later, this information is used as an additional feature in the developed pipeline of the supervised ML algorithm.

The remainder of this section explains each component of the presented data pipeline. Near the stamping press exists a PLC which acquires measurements from sensors installed on the press, and sends data to a REST API. After acquiring each measurement, the edge device creates a JSON file retaining all information related to the measurement and sends this file to the REST API, through a POST request. The REST API, when receiving the request, generates a Kafka message that is forwarded to the Kafka Cluster, where processing will take place. Besides the REST API, there is an alternative way for ingesting press data into the Kafka Cluster: the edge device sends the press data in a XML telegram to the MES, and then, a micro-service is responsible for generating a Kafka message for each measurement data, simulating real-time data transmission. The real processing starts when the Kafka messages reach the Kafka Cluster. At first, the messages are grouped according to the sensor in which they belong. Then, all groups of messages are aggregated into 10-second time-windows, and for each aggregation the number of messages is calculated. If all aggregations have more than 5 messages the *Working* state is activated, otherwise, the *Stopped* state is activated. Through this comparison the frequency explained above is analyzed, and it is possible to activate the *Working* state only when the stamping press is truly working. The micro-service responsible for activating each state is called "Formal State Control", and besides managing the active state, it also associates each Kafka message that arrives with the active state. Therefore, as each incoming message is associated with the active state, segmentation is performed in real-time. Although time series segmentation is running in real-time, the micro-service "Anomaly Detection" only runs every 10 minutes, and before anomalies are identified it is necessary to perform a segment pre-processing. Initially, only data belonging to the *Working* segment of all time series are acquired. Then, this segments must be merged in order to eliminate empty spaces left by data belonging to the *Stopped* segments. This merging originates a new base time, *Virtual Base Time*, which is common to all merged time series. Subsequently, the method scaling to minimum and maximum was applied in order to transform all time series values between 0 and 1. After that, we applied Principal Component Analysis (PCA) as a dimensionality reduction algorithm, reducing the number of features (time series) to be analyzed. Finally, we applied an unsupervised AD algorithm on the principal components determined in the previous step. After the AD algorithm detects anomalies in the *Virtual Time Base*, these are converted to the *Real Time Base* and are associated with the observations belonging to the same temporal instant.

The ”ML Models Training” component has access to the ”Database” module, which can be considered as the repository of the entirety of data flowing through the pipeline. This means that during training of the ML models, historical data can already be complemented with additional metadata elements such as maintenance logs for the creation and labelling of features. The first stage of the ML training procedure consists on the creation of an additional feature, derived from the results of the preliminary AD algorithm. Adding to the initial selection of features, this new feature will be a binary indicator of whether the segmented time series analysis identified any anomalous data segment. Data will then be aggregated into 10-minute time-windows, and for each one of these time-windows aggregation primitives are calculated for every feature apart from tool height, namely: total count; mean; maximum; minimum; sum; number of null values; standard deviation; and the number of consecutive zeros (consecutive null count values). After the feature extraction phase, the total number of existing features will increase 8-fold. As most of these features are redundant or of no added value to the ML model, feature selection methods are required. The very choice of feature selection methods can be done through an iterative process as tuning any other hyperparameter. Yet, for simplicity, initially the Pearson Correlation Coefficient is calculated between features and those with a correlation threshold superior to 90% are removed. During the last pre-processing stage, time series data is converted into a ML classification problem format, however, to maintain the historical relationship between each data entry, shifted features are created containing the feature value for the two previous time steps. Furthermore, the label of each entry is also shifted back a time step, so we can classify a fault-event within a 10-minute interval.

The following ML stages are implemented based on software solutions provided by Python's Scikit-Learn package module [29]. Training, testing, and validation dataset split proportions can be altered at will. For simulation purposes the default value is the traditional proportion of a 70%-30% train-test split. Hyperparameters are selected after a Randomized Search through extensive search spaces for each of the ML algorithms wished to test. At each stage of the random search, 5-fold cross-validation takes place and the results with the higher macro average F1-score are selected. This process is repeated for 150 iterations, to ensure more robust results [11]. Any number of metrics can be used to evaluate model's performance, however macro F1-score, accuracy, and ROC AUC are particularly pertinent. This model will be saved and used for inference in "Maintenance Prediction". Periodically, models will be retrained using fresh data and metadata, assuring that models are adjusted to changing environmental and equipment conditions. In case critical faults are predicted to happen in short order, the "Alert System" will be activated, and the maintenance team warned.

3. Main Results

The initial dataset used to simulate the proposed architecture's operation is comprised of real data and is representative of a full manufacturing day. It consists of measurements of the 7 initial features (OilTemperature, OilLevel, EngineActualSpeed, ShaftVibration, PressureLeft, PressureRight, and ToolHeight) across a 17-hour period, totaling in 13.568 instances. The dataset was binarized into two classes: working state (0), or failure state (1). The former is represented by 13.488 data-points, whilst the latter by 80 data-points that describe two occurrences of mechanical failures. With only 0.59% of instances corresponding to anomalous behavior, this leads to highly unbalanced classes. However, to maintain solution generalization, and seeing that failures do not follow a steady occurrence pattern (e.g., they can either occur near each other, or not at all during prolonged periods of time), no over or under-sampling techniques were used [11]. Anomalous points are automatically labelled through parsing tools developed for the platform. Maintenance logs are parsed and only machine downtimes corresponding to technical faults are annotated in the ML dataset. Additionally, as each probable cause of failure is also identified, only mechanical breakdowns that can be predicted using selected features are labelled as failures. This avoids tainting point labels with faults external to the equipment, but that caused a production line downtime (e.g., due to power shortages). As such, three types of technical failures are maintained: mechanical machine failures, tool failures, and obstructions to the press punch.

Fig. 2 shows the Pearson Correlation Coefficient between the 7 initially chosen features. Within this set, it is clear that only the relationship between OilTemperature and PressureLeft reveals a high correlation value. In fact, OilTemperature is the feature with the highest correlation with most features, on average; showing also a strong connection with OilLevel, and to a lesser extent, PressureRight. The remaining features remain with relatively low correlations, and thus it is expected that each one will provide additional information to the ML model. In Fig. 3 the OilLevel time series is shown, properly segmented in the considered states. Different segments are identified with a different background color: *Working* segments are represented with the green color; and *Stopped* segments are represented with the red color. Around the 19:50h time-mark, there is a period when the temporal space between successive observations increases, symbolizing a period where some test is being performed, and, as we can see, the segmentation acknowledged that period and activated the *Stopped* state, instead of the *Working* state. Therefore, when performing AD, test data will not be used, otherwise, it would be expected the detection of various anomalies in that period, due to high disparity between the test period and the working period.

When there are few features, in this case, time series, there may be poor results due to the lack of variety in data. However, when there are many features, there is the possibility that any existing observation would appear different from the rest due to the high number of comparison features. PCA, as explained before, was used for dimensionality reduction, after applying a min-max normalization data scaler. Data was reduced to 2 principal components, obtaining 92.60% of the variability contained in the original features. Thus, it is possible to reduce the complexity of data by losing only 7.4% of information. It is important to note that these values were obtained by simulating the data analysis at a given moment. Nonetheless, as the analysis of the segments occurs every 10 minutes, each time the analysis is carried out these percentages are different. Fig. 4 depicts the two principal components used. The first principal component, PC1, explains 77.89% of the variance of the 7 initial features, whilst the second principal component, PC2, explains 14.71%. By using these 2 principal components, instead of the full 7 initial features, we increase the processing speed of AD without losing any significant information.

For the unsupervised AD algorithm, several algorithms were tested, such as Angle-Based Outlier Detection (ABOD) [30], Stochastic Outlier Selection (SOS) [31] and Copula-Based Outlier Detection (COPOD) [32], however, the one that yields the best results was COPOD. Fig. 4 also contains the anomalies detected by COPOD, superimposed on the two principal components. The performance of the preliminary results obtained using only AD techniques on the segmented time series to predict unplanned stops based on anomalous segments is depicted in Table 2. Despite initial positive results, this method only points to the possible existence of a failure within the useful forecast period, without any precise information regarding when machine failure occurs, limiting estimations of component's RUL. Furthermore, this does not consider any historical data patterns that point to the state of the machine degradation leading up to a failure over time. By combining these results and extracting a new feature to be used together with a strong ML model, we can more accurately identify a failure within a 10-minute time window.

Performance of the strong ML models was evaluated for the testing set that was set aside from the training procedure and thus contained data previously unseen by the models. Several techniques were used to test the influence of the added segmented time series feature, including: DT, GB, XGBoost, RF, SVM, Gaussian Naive Bayes, K-Neighbors, Logistic Regression, Multi-layer Perceptrons, Naive Bayes, Randomized DT, and Regularized Linear models. All algorithms were implmented using Scikit-Learn [29] version 0.24.2 and PyTorch [33] version 1.9.0. They were tested for both scenarios, with and without the segmented time series feature. Accuracy, ROC AUC, and macro F1-score were calculated for each iteration of the randomized search and results are shown in Table 3. The results clearly point to an improvement in performance after adding the segmented time feature, for practically all algorithms and metrics. Randomized DT in particular were the best performers with segmentation, achieving a macro F1-score of 91.66%, an extremely satisfactory value, taking into consideration the deep unbalance present in the fairly small dataset. Reg-

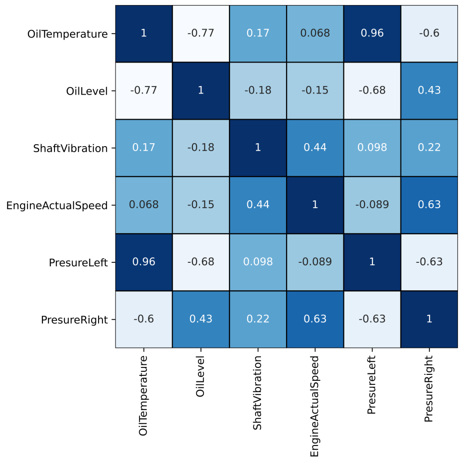

Fig. 2. Correlation matrix of the most relevant features.

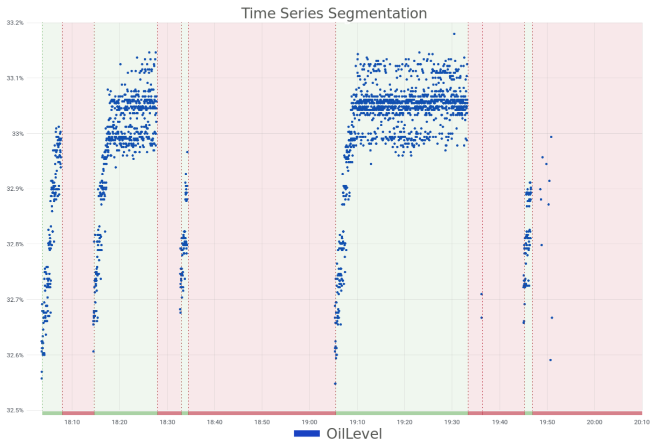

Fig. 3. Time series OilLevel segmented

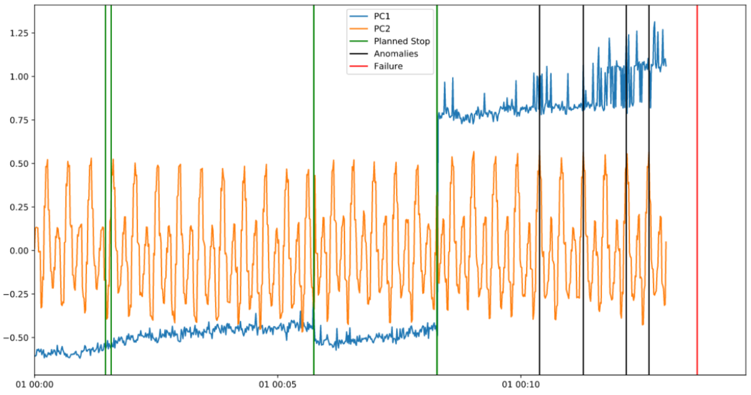

Fig. 4. Anomalies detected using two principal components

Table 2. Calculated performance metrics for the segmented time series anomaly detection algorithm.

Accuracy	Precision	Recall	F1-score
0.939	0.500	1.000	0.667

Table 3. Calculated performance metrics for the testing set, where the values are the average of 5 runs.

	Accuracy		ROC AUC		Macro F1	
Method	without seg.	with seg.	without seg.	with seg.	without seg.	with seg.
Decision Trees	77.42%	90.63%	77.35%	83.72%	77.04%	85.52%
Gaussian Naive Bayes	80.65%	81.50%	80.13%	92.00%	80.13%	84.54%
Gradient Boosting	77.42%	93.75%	74.15%	90.86%	74.80%	90.86%
K-Neighbors	74.19%	90.63%	71.37%	88.86%	71.82%	86.94%
Logistic Regression	74.20%	71.88%	71.37%	66.67%	71.81%	63.95%
Multi-layer Perceptron / LBFGS	77.42%	84.34%	73.08%	74.57%	73.44%	75.87%
Naive Bayes / Multivariate Bernoulli	77.42%	78.12%	76.28%	80.86%	75.54%	73.80%
Random Forest	70.97%	87.15%	68.59%	71.43%	68.90%	76.29%
Randomized Decision Trees[1]	80.65%	**93.75**%	77.99%	**96.00**%	78.86%	**91.66**%
Regularized Linear models / SGD[2]	**83.87**%	81.25%	**82.91**%	88.00%	**83.24**%	78.18%
Support Vector Machine	80.65%	87.50%	76.92%	76.75%	77.86%	79.48%
XGBoost	74.19%	87.50%	69.77%	85.71%	70.48%	86.67%

[1] Bootstrap=True, criterion=entropy, max_features=0.4, min_samples_leaf=5, min_samples_split=10, n_estimators=100.

[2] Stacking of 2 models with alpha=0.001, eta_0=[1.0,0.01], fit_intercept=[True,False], l1_ratio=[0.0,0.25], penalty=elasticnet, learning_rate=invscaling, loss=squared_hinge, power_t=[10.0,0.5].

ularized Linear models showed the best performance for the dataset without segmentation, falling short of several tree-based algorithms using segmentation. In fact, tree-based are generally the ones that most benefit from the additional feature, with DT, GB, RF, Randomized DT, and XGBoost, seeing macro F1-score percentage increases of 11.00%, 21.47%, 10.73%, 16.23%, and 22.97%, respectively. XGBoost, although not the best performer overall, saw the highest F1-score improvement, and in Fig. 5 we see the importance for the top-10 features acting on the model, where the sum of anomalies contained in the 10-minute aggregation window exerts a much higher influence over the model comparatively to other features.

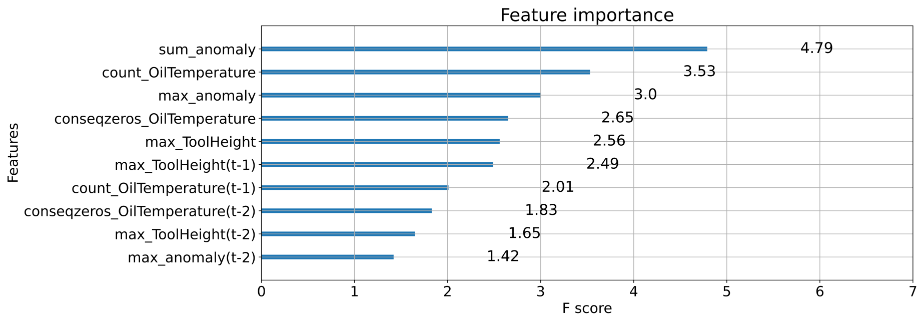

Fig. 5. Feature importance (gain) of top-10 features acting in XGBoost model

4. Conclusions

In this paper, we proposed the combination of time segmentation with feature reduction and AD, together with strong ML classification algorithms, to be used for downtime prediction in sheet metal forming tools (sensorized stamping presses). Furthermore, we proposed a comprehensive architecture for implementing predictive maintenance aimed at solving the real-world use-case at Bosch ThermoTechnology. First, the relevant state-of-art was discussed and advantages of PdM applications shown. Moreover, preliminary analysis of data gathered for the use-case revealed a deeply unbalanced dataset, that proves to be challenging to more conventional ML techniques. During result discussion, we showed the successful integration of time series segmentation into strong ML models, with a percentage increase of 16.23% in macro F1-score for the proposed approach, compared to a common ML application.

However, a few limitations of the platform are clear, and highlighted as future work and research directions. Namely, the used dataset is fairly small and proper validation of the proposed solution still requires more data. Additionally, the implemented segmentation has several hyperparameters, such as, aggregation time, minimum number of messages in the aggregation to activate the *Working* status, time interval between each anomaly detection, maximum and minimum virtual time admissible in the anomaly detection, and proportion of anomalies in the dataset. The exorbitant number of possible combinations demands the development of an optimization algorithm, that would select the conjunction of hyperparameters that maximizes efficiency of a specific metric, namely F1-score. Moreover, it would be of use a platform capable of performing the segmentation with a time reduction factor, enabling data segmentation representative of several days in just a few hours. This is also fundamental for the optimization algorithm, otherwise, time taken in hyperparameter tuning would be unmanageable. It would also be of interest to evolve the segmentation performed into two segmentation phases, the first phase segmenting the time series into *Working* and *Stopped*, and the second phase, segmenting the *Working* segments into *Warming* and *Steady* segments. This way, data would be grouped into more similar segments, expecting a sensitivity increase of the AD algorithm. In case different states must be analyzed, instead of using a single ML model we could use one ML model for each state. Thus, depending on the active state, the forecast is issued by the ML model associated with the active state.

Although several improvements are to be made, if the developed platform had been in production on the studied day, at its current form, the press would have worked an additional 4 hours, 55 minutes, and 6 seconds (28.9% of the full manufacturing day), vastly increasing productivity. Furthermore, results obtained by the predictive platform can be used to conduct real-time cross-correlation between machine failures and nefarious operational practices that affect equipment health, that would stand otherwise hidden or hard to detect.

Acknowledgements. The present study was developed in the scope of the Project Augmented Humanity (PAH) [POCI-01-0247-FEDER-046103], financed by Portugal 2020, under the Competitiveness and Internationalization Operational Program, the Lisbon Regional Operational Program, and by the European Regional Development Fund. The second author has a PhD grant supported by PAH. The third author was partially supported by the Center for Research and Development in Mathematics and Applications (CIDMA), through the Portuguese Foundation for Science and Technology, reference UIDB/04106/2020. The fourth author was supported by Bosch Termotecnologia.

Data availability statement. The data sets used to obtain the results are confidential information of Bosch company manufacturing system, so they are not publicly available.

References

[1] T. Zonta, C. A. da Costa, R. da Rosa Righi, M. J. de Lima, E. S. da Trindade, G. P. Li, Predictive maintenance in the Industry 4.0: A systematic literature review, Computers and Industrial Engineering 150 (April 2019) (2020) 106889. `doi:10.1016/j.cie.2020.106889`. URL `https://doi.org/10.1016/j.cie.2020.106889`

[2] M. Nacchia, F. Fruggiero, A. Lambiase, K. Bruton, A systematic mapping of the advancing use of machine learning techniques for predictive maintenance in the manufacturing sector, Applied Sciences (Switzerland) 11 (6) (2021) 1–34. `doi:10.3390/app11062546`.

[3] C. Colemen, S. Damodaran, M. Chandramoulin, E. Deuel, Making maintenance smarter, Deloitte University Press (2017) 1–21.

[4] M. Calabrese, M. Cimmino, F. Fiume, M. Manfrin, L. Romeo, S. Ceccacci, M. Paolanti, G. Toscano, G. Ciandrini, A. Carrotta, M. Mengoni, E. Frontoni, D. Kapetis, SOPHIA: An event-based IoT and machine learning architecture for predictive maintenance in industry 4.0, Information (Switzerland) 11 (4) (2020) 1–17. `doi:10.3390/INFO11040202`.

[5] B. Steenwinckel, D. De Paepe, S. Vanden Hautte, P. Heyvaert, M. Bentefrit, P. Moens, A. Dimou, B. Van Den Bossche, F. De Turck, S. Van Hoecke, F. Ongenae, FLAGS: A methodology for adaptive anomaly detection and root cause analysis on sensor data streams by fusing expert knowledge with machine learning, Future Generation Computer Systems 116 (2021) 30–48. `doi:10.1016/j.future.2020.10.015`. URL `https://doi.org/10.1016/j.future.2020.10.015`

[6] V. Chandola, A. Banerjee, V. Kumar, Anomaly Detection: A Survey, ACM Comput. Surv. 41 (3) (2009). `doi:10.1145/1541880.1541882`. URL `https://doi.org/10.1145/1541880.1541882`

[7] H. H. W. J. Bosman, Anomaly detection in (networked) embedded systems, Technische Universiteit Eindhoven, 2016.

[8] M. Mohammadi, A. Al-Fuqaha, S. Sorour, M. Guizani, Deep learning for IoT big data and streaming analytics: A survey, arXiv 20 (4) (2017) 2923–2960.

[9] T. P. Carvalho, F. A. Soares, R. Vita, R. d. P. Francisco, J. P. Basto, S. G. Alcalá, A systematic literature review of machine learning methods applied to predictive maintenance, Computers and Industrial Engineering 137 (April) (2019) 106024. `doi:10.1016/j.cie.2019.106024`. URL `https://doi.org/10.1016/j.cie.2019.106024`

[10] Z. M. Çinar, A. A. Nuhu, Q. Zeeshan, O. Korhan, M. Asmael, B. Safaei, Machine learning in predictive maintenance towards sustainable smart manufacturing in industry 4.0, Sustainability (Switzerland) 12 (19) (2020). `doi:10.3390/su12198211`.

[11] O. Gómez-Carmona, D. Casado-Mansilla, F. A. Kraemer, D. López-de Ipiña, J. García-Zubia, Exploring the computational cost of machine learning at the edge for human-centric Internet of Things, Future Generation Computer Systems 112 (2020) 670–683. `doi:10.1016/j.`

future.2020.06.013.
URL https://doi.org/10.1016/j.future.2020.06.013

[12] W. Zhang, D. Yang, H. Wang, Data-Driven Methods for Predictive Maintenance of Industrial Equipment: A Survey, IEEE Systems Journal 13 (3) (2019) 2213–2227. doi:10.1109/JSYST.2019.2905565.

[13] L. Erhan, M. Ndubuaku, M. Di Mauro, W. Song, M. Chen, G. Fortino, O. Bagdasar, A. Liotta, Smart anomaly detection in sensor systems: A multi-perspective review, Information Fusion 67 (June 2020) (2021) 64–79. arXiv:2010.14946, doi:10.1016/j.inffus.2020.10.001.
URL https://doi.org/10.1016/j.inffus.2020.10.001

[14] A. Dogan, D. Birant, Machine learning and data mining in manufacturing, Expert Systems with Applications 166 (February 2019) (2021) 114060. doi:10.1016/j.eswa.2020.114060.
URL https://doi.org/10.1016/j.eswa.2020.114060

[15] J. Dalzochio, R. Kunst, E. Pignaton, A. Binotto, S. Sanyal, J. Favilla, J. Barbosa, Machine learning and reasoning for predictive maintenance in Industry 4.0: Current status and challenges, Computers in Industry 123 (2020) 103298. doi:10.1016/j.compind.2020.103298.
URL https://doi.org/10.1016/j.compind.2020.103298

[16] M. Ahmed, R. Mumtaz, S. M. H. Zaidi, M. Hafeez, S. A. R. Zaidi, M. Ahmad, Distributed fog computing for internet of things (Iot) based ambient data processing and analysis, Electronics (Switzerland) 9 (11) (2020) 1–20. doi:10.3390/electronics9111756.

[17] L. Greco, G. Percannella, P. Ritrovato, F. Tortorella, M. Vento, Trends in IoT based solutions for health care: Moving AI to the edge, Pattern Recognition Letters 135 (2020) 346–353. doi:10.1016/j.patrec.2020.05.016.

[18] R. Sanchez-Iborra, A. F. Skarmeta, TinyML-Enabled Frugal Smart Objects: Challenges and Opportunities, IEEE Circuits and Systems Magazine 20 (3) (2020) 4–18. doi:10.1109/MCAS.2020.3005467.

[19] G. Mohindru, K. Mondal, H. Banka, Internet of Things and data analytics: A current review, Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 10 (3) (2020) 1–27. doi:10.1002/widm.1341.

[20] C. J. Jonsson, R. Stolt, F. Elgh, Stamping tools for sheet metal forming: Current state and future research directions, Advances in Transdisciplinary Engineering 12 (2020) 281–290. doi:10.3233/ATDE200087.

[21] C. Zhou, K. Liu, X. Zhang, W. Zhang, J. Shi, An Automatic Process Monitoring Method Using Recurrence Plot in Progressive Stamping Processes, IEEE Transactions on Automation Science and Engineering 13 (2) (2016) 1102–1111. doi:10.1109/TASE.2015.2468058.

[22] C. Zehetner, C. Reisinger, W. Kunze, F. Hammelmüller, R. Eder, H. Holl, H. Irschik, High-quality sheet metal production using a model-based adaptive approach, Procedia Computer Science 180 (2019) (2021) 249–258. doi:10.1016/j.procs.2021.01.162.
URL https://doi.org/10.1016/j.procs.2021.01.162

[23] F. Alves, H. Badikyan, H. J. Antonio Moreira, J. Azevedo, P. M. Moreira, L. Romero, P. Leitao, Deployment of a Smart and Predictive Maintenance System in an Industrial Case Study, IEEE International Symposium on Industrial Electronics 2020-June (2020) 493–498. doi:10.1109/ISIE45063.2020.9152441.

[24] B. Ahmad, R. Forest, Intelligent Predictive Maintenance Model for Rolling Components of a Machine based on Speed and Vibration (2021) 459–464.

[25] M. Cakir, M. A. Guvenc, S. Mistikoglu, The experimental application of popular machine learning algorithms on predictive maintenance and the design of IIoT based condition monitoring system, Computers and Industrial Engineering 151 (October 2020) (2020) 106948. doi:10.1016/j.cie.2020.106948.
URL https://doi.org/10.1016/j.cie.2020.106948

[26] S. Ayvaz, K. Alpay, Predictive maintenance system for production lines in manufacturing: A machine learning approach using IoT data in real-time, Expert Systems with Applications 173 (January) (2021) 114598. doi:10.1016/j.eswa.2021.114598.
URL https://doi.org/10.1016/j.eswa.2021.114598

[27] T. Roosefert Mohan, J. Preetha Roselyn, R. Annie Uthra, D. Devaraj, K. Umachandran, Intelligent machine learning based total productive maintenance approach for achieving zero downtime in industrial machinery, Computers Industrial Engineering 157 (June 2020) (2021) 107267. doi:10.1016/j.cie.2021.107267.
URL https://doi.org/10.1016/j.cie.2021.107267

[28] A. C. Glock, Explaining a Random Forest with the Difference of Two ARIMA Models in an Industrial Fault Detection Scenario, Procedia Computer Science 180 (2019) (2021) 476–481. doi:10.1016/j.procs.2021.01.360.
URL https://doi.org/10.1016/j.procs.2021.01.360

[29] 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, Journal of Machine Learning Research 12 (2011) 2825–2830.

[30] H. P. Kriegel, M. Schubert, A. Zimek, Angle-based outlier detection in high-dimensional data, Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (2008) 444–452doi:10.1145/1401890.1401946.

[31] J. Janssens, F. Huszar, E. Postma, H. van den Herik, Stochastic Outlier Selection (2012).

[32] Z. Li, Y. Zhao, N. Botta, C. Ionescu, X. Hu, COPOD: Copula-based outlier detection, Proceedings - IEEE International Conference on Data Mining, ICDM 2020-Novem (September) (2020) 1118–1123. arXiv:2009.09463, doi:10.1109/ICDM50108.2020.00135.

[33] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, S. Chintala, Pytorch: An imperative style, high-performance deep learning library, in: H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, R. Garnett (Eds.), Advances in Neural Information Processing Systems 32, Curran Associates, Inc., 2019, pp. 8024–8035.
URL http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf