Computers & Industrial Engineering 184 (2023) 109566

Contents lists available at ScienceDirect

Computers & Industrial Engineering

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

Remaining useful lifetime prediction for predictive maintenance in manufacturing

Bernar Taşcı [a], Ammar Omar [b], Serkan Ayvaz [c,d,*]

[a] Department of Computer Engineering, Bahcesehir University, Istanbul, Turkey
[b] Department of Gebze Plant Digital Team, Gebze Development Center, Procter & Gamble, Kocaeli, Turkey
[c] Department of Computer Engineering, Yildiz Technical University, Istanbul, Turkey
[d] Centre for Industrial Software, University of Southern Denmark, Sonderborg, Denmark

ARTICLE INFO

Keywords:
Predictive maintenance
Machine learning
Remaining useful lifetime
Manufacturing

ABSTRACT

Traditional maintenance approaches often result in either premature replacement of machine parts or downtime in production lines due to malfunctions. Consequently, these lead to significant amount waste in material, time and, ultimately, money. In this study, a machine learning-based predictive maintenance approach is proposed to predict the Remaining Useful Life of production lines in manufacturing. Using data collected from integrated IoT sensors in a real-world factory, we attempted to address the problem of predicting potential equipment failures on assembly-lines before they occur through machine learning models in real-time. To evaluate the effectiveness of the approach, we developed several predictive models using ML algorithms, including Random Forests (RF), XGBoost (XGB), Multilayer Perceptron (MLP) and Support Vector Regression (SVR) and compared the results for all possible variations. Furthermore, the impact of noise filtering, smoothing and clustering techniques on the performance of ML models were investigated. Among all the methods evaluated, RF, an ensemble bagging method, showed the best performance, followed by XGB and implemented in production systems. The implemented prediction model achieved successful results and was able to prevent about 42 percent of actual production line failures.

1. Introduction

While Industry 4.0, a new era in industrial production, is rapidly transforming the mechanisms by which companies produce products, artificial intelligence-driven manufacturing is expanding its place in our daily lives day by day (Lee, Davari, Singh, & Pandhare, 2018). The rapid increase in population created the need for complex production lines to meet demand.

With the technological and industrial developments in manufacturing, more intelligent and complex machines have taken their place in production lines, creating smarter ecosystems. However, the maintenance of such complex systems has become vital for reliability and sustainability.

Recent innovations in technology have begun to change the traditional behavior of human–machine interaction. Advances in artificial intelligence (AI) and big data along with ever-increasing processing power of computers have made this interaction bidirectional. This change has found its place not only in daily life, but also in industry and production environments in terms of automation. In order to stay competitive, companies began to depend more and more on automation to increase productivity (Wright & Schultz, 2018).

Thanks to the advances in device-to-device communication technologies, the concept of the Internet of Things (IoT) has emerged, which facilitates interconnectivity and enables active monitoring of systems in production lines. The constant communication of such devices creates what is called big data, which must be dealt with systematically in industrial facilities. Big data provides the capability for self-calibration and system performance improvements, enabling system coordination and feedback across devices (Wang, Wan, Zhang, Li, & Zhang, 2016).

While conventional maintenance approaches continued until recent years, the integration of AI and Industrial Internet of Things (IIoT) into the manufacturing formed a new phase where Predictive Maintenance (PdM) become the main objective. With the implementation of PdM, planned maintenance approaches led the way to efficient use of machinery up to their useful life. This has diminished the effects of unintentional and unnecessary stops for maintenance purposes, creating a cost-saving, high-efficiency, fully optimized manufacturing perspective.

Correctly predicted alerts can help prevent unnecessary waste from downtime until the time of a restart, as production lines cannot produce usable goods until they are fully functional again. The aim of this

* Correspondence to: Alsion 2, Building A 6400, Sonderborg, Denmark.
E-mail addresses: bernar.tasci@alumni.bau.edu.tr (B. Taşcı), omar.aa@pg.com (A. Omar), seay@mmmi.sdu.dk (S. Ayvaz).

https://doi.org/10.1016/j.cie.2023.109566
Received 27 February 2023; Received in revised form 15 July 2023; Accepted 22 August 2023
Available online 23 August 2023

study was to develop a predictive maintenance system to detect future failures and send alerts before they occur, using machine learning algorithms based on data collected through IoT sensor readings in a real-world manufacturing facility. In order to create such a system, different data-driven approaches and machine learning algorithms were applied on a real-world dataset in a comparative manner.

The main contributions of this paper can be listed as follows:

- The study proposes a hybrid machine learning-based predictive maintenance approach for manufacturing that combines filtering, feature engineering using Autoencoders, clustering, and prediction.
- Machine learning models are developed using real-world high-dimensional data from IoT sensors on consumer goods production assembly lines.
- The proposed system provides a validated, end-to-end solution for predicting the remaining useful lifetime before production lines stop, based on extensive evaluations.

The rest of the paper is organized as follows: In Section 2, the related work is reviewed. Section 3 introduces the dataset used for the study, describes the data preprocessing steps and explains the proposed predictive maintenance system and the algorithms in detail. In the subsequent section, the results of the evaluations assessing the effectiveness of the proposed methods are presented. Section 5 discusses the findings of this study. Finally, the conclusion and the plans for future work are given in Section 6.

2. Literature review

Predictive Maintenance (PdM) is an important issue in industry and manufacturing due to the costs associated with unplanned, failure-induced downtime in plants or operating machines. Various studies have investigated the predictive maintenance problem from different perspectives. Some studies offered statistical approaches (Francis & Mohan, 2019; Rivera-Gómez, Gharbi, Kenné, Ortiz-Zarco, & Corona-Armenta, 2021; Shimada & Sakajo, 2016), while others proposed data-driven solutions for predictive maintenance (Ayvaz & Alpay, 2021; Cakir, Guvenc, & Mistikoglu, 2021; Del Buono, Calabrese, Baraldi, Paganelli, & Guerra, 2022; Nasir & Sassani, 2021; Soltanali, Khojastehpour, Farinha, & Pais, 2021).

2.1. Statistical approaches to predictive maintenance

From a statistical perspective, Shimada & Sakajo proposed a method based on probability distributions for maintenance scheduling in their study (Shimada & Sakajo, 2016). The focus of their approach was mainly on the timing of maintenance after the warning given by the model. This was developed using a characteristic index obtained by the cumulative failure probability distribution. An approach that analyzes the alert state to failure can improve the performance of model predictions by estimating the time required for maintenance after the model alert occurs and reducing the burden on maintenance workers. Their method was able to reduce the failure rate by 12 percent.

In another study, Ho et al. utilized Autoregressive Integrated Moving Average (ARIMA) and Recurrent Neural Network (RNN) models to predict failures (Ho, Xie, & Goh, 2002). A combination of historical values and errors was used in ARIMA to model a non-seasonal time series. Their findings showed that both methods perform better at short-term dependencies. However, a comparison demonstrated that feedforward neural networks performed poorly compared to ARIMA and RNN.

Similarly, Kanawaday and Sane investigated the ARIMA forecasting to predict failures and quality defects in real-world data collected from Slitting Machines (Kanawaday & Sane, 2017). They proposed a two-stage approach: the first stage begins with data analysis, clustering and supervised learning methods to obtain information, and it is followed by the implementation of the predictive model using ARIMA. Their proposed system architecture consisted of two stack systems, one with ARIMA to predict parameter values for upcoming cycles, and the other with supervised models that predicted whether the cycle was abnormal or not. Deep Neural Network (DNN) model used in the study outperformed the other models.

Francis & Mohan explored the use of ARIMA for trend analysis and forecasting process (Francis & Mohan, 2019). Their proposed method used ARIMA for trend analysis on time series data. The predicted values were then passed through Principal Component Analysis (PCA) for feature reduction by removing highly correlated variables. Finally, the Support Vector Regression (SVR) algorithm was used for the maintenance forecasting approach.

While the performance of statistical approaches has been extensively studied in the literature, it is widely known that Long Short-Term Memory (LSTM) tends to outperform ARIMA when it comes to long-term dependencies. It has also been observed that LSTM and Gated Recurrent Unit (GRU) perform significantly better than ARIMA while producing similar predictions (Karpathy, Johnson, & Fei-Fei, 2015).

2.2. Data-driven artificial intelligence applications

In recent years, there have been numerous studies applying data-driven Artificial Intelligence models in the field of predictive maintenance and failure prediction from various perspectives (Del Buono et al., 2022; Nasir & Sassani, 2021; Soltanali et al., 2021). Some studies focused on failure prediction in the context of data classification (Coelho, Costa, Rocha, Almeida, & Santos, 2022; Lee et al., 2019), while others used machine learning methods as a regression task to estimate the time remaining until an error occurs (Chen, Shi, Lu, Zhu, & Jiang, 2022; Jiang et al., 2022; Lee, Kim, & Lee, 2022; Pacheco, Flesch, Flesch, Iervolino, & Barros, 2022).

In the study by Zhang et al. NASA's C-MAAPSS engine dataset was used to predict system performance degradation (Zhang, Wang, Yan, & Gao, 2018b). Using data from sensors commonly found in aircraft engines, the authors revealed the temporal dependence of sensor data in predicting engine degradation. Due to the nature of time series data, it is preferable to use deep learning-based models that are successful in capturing data dependency. The main focus of the study was the use of LSTM-based models for prediction as LSTM is capable of capturing long-term dependencies. They evaluated the proposed model and compared its performance with various ML algorithms, including SVR, Multilayer Perceptron (MLP), Relevance Vector Regression (RVR) and Deep Convolutional Neural Networks (DCNN) using 200 synthetically generated curves. They showed that the LSTM model performed significantly better than the others, and the model prediction performance increased, respectively, if the prediction curve decreased (Zhang, Wang, Yan, & Gao, 2018a). The authors compared various machine learning methods, as in the proposed framework. Unlike our study, they evaluated their approach using only simulation-based synthetic data, not real-world data.

Similarly, W.J. Lee et al. investigated the predictive maintenance algorithms in two different datasets, the cutting tool and spindle motor wear datasets. The milling cutter wear estimation is essential for cutting performance where an estimation of Remaining Useful Life (RUL) is required to avoid wasted material as well as faulty manufacturing. Since the spindle motor is a rotating element similar to a milling cutter, it is a vital component of production quality. The datasets in the study were collected until a system failure occurred. The authors applied two different classification algorithms, SVM and Convolutional Neural Network (CNN), to evaluate wear prediction performance. Using a confusion matrix, the predictions were classified into three groups: normal, warning, and failure. They reported that frequency domain results using CNN yielded highly accurate prediction results (Lee et al., 2019). While this study employed an approach to predict the remaining useful life using machine learning methods as in the case of our current

study, however, the authors simulated only two ML algorithms for predictive maintenance modeling on two experimental datasets.

In another study, Traini et al. proposed an approach for milling cutter maintenance. They provided a framework that combines both regression and classification algorithms in analog sensor readings to improve the human–machine interaction from a PdM perspective. The regression algorithms they used in this study were Linear Regression (LR), Decision Forest Regression (DF), Bayesian Linear Regression (BLR), Boosted Decision Tree Regression (BDT), and Neural Network Regression (NN). Classification algorithms used included Logistic Regression (LogR), DF, Decision Jungle (DJ), BDT, Neural Network (NN). The estimation of flank wear (VB) and RUL values was evaluated using regression algorithms, while the classification algorithms were applied to predict the type of samples as Safe and Worn. They found that the best results for regression were obtained in the NN regression models with the highest R^2 value, and the best results for classification were obtained using Two-Class BDT in terms of accuracy metric (Traini, Bruno, D'Antonio, & Lombardi, 2019). Although they have applied a similar approach in another domain to predict the remaining useful life of milling machines for predictive maintenance, their work has focused on a single historical dataset rather than a real-time implementation as in our framework.

Rivera et al. presented an approach for predictive maintenance of a hydraulic pump component in an injection molding machine. This study focused on the same data with a top-down approach, in which expert knowledge was omitted from the preparation process unlike their previous work (Rivera, Scholz, Fritscher, Krauss, & Schilling, 2018). Differently, their current estimation approach relies on anomaly detection based on a statistical method, kernel density estimation. Without depending on the knowledge of underlying machine physics, they fed the entire raw data into a local-outlier-factor (LOF) algorithm to detect unusual loops, after applying PCA. The detection of hidden patterns among the variables can be considered as one of the advantages of this approach. Ultimately, the authors noted the importance of expert knowledge, as reducing unnecessary data could improve performance by reducing computational efforts (Rivera et al., 2018). We also acknowledged the benefit of leveraging expert knowledge, particularly in the feature engineering and data interpretation phases, as predictive maintenance is a domain dependent subject.

From a similar perspective, Li et al. offered a data mining-based fault diagnosis framework consisting of five modules. They studied failure prediction and maintenance strategy optimization using online machine condition monitoring data at a remote customer site. Their study focused particularly on the prognosis of a backlash error. They have implemented error correction and maintenance application for practical use. As part of the data mining module, they developed an Artificial Neural Network (ANN) model to predict future errors based on data from the past three weeks. Although the proposed ANN model has capabilities such as fault tolerance, adaptability and generalization, the model is still data dependent and vulnerable to the influence of various factors. Similar to Rivera et al. (2018), they also stated the importance of expert knowledge for the development of an accuracy model (Li, Wang, & Wang, 2017).

The predictive maintenance problem has also attracted a significant number of research studies involving deep learning solutions (Canziani, Paszke, & Culurciello, 2017; Chen et al., 2020; Gensler, Henze, Sick, & Raabe, 2016; Mohammadi, Al-Fuqaha, Sorour, & Guizani, 2018; Xie, Wu, Liu, & Li, 2021). In Xie et al. (2021), Xie et al. emphasized the importance of deep learning in IoT and Smart City applications containing time series data. In particular, the predictive performance of LSTM networks was stated in time series data due to LSTM's ability to preserve long-term dependencies. They developed an LSTM model combined with Gaussian Naive Bayes to detect anomalies for their use case. Later, they evaluated the performance of the proposed combined model in comparison to an LSTM model and an MLP model by using three different real-world datasets.

From a different perspective, Chen et al. proposed a new approach called Cox proportional hazards deep learning (CoxPHDL) to address data resilience and data censorship that are commonly cited in the analysis of operational maintenance data (Chen et al., 2020). Their main idea was to offer an integrated solution by leveraging deep learning and reliability analysis. In the first phase, an autoencoder was adopted to transform the nominal data into a robust representation. Secondly, a Cox proportional hazards model (Cox PHM) was utilized to estimate the Time Between Failure of censored data. Lastly, a long short-term memory (LSTM) network model was developed to predict the Time-to-Failure based on preprocessed maintenance data. They found that LSTM provided the best results on a real-world dataset compared to the RNN, ANN, DCNN, and SVM models.

From a predictive maintenance point of view, Canziani et al. examined the performance requirements in a real-world deep neural network application. Their findings showed a detailed comparison of the performance requirements of a deep learning application while preserving efficiency. They noticed the relationship between accuracy and computational cost, and a small amount of increase in accuracy resulted in a gradual increase in computation time (Canziani et al., 2017). These findings are in line with the approach in our study, which takes into account computational power requirements and processing times for model selection. Data-driven machine learning algorithms are evaluated in this study because of their applicability in real-time prediction applications.

Predictive maintenance systems face different challenges depending on the field of application. Gupta et al. explored the feasibility of applying a machine learning based predictive maintenance system to baggage handling conveyors at airports (Gupta, Mitra, Koenig, Kumar, & Tiwari, 2023). The authors drew attention to the problem of random noise captured by IoT sensors due to the movement of luggage on the interconnected components of the conveyors that they encountered. In our study, we also observed the problem of random noise generated by high-frequency IoT sensors in our use case production environment (Ayvaz & Alpay, 2021). Additionally, we had to take into account the latency and scalability of the approach due to the issues of high dimensionality and complexity of the manufacturing assembly lines.

Recently, Ayvaz and Alpay have proposed a data-driven predictive maintenance framework for assembly lines producing customer hygiene products in their most recent work. The framework uses a set of machine learning models to detect signals in data generated from IoT sensors in real-time before potential failures occur. They focused on the scalability of the proposed architecture, pointing to the performance of the implemented algorithms. It is noteworthy that the ensemble learning algorithms used, Random Forest (RF) and XGBoost (XGB), achieved the highest R^2 scores of 0.982 and 0.979, respectively. Besides, they also pointed out another additional contribution of the implementation of the framework to the advancement of the facility's digital transformation (Ayvaz & Alpay, 2021).

In this article, we aimed to present a brief review of related studies with an emphasis on IoT-based predictive maintenance using machine learning. For a more comprehensive review of relevant predictive maintenance studies, we refer interested readers to the following literature review papers (Carvalho et al., 2019; Rieger, Regier, Stengel, & Clarke, 2019; Zonta et al., 2020). Rieger et al. provided a review of real-time IoT-based data processing using deep learning algorithms for predictive maintenance (Rieger et al., 2019). In another study, Zonta et al. offered a comprehensive survey of Industry 4.0 based predictive maintenance approaches to categorize the applications, standards, and methods (Zonta et al., 2020).

3. Methods & analysis

An overview of the process flow of the proposed approach is presented in Fig. 1. In the proposed approach, instant data from Internet of Things (IoT) sensors embedded on production lines in the factory

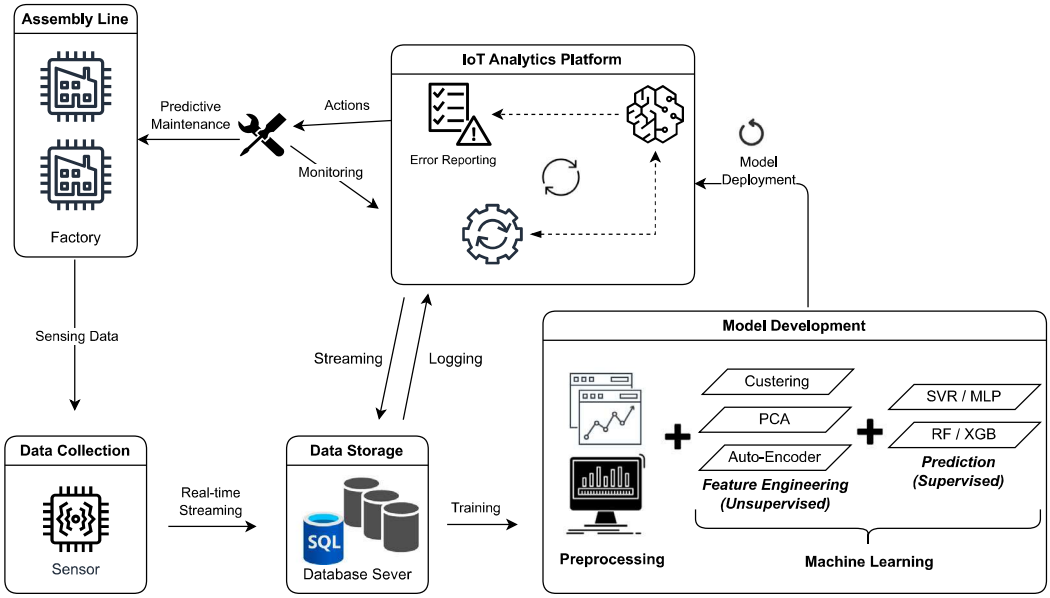


Fig. 1. Process flow of the proposed system.

are transmitted to a database for storage. Simultaneously, the recorded data are sent to the pre-trained machine learning model deployed on the live system for real-time predictions. Based on data from sensors, the ML model in the system automatically detects signals for potential stops and guides operational preventive actions for upcoming stop.

3.1. Data collection and preprocessing

The data used in this study were collected from a real-world assembly line that manufactures consumer hygiene products. Each production line is integrated with various IoT sensors that measure and collect various sensor readings. These sensors measure the instantaneous change of relevant values in the production environment such as weight, speed, temperature, electric current, vacuum and air pressure over time. Each sensor collects data in periods of 3 to 6 s.

In this study, two complementary datasets, sensor data and unplanned stop data, were used. The sensor dataset was collected and stored in a database for one year. This dataset consisted of 101 features and 8,668,431 instances. Of these features, 50 were sensor readings and 50 timestamps were related to each sensor type in this dataset. The dataset also contained an output feature named status, which represented the state of the line. A value of zero means the production line is running and any non-zero value indicates the line is not working properly. This variable was used as a control mechanism in the analysis.

Real-time sensor data from production lines do not have RUL values until the next unplanned downtime. Because a production error occurs unexpectedly and data can only be collected retrospectively, after the error has occurred. Consequently, this prediction problem cannot be addressed with a real-time time series approach, as RUL values are not available at the time of sensor readings.

The second dataset used in this study was the stops dataset, which consisted of 34 columns and 6787 rows. This dataset was generated from the production lines, where any instances of production stops were noted over the course of a year with various features including the cause of stop, time, duration, team, line, status (if planned or not), and comments.

The dataset had 27 unique failure types that caused a production stop. 228 of these were related to a single type of failure, which was the most common to avoid on the production line. Due to the nature of the problem in question, this makes the prediction task very challenging, as having 228 stops per year is a small number for model training.

3.1.1. Data cleaning and normalization

Due to the nature of IoT sensor data, the value ranges of features in the dataset vary widely. Therefore, it requires data normalization before modeling. Min–max normalization was applied to the sensor values by using the scikit-learn library.

On the other hand, most of the features were completely populated in the dataset. Approximately one percent of the data was missing only. It was not clear what the reasons for the missing data were. It is anticipated that the missing values may have occurred due to connectivity issues between IoT devices or temporary sensor measurement issues. Since a very small number of samples contained missing values, the instances were filled with median values of the corresponding feature before model fitting.

The database containing the sensor readings provided UNIX timestamps with the stops, recording the time information in the local time domain (Turkey GTM+3). This issue was noticed in the first trials of tests, where poor results were observed and thoroughly investigated. Actual error times in the stop dataset and the newly created dataset did not match. We were able to overcome this problem by converting the local time of the stops to UNIX timestamps.

3.1.2. Feature engineering

The prediction problem in this study can be treated simply as a classification task. However, it is not appropriate to predict the failure as a classification problem at the time of stop, but to detect warning signals in a continuous stream of data so that potential failures can be prevented before they occur. For this reason, the problem is considered as a regression task, which allows that necessary actions can be taken before malfunctions occur and unnecessary stops in the production line can be avoided. Therefore, the preprocessing task includes generating the necessary RUL values for the dataset. Given the size of the data stored, SQL was useful in handling data preparation tasks such as selecting and combining required features and creating the RUL feature.

The information required for RUL feature creation is as follows: i. StartTime: Start time and date of stop instance. ii. Downtime: Duration of stop in seconds. iii. Cause: Cause of the stop.

The next step in the preprocessing was to rank the samples to find the exact stop instance by calculating the difference between the Unix time of each stop and the timestamp of the sensor data. Because the sensors data were collected as a series of rows, the data rows had to be associated with the next production stop record to calculate the RUL

Table 1
Output of newly generated dataset.

Sts	Timestamp	a1	a2	a3	...	a16	a17	a18	RUL
0	1558138770	887.0	855.0	24210	...	1212.0	−16.1	40.9	37331
0	1558138776	886.0	852.0	2465.0	...	1212.0	−16.6	41.7	37325
0	1558138779	884.0	853.0	2406.0	...	1212.0	−16.6	41.5	37322
0	1558138785	885.0	852.0	2462.0	...	1212.0	−16.3	41.2	37316

value. The sensor readings were mapped to the next production stop by sorting and subtracting the time difference of the corresponding records as Unix timestamps.

After this step, each instance of stop was marked. The records that fall between start time and end time of stops were marked as error and thus have no RUL value. The remaining records in the sensor data were updated with the calculated RUL value.

To reduce the number of features in our dataset, the use of Principal Component Analysis was explored in feature engineering. Additionally, the effect of PCA on the prediction performance was compared against the features selected using expert knowledge. The model results were comparable in terms of prediction performance when using PCA. However, the PCA had a negative impact on interpretability of the results since PCA extracts new features out of original ones. Therefore, we proceeded with the list of features compiled by the experts instead of PCA in the study for better explainability and insights while alerting the operators for preventive actions. With the help of expert knowledge, a total of 18 sensor features related to the error type were selected and a new dataset was created to be used in the modeling phase, as shown in Table 1.

3.1.3. Alternative dataset – Stops removed

Another important observation during the preprocessing stage was that the sensors on the production line were constantly populating data, whether or not production is stopped.

Despite the data samples having a value of zero for RUL for the duration of each stop, the sensors on the production line continue to read data from the environment since the sensors remain connected. However, these measurements can be a misleading input for the trained model. For this reason, another dataset was created in which each row with a value of zero after the first stop was removed from the dataset before model training.

3.1.4. Dataset splitting

Splitting the dataset into training and test sets is a vital step to avoid potential over-fitting and under-fitting issues prior to modeling. To obtain reliable results, 70 percent of the dataset was used for model training while models were tested using the test set containing 30 percent of the original dataset. During hyperparameter tuning of the model training, 5-fold cross validation was applied to the dataset. A separate validation set was used in hyperparameter tuning to prevent potential data leakage to model fitting.

3.1.5. Correlation

While the sensors selected for the prediction problem are known for their direct association with the relevant error, detailed exploratory correlation analysis was performed. As shown in Fig. 2, several highly correlated features can be observed in the correlation plot.

This can be expected in our case because some sensors transmit information from the same production area where multiple sensors are placed and measure different features for detailed monitoring. As shown in Fig. 3, another dataset was created where highly correlated features were removed from the dataset for comparison purposes. This newly created dataset had eleven features, while the original dataset contained eighteen.

3.1.6. Developing autoencoder models

We examined the potential of autoencoders to capture the signal effect from sensor data before line stops. For this purpose, unsupervised autoencoder models were trained using sensor values at normal runtime with no errors from the production lines. Then, the autoencoder model was optimized using a validation dataset containing data from both normal runtime and error times. We integrated the reconstruction error value, which is the output of the obtained autoencoder model, into the prediction model as a new feature and trained the regression models. Consequently, we measured the effect of the autoencoder model on the prediction performance of the regression models. The details of the evaluation results are given in Section 4.2.6.

After evaluating several different options within the hyperparameter space for the models, the following parameters were found optimal under the specified conditions. For the XGBoost Regressor model, we noticed that the hyperparemeters of maximum depth of 5, the learning rate of 0.3 and the number of estimators of 100 were the best options. Likewise, for the training of the Random Forest Regressor model, various alternatives were tested to optimize the number of tree estimators. The model parameter with 51 tree estimators was found to be performing well. We evaluated different options for optimizing the hyperparameters of the autoencoder model. Based on our evaluations, we found that hyperparameters including the batch size of 256, three encoder layers with 18, 8 and 4 shapes, a bottleneck layer with two neurons, and three decoder layers with shapes symmetrical to the encoder, as well as the Exponential Linear Unit (ELU), activation functions and adam-optimizer were optimal. We trained the models over 100 epochs using the mean squared error as the loss metric.

3.2. Prediction models

The machine learning models were created using the Python programming language in this study. Keras and xgboost python libraries were used to develop the machine learning models.

3.2.1. Constituent models

Support Vector Regression. As a supervised learning approach, Support Vector Regression (SVR) is a popular algorithm for nonlinear problems as in this study. It can map data with high dimensional features to achieve good performance. In our case, an SVR model with a Radial Basis Function (RBF) core was used in comparison to a linear SVR.

Multilayer perceptron. MLP is a type of feed-forward NN that includes three layers: input layer, hidden layer and output layer. It is a powerful algorithm for solving nonlinear problems. MLP has many algorithmic hyperparameters that can significantly affect the prediction results. In this study, we examined various hyperparameters in model development. Hyperparameters of Relu activation function, adam optimizer, 50 hidden layer nodes and epsilon 1e−08 were found to fit well to the dataset based on hyperparameter tuning. More details about the hyperparameter tuning process are given in the subsequent sections.

3.2.2. Ensemble learning

Ensemble learning algorithms consist of a combination of multiple weak learners. The strength of the ensemble learning algorithm comes from the fact that the output of multiple weak learners can create a well-performing predictive model, even though the performance of these individual algorithms may not be high. These algorithms can be of three types: Bagging, Boosting and Stacking. The following subsections describe the ensemble learning algorithms used in this study.

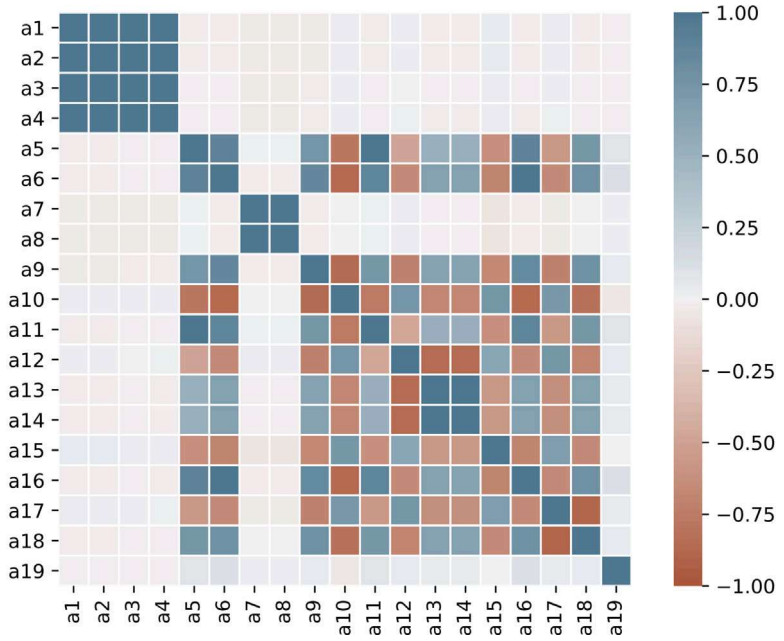


Fig. 2. Correlation plot of the dataset.

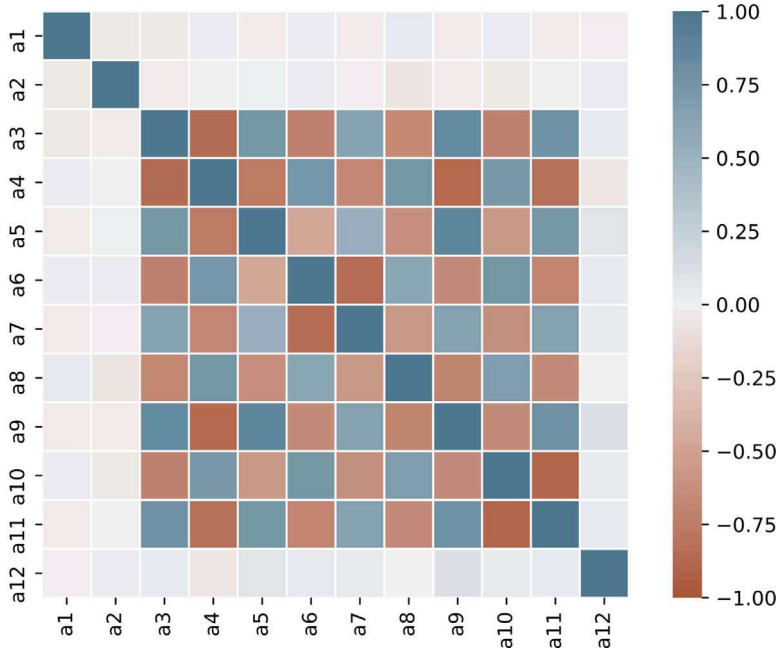


Fig. 3. Correlation plot after removing highly correlated features.

Random forest. RF is a bagging ensemble method consisting of multiple decision trees that can be applied to classification or regression problems. The main issue with decision trees is their tendency to overfit the data. RF addresses this problem by creating subsets of data and training each with different tree nodes. Each tree in the RF makes a prediction, and the resulting prediction is either the majority vote or the average of all predictions generated by the decision trees.

In this study, RF algorithm was used for training a regression model. The estimator parameter of 51 was found to perform well for RF model based on hyperparameter tuning with random search with 5-fold cross validation.

eXtreme gradient boosting. XGB is a type of optimized Gradient Boosting algorithm. Compared to RF, XGB creates trees with fewer splits and so it can overcome the overfitting problem. Due to its precision and efficiency, it is widely used in prediction problems. The results of hyperparameter tuning showed that XGB performed best at maximum depth of 5, learning rate of 0,3 and the number of estimators was selected as 100.

3.2.3. Clustering analysis

Due to the nature of the problem in this study, it is essential to estimate the RUL value until the next error rather than the class value if there is an error at the time of recording. However, it was important to examine whether the similarities between data records had a signal effect on the estimation of RUL. Therefore, unsupervised clustering algorithms were applied to the dataset as the output of the clustering can provide additional information for the prediction approach.

Elbow method was applied to determine the number of clusters. This method simply generates and calculates each number of possible clusters (k), and for each k the Within-Cluster Sum of Square (WCSS) was calculated. WCSS represents the sum of the distances between the centroid and each point in the cluster. The resulting plot provides an elbow-like graph from which the method gets its name. The point where the graph begins to continue almost parallel to the x-axis represents favorable cluster numbers. KMeans algorithm from scikit-learn library was used for clustering.

4. Results

4.1. Evaluations

In the study, four measurement techniques were applied to evaluate model performance results. The first metric was the coefficient of determination, also known as R-squared (R^2), a parameter commonly used for performance evaluation in regression models. It represents the ratio of variance explained for a dependent variable. The Root Mean Squared Error (RMSE) was another metric used, which describes how far the predictions disperse from the actual values. In other words, it represents the dispersion of the prediction according to the best fit (Hyndman & Koehler, 2006).

Another performance metric used in the evaluations was Mean Absolute Error (MAE), which measures the mean difference between actual and predicted values by removing their direction. Similarly, the Mean Absolute Percentage Error (MAPE) was used to express the mean absolute error as a percentage. The main advantage of this metric is the use of absolute error over percent deviation. Moreover, to help better interpret the model performance results, the inverse MAPE was calculated (Willmott & Matsuura, 2005).

4.1.1. Median filter

The median filter is a type of non-linear filter known for its performance in reducing noises in the dataset. In this study, it was used to eliminate the effects of random spikes caused by noisy inputs from sensors and to smooth the prediction results.

The Medfilt algorithm from the SciPy library was utilized to perform the median filter on the estimation dataset. In the evaluations, the kernel values define the size of the windows to calculate 10, 50 and 100 of the Kernel values used for filtering.

4.1.2. True Positive – False Positive approach

In the evaluations, we noticed that although the ML models were able to accurately predict some errors, they also produced a large number of false positive signals. Therefore, it was important to measure how well the models performed in scenarios where the RUL estimates were consistent before a failure and the alarms allowed ample time to take preventative action.

For the new assessment, “True Positive” (TP) and “False Positive” (FP) metrics had to be redefined in our case. As shown in Fig. 4, if the prediction model predicts the RUL value of 14400 (4 h in UNIX time), the timestamp of that instance is flagged. Starting from that timestamp until 4 h have passed, it is checked whether there was a stop within the four-hour time frame using the real data set. If so, it is marked as “True Positive” otherwise it is marked “False Positive”.

In order to verify the results, actual label values (RUL values) were checked. The main purpose of this evaluation was to identify models that increase the True Positive values as well as reduce the False Positive values.

For the reliability of the forecasts, it is essential that the production line reduces false alarms that cause unnecessary operations and, as a result, system stops for maintenance.

During production, the system was set to send an alarm if it detects a RUL value of 4 h or less. A new performance evaluation was performed that aimed to change the way alarms were sent and reduce FP alarms during production, taking into account the previous assessment results.

4.1.3. True Positive – False Positive approach with median filtering

To reduce the effect of high variation of predicted RUL values, we applied a data smoothing technique to the prediction results. Instead of choosing the instance of predicted RUL value, we considered the median of the last n number of predicted RUL values, where n is the window size for smoothing.

Our initial trials produced a high number of TP values compared to FP values, which was not expected considering the performance implications of the model. This was due to the methodology used to generate the RUL values in the preprocessing task, where RUL values were marked as zero within a downtime. In the calculation of TP values, repeating zero values in the RUL feature were used.

The process of calculating TP-FP scores after completing the prediction of RUL value for each record instance is illustrated in Fig. 5 and explained as follows:

i Calculate the median of “n” number of rows in the window.
ii If the median value of the “n” number of rows is between 0 and 4 h 15 m, check if there is a stop in the next 4 h duration in the original dataset (starting from *n*th row).
- If so, mark TP and return to ii for the next “n” number of rows in prediction.
- If not, mark FP and return to ii for the next “n” number of rows in prediction.

iii If not, return to ii and continue to the next n number of rows.

The selected median sizes were 50, 100, 150 and 200 rows. Each line represents about 3 s, with a total of 10 min of median RUL calculated at most.

As the dataset contains stops that last a long duration, the actual RUL values are the recurring rows of zero values, which causes the algorithm to generate a large number of TP values. To resolve the recurring TP values, the timestamps of each TP value were stored as an array and checked for unique timestamps of the TP values. By doing this, we were able to obtain unique TP values.

As a minor improvement in the computation process, we changed the behavior after TP detection, where previously the filter scanned the next n rows. With the modification, the process calculates the median of n rows immediately after the row where the TP value is determined. This last approach slightly reduced the TP and FP values, but increased the accuracy of the model performance.

4.1.4. Confusion matrix

To measure the real-world preventive power of the models, there was a need to link specific model signals to the observed stops. The evaluation metrics used previously were not actionable metrics. Using the confusion matrix distribution, we were able to calculate the sensitivity of the applied model, also called Recall, to measure model performance.

Our focus on a confusion matrix in this study pointed to TP and FP values. The TP value represents the situation where an actual error occurs as predicted by the model within the time frame of the prediction window. On the contrary, FP represents modeling error prediction when there was actually no error. In our case, the total number of stops and the number of positive values (P) were already known because the dataset included the stops that occurred. Using these three variables, we were able to calculate the Precision and Sensitivity of the prediction model.

4.2. Test set

4.2.1. Standard dataset

After the model fitting processes were completed and the most suitable parameters were determined for each model, the models created were evaluated for the test dataset. The evaluation results of the models

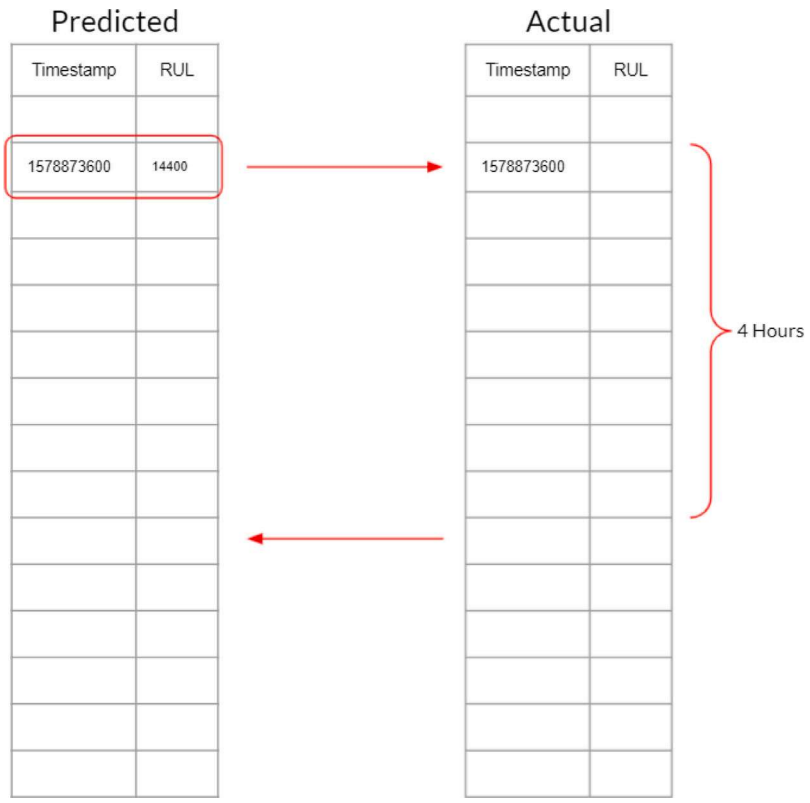


Fig. 4. True Positive–False Positive Approach with Median Filtering.

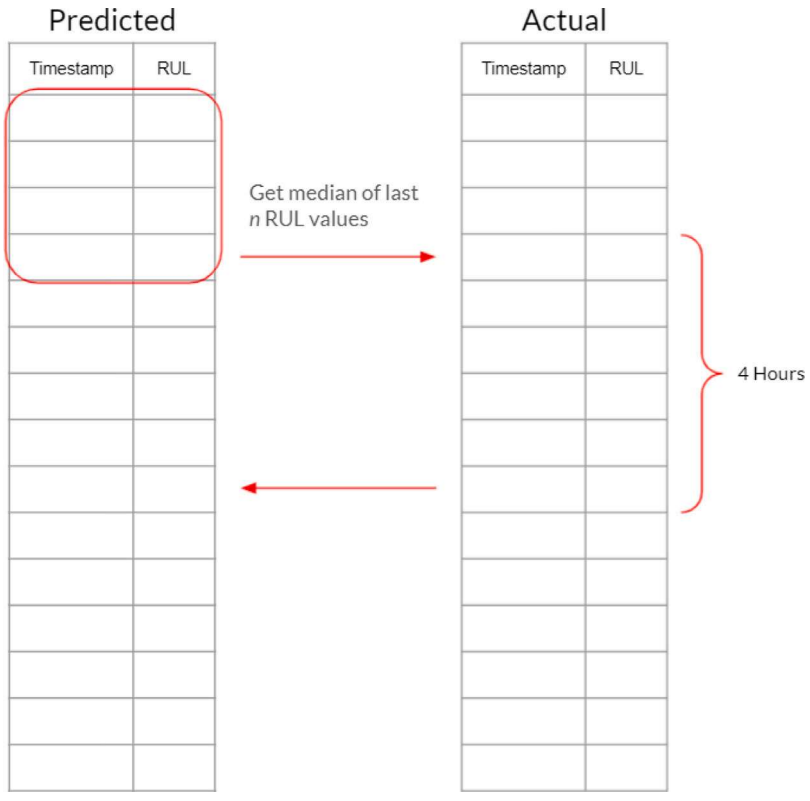


Fig. 5. TP-FP approach with median filtering script diagram.

are demonstrated in Table 2. As a model evaluation technique, inverse MAPE was calculated by subtracting the result from 100 and used to describe the model's performance in terms of an error-free percentage.

The best results were obtained using the Random Forests model, which seemed to fit the data significantly better, with an R^2 value close to one showing how accurate the prediction was (Table 2). This was

Table 2
Performance results of standard dataset.

Algorithm	R^2	MAE	MAPE	Inv. MAPE	RMSE
RF	0.990	8855.15	2.09	97.91	45316.91
XGB	0.650	173771.26	41.04	58.96	272467.00
SVR	−0.435	407963.56	96.3	3.7	551882.41
MLP	0.450	234300.15	55.31	44.69	341651.11

Table 3
Performance results of stops-removed dataset.

Algorithm	R^2	MAE	MAPE	Inverse MAPE	RMSE
RF	0.994	4895.14	1.12	98.88	37567.86
XGB	0.744	167505.2	38.49	61.51	253658.63
MLP	0.695	192165.52	44.15	55.85	276604.36

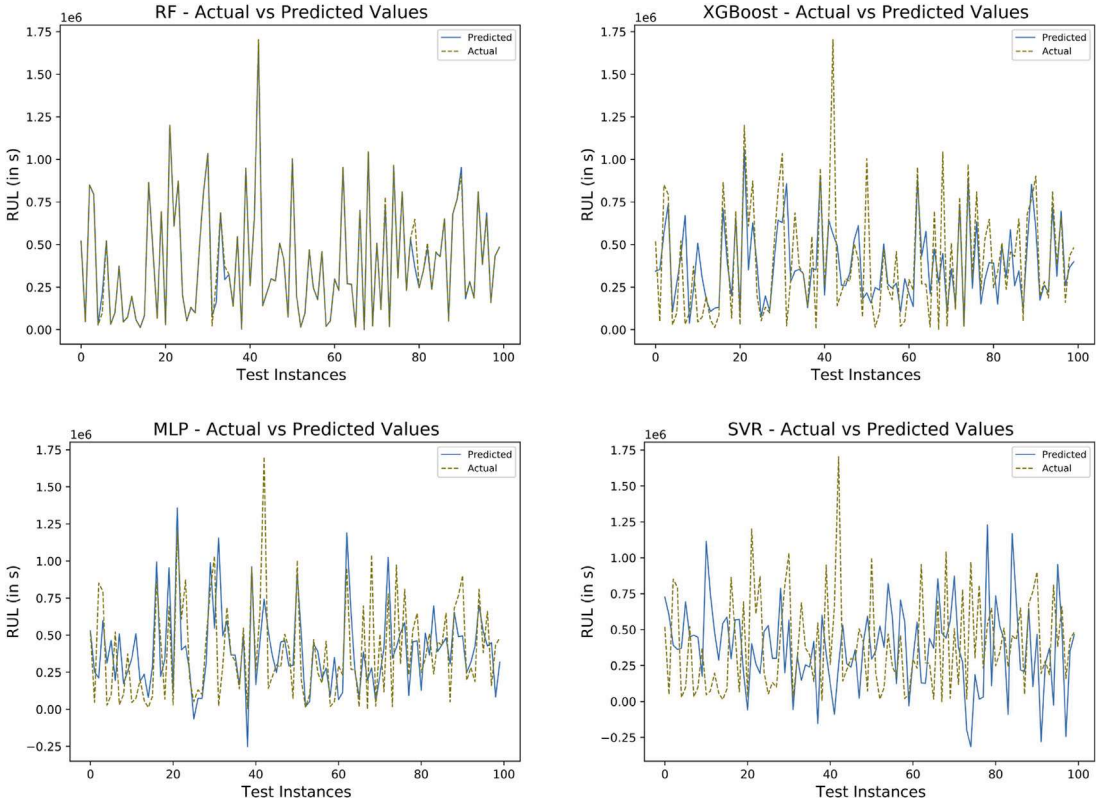


Fig. 6. Predicted vs. actual plot of RUL for standard dataset.

followed by XGBoost, which achieved an R^2 of 0.65. Similarly, when MAPE scores were compared, the RF model outperformed the others with a score of 2.09 percent.

Model prediction results were plotted and observed for detailed analysis. From the actual and predicted plots shown in Fig. 6, we observed how well the RF model fit the model and plotted a prediction line similar to the actual RUL values. These prediction plots also showed potential in prediction, although XGB and MLP scored relatively lower. The worst performance was obtained with the SVR model and was not used in further tests.

4.2.2. Stops-removed dataset

When examining the poor performance of MLP and SVR compared to RF and XGB, it was clear that RUL values of zero for downtime were misleading some models. The Stops-Removed dataset was used to generate and predict model results for comparison purposes. Because this dataset contained only stop occurrence instances rather than the entire downtime, filtered information can improve the performance of the prediction model.

This problem was addressed after reviewing the sensor readings during stops. It was observed that some sensor readings did not change during the stops, while others changed during the observations. This issue can cause a promising prediction model to fail to fit. The results of the stops-removed dataset are shown in the Table 3, where improvements were observed in all models supporting the approach for the filtered dataset. The evaluation results of the filtered datasets, including Stops-removed, Correlations-removed, and Stops-removed and cluster applied, are demonstrated in Figs. 7, 8 and 10, respectively.

4.2.3. Correlations removed dataset

The third approach for the test set continued with the use of the correlations removed dataset in which the high correlated features were removed. The test results for the dataset are demonstrated in Table 4.

Compared to previous models, a slight decrease in prediction performance was observed. Similar to our first two trials, the RF model fitted the test set well. XGB, on the other hand, scored its the lowest performance in the evaluations when using the correlations removed dataset.

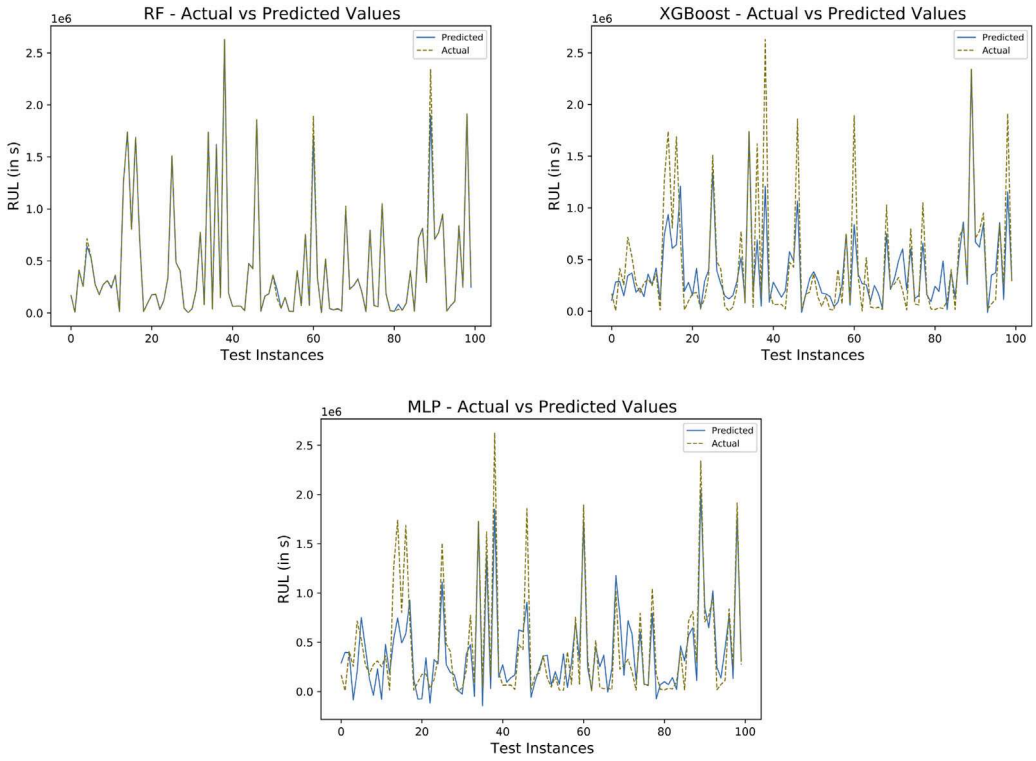


Fig. 7. Predicted vs. actual plot of RUL for stops-removed dataset.

Table 4
Performance results of correlations removed dataset.

Algorithm	R^2	MAE	MAPE	Inverse MAPE	RMSE
RF	0.99	9093.31	2.15	97.85	46234.43
XGB	0.626	181337.89	42.81	57.19	281839.40
MLP	0.467	235424.8	55.57	44.43	336252.33

4.2.4. Stops-removed and clustering applied dataset

Our next approach was to measure the impact of clustering on model performance. The purpose of this approach was to add external information to the dataset using internal dynamics. We aimed to observe the cluster distributions and add the cluster label as a new feature to the dataset. Clustering methods are often used in the literature as classification approach to maintenance. Since our problem was based on regression and RUL prediction in terms of time interval, a meaningful class distribution could lead to improvements in prediction performance.

By calculating the WCSS score and using the elbow method, we were able to determine the number of optimal candidate clusters present in the dataset distribution. Fig. 9 shows the number of clusters explored during observations and their corresponding WCSS scores.

When the original dataset was used for clustering, the resulting graph did not provide a clear cluster breakpoint. The reason behind the creation of the stops-removed dataset mentioned in Section 4.2.4 can help explain this observation. In this regard, the stops-removed dataset was used for clustering and there were three significant clusters as illustrated in Fig. 9.

The created cluster labels were added to the dataset as a new feature and the models were developed using the dataset including this extra attribute. Meanwhile, the cluster model was also stored and label values were added to each new row for post-testing and predictions were made using this new feature as shown in Table 5.

The results regarding the clustering of the applied dataset were very promising as performance improvements were observed in each model. This showed that our clustering approach was effective. Checking the plots for predictions versus actual data, both XGB and MLP better fitted the actual values compared to previous approaches, keeping RF performance nearly the same with a slight improvement.

4.2.5. PCA applied dataset

Our next approach was focused on dimension reduction to see if it was possible to obtain more meaningful results while decreasing the number of features and computational costs. When applying PCA, we set the proportion of variance explained to 97 percent. This resulted in the creation of 12 features, while our original dataset contained 18 features by taking expert knowledge into consideration. We compared predictive performance models using PCA-generated features and a feature set based on expert knowledge. The performances of the models developed by the newly created dataset are shown in Table 6. It was observed that similar performance outputs were obtained with RF, but other models showed poor results. Considering the interpretability issues of PCA features, we decided to exclude PCA from real-time implementation and move forward with a set of expert-derived features for better explainability of generated alerts.

Although variance for the PCA was selected 97 percent, it was observed that considering expert knowledge in determining dataset features resulted in better model fit and predictions. Although RF managed to predict salient results, XGB and MLP showed poor results as shown in Fig. 11.

4.2.6. Autoencoder applied dataset

After integrating unsupervised autoencoder (AE) models into our proposed framework, we evaluated the effect of AE in capturing signals from sensor data before the line stops. We compared the prediction performance of regression models with integrating autoencoder models

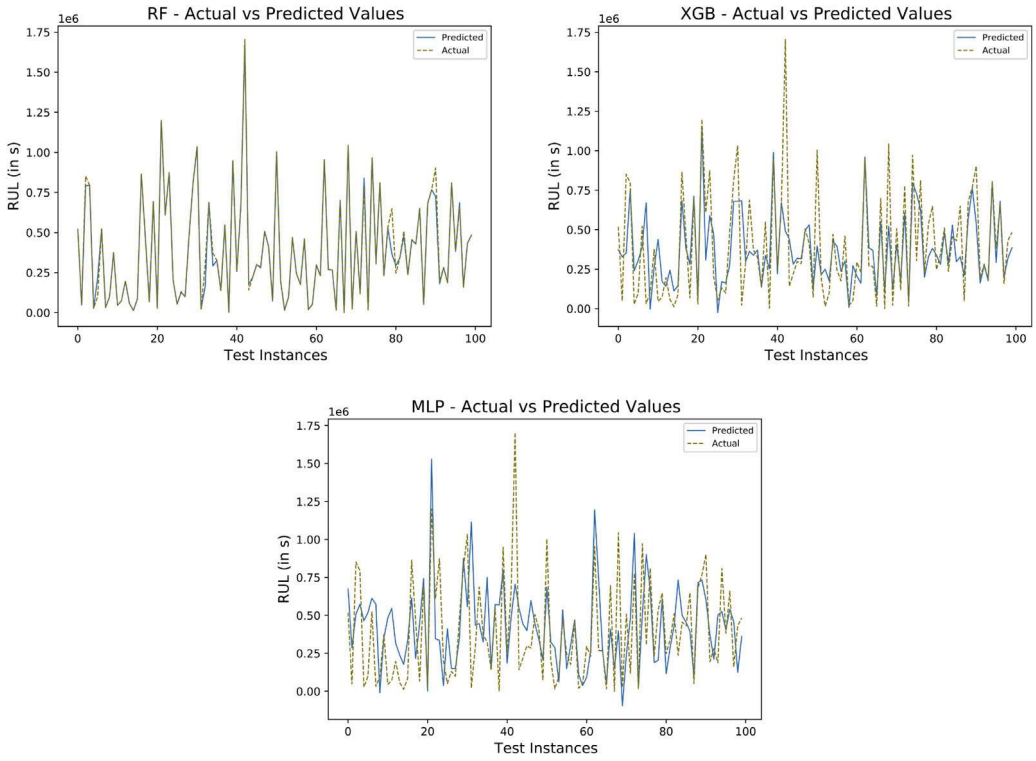


Fig. 8. Predicted vs. actual plot of RUL for correlations removed dataset.

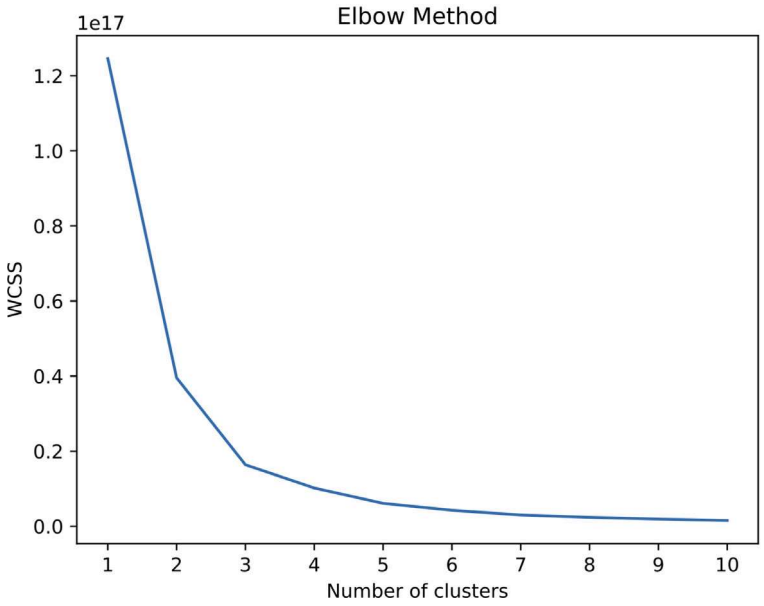


Fig. 9. Elbow method plot of WCSS vs. number of clusters.

Table 5
Performance results of stops-removed and clustering applied dataset.

Algorithm	R^2	MAE	MAPE	Inverse MAPE	RMSE
RF	0.994	4954.07	1.14	98.86	38118.35
XGB	0.748	165838.26	38.1	61.9	251425.99
MLP	0.709	186614.23	42.88	57.12	270425.70

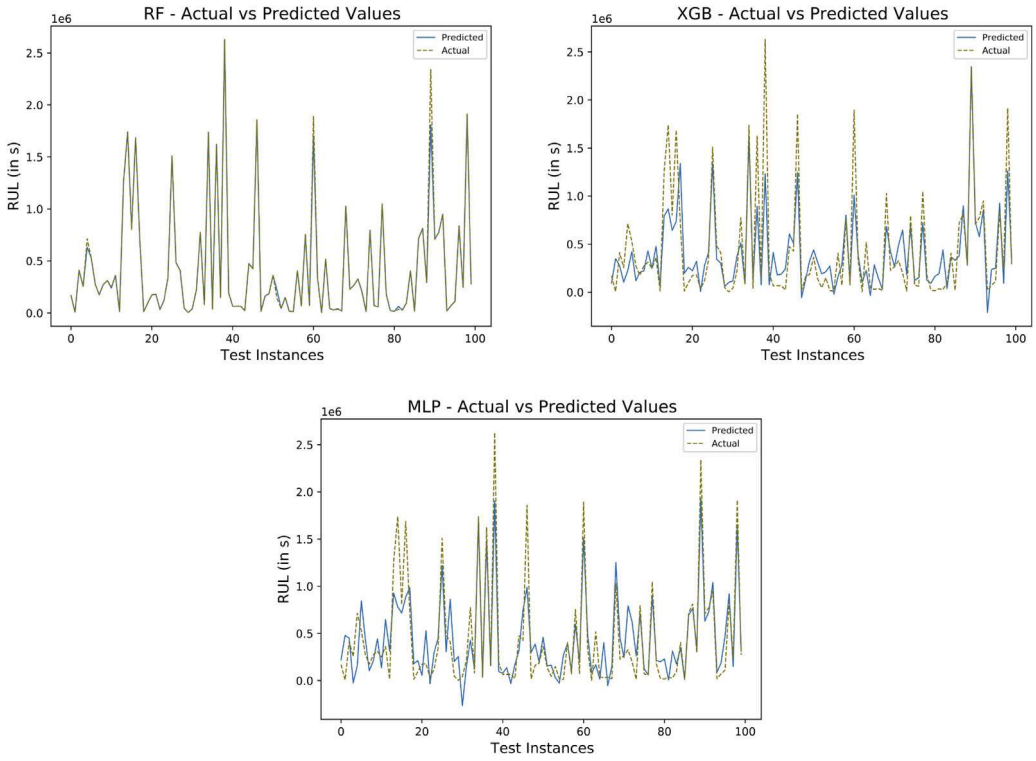


Fig. 10. Predicted vs. actual plot of RUL for stops-removed and clustering applied dataset.

Table 6
Performance results of PCA applied dataset.

Algorithm	R^2	MAE	MAPE	Inverse MAPE	RMSE
RF	0.98	9288.11	2.12	97.88	59635.33
XGB	0.083	199411.1	45.48	54.52	304102.19
MLP	−5.07	312793.55	71.33	28.67	427923.49

Table 7
Performance results of autoencoder applied dataset.

Algorithm	R^2	MAE	MAPE	Inverse MAPE	RMSE
RF	0.99	8896.01	2.1	97.9	37557.42
XGB	0.36	171668.92	40.52	59.48	45055.75
MLP	−0.211	231827.09	54.72	45.28	339401.20

against the standard dataset results as baseline. The test results with the AE reconstruction loss added dataset are shown in Table 7. The results showed that integrating the AE model is effective and the hybrid approach incorporating AE models was able to moderately improve the prediction results for all evaluated ML models.

The most important output of AE models can be explained by the results of reconstruction loss. When examined, we expected to see high peaks close to the unplanned stop points. Since we trained our model excluding stops with three-hour intervals, these stop points were causing anomalies in the model. In Fig. 12(a), it can be noticed that a high peak is generated in terms of reconstruction loss, indicating an upcoming stop. It was also observed that when two stops occur in a short time interval in the dataset, the first stop adds a random noise and distortion to the upcoming stop as demonstrated in Fig. 12(b).

Similar to the standard dataset results, the RF model prediction performance was the most promising among other models as presented in Fig. 13.

4.3. New validation set

After completing the evaluations on the existing dataset, we were able to obtain a new dataset from the following month and used it to validate the results of the aforementioned models. The new dataset was preprocessed in the same way as previously described. The results were unexpected for the standard dataset. The model performance results indicated how poorly each model performed in the prediction as demonstrated in Table 8.

These results required detailed monitoring of the predictions to find a meaningful explanation. After observing the graphs, we were able to see how the model managed to fit the actual values in some intervals. While RF clearly achieved the best fit among the other models, the trend in the prediction was somewhat representative of the actual values in the other models shown in Fig. 14.

Similar results were obtained when other models were checked for the remaining three dataset approaches. Consequently, those results were omitted from this study as no additional information was obtained.

4.4. New evaluation approach

Due to the unsatisfactory performance results in the new validation test set, our focus shifted to the evaluation of model performances.

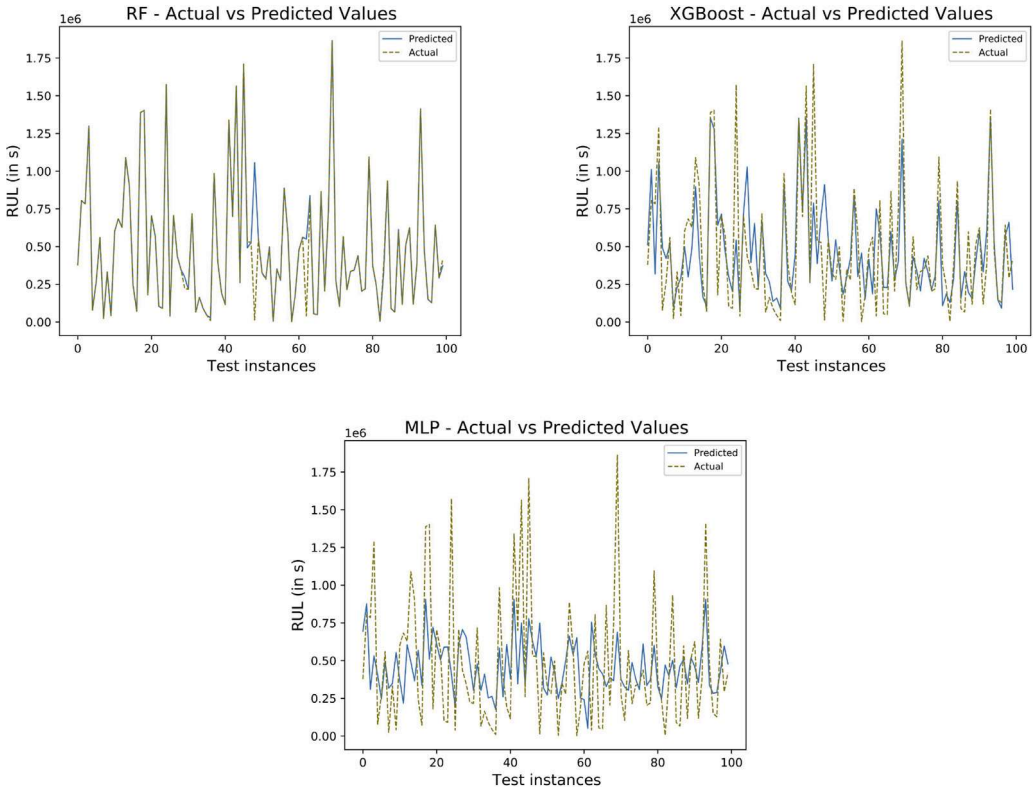
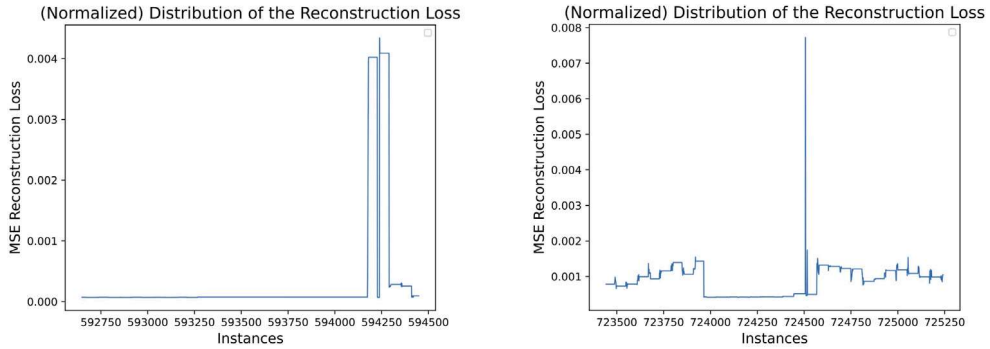


Fig. 11. Predicted vs. actual plot of RUL for PCA applied dataset.



(a) Instances before 6^{th} stop (b) Instances before 11^{th} stop

Fig. 12. Two sample plots for Reconstruction loss before unplanned stops.

Table 8
Performance results of newly generated validation set.

Algorithm	R^2	MAE	MAPE	Inverse MAPE	RMSE
RF	−2.15	210479.27	121.56	−21.56	286963.58
XGB	−0.295	140958.46	81.41	18.59	183876.99
MLP	−222.02	2079551.73	1200.98	−1100.98	2413377.23

4.4.1. Median filter

After completing the model performance evaluations using the training and test sets and deploying the models, the newly created validation set was used for real-time prediction. Using the preferred prediction model with the newly created 1-month dataset, we were able to see the model prediction performance results shown in Fig. 14.

The variance of the prediction compared to the actual RUL values resulted in poor prediction results. Since the predicted RUL values decreased linearly and the prediction results were not distributed as expected, new methods were applied. Random spikes in prediction caused the model to perform poorly. Our first approach to tackling this problem was to use median filtering.

The median filtering was applied to the results after the predictions to smooth out the predicted RUL values as well as the overall distribution. However, since the predictions were already unevenly distributed, it was not sufficient to simply use a median filter for improving model performance in cases where random spikes were still present as shown in Fig. 15.

Different parameters were explored for the median filter. These parameters represent the number of rows used to calculate the median

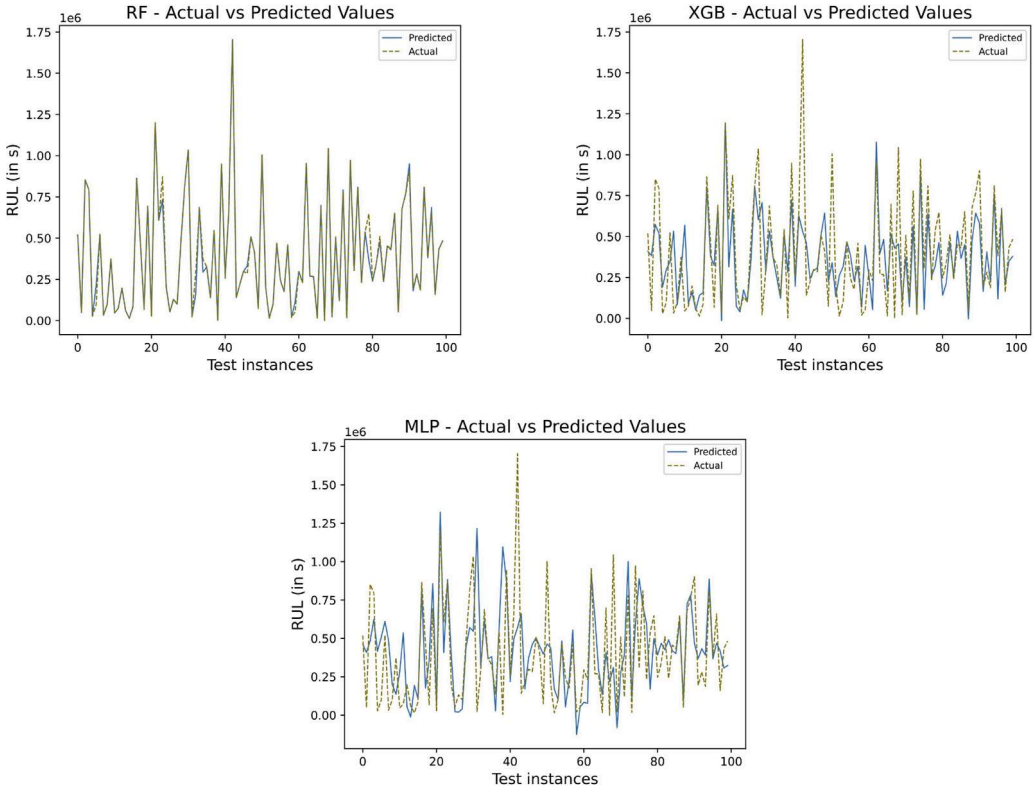


Fig. 13. Predicted vs. actual plot of RUL for autoencoder applied dataset.

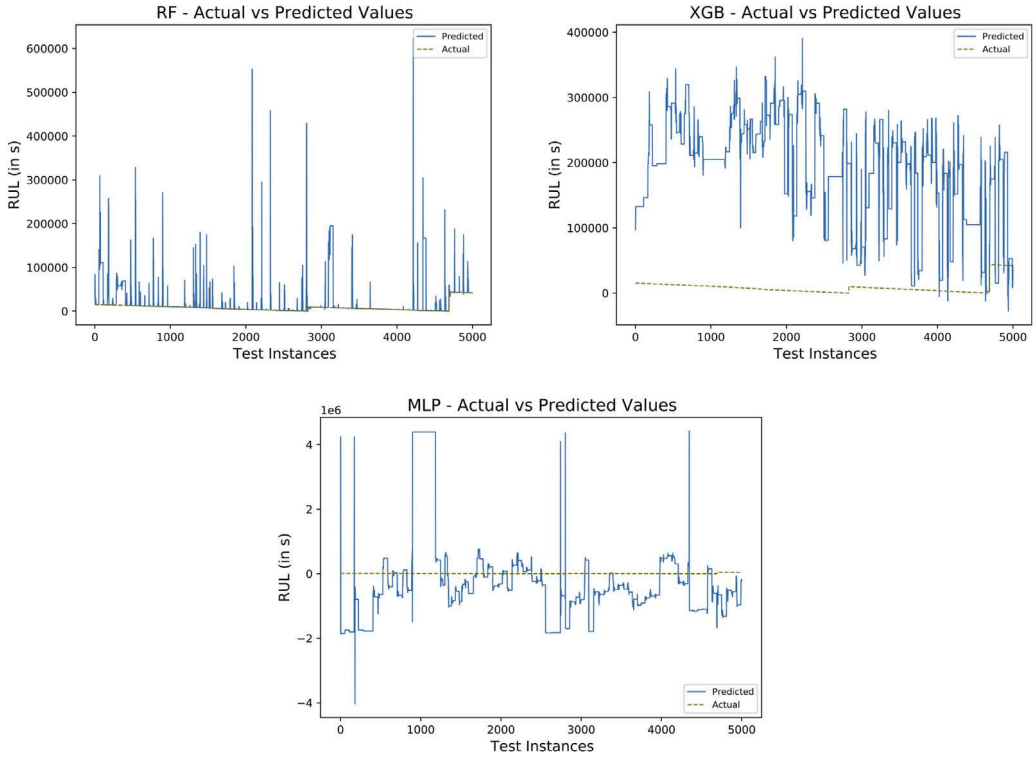


Fig. 14. Predicted vs. actual plot of RUL for new validation set.

and applied to the prediction results. Increasing the median range resulted in a smoother treatment on the RUL values, but had a high impact on the data as it changed the results more drastically than anticipated. Lower values of median parameters performed poorly.

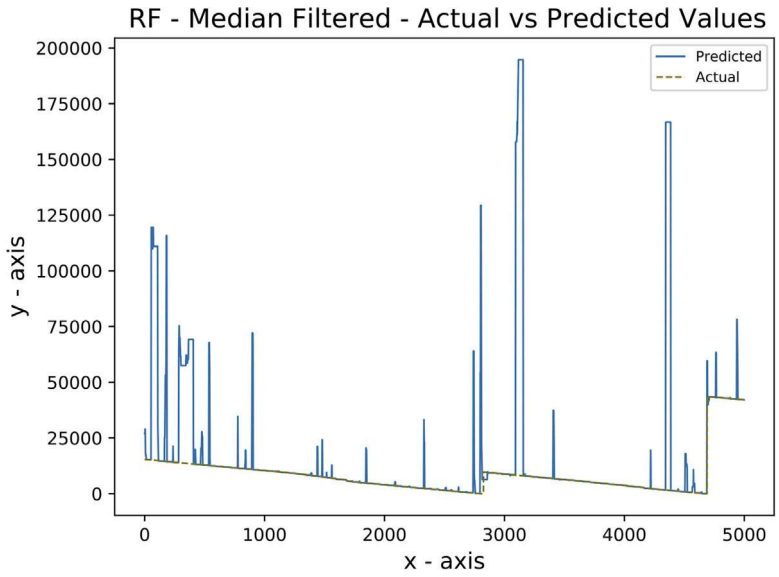


Fig. 15. Median filter applied prediction results for RF model.

Table 9
Validation set TP-FP results.

Model	TP	FP
RF	234	667
XGB	33	3834
MLP	5	557

Using the results obtained from the median filtered RUL values, no significant change was observed in the performance of the model. Our next approach in this issue was to focus on computing True-Positive and False-Positive values.

4.4.2. True Positive – False Positive approach

The fact that the median filter alone did not address the high variance in prediction results using the new validation set shifted our attention to defining global True and False variables. This task was a method of filtering out false alarms while providing appropriate RUL predictions as signals for the assembly line. Another output of this task was to generate an evaluation metric for model performance and calculate sensitivity using the confusion matrix technique. The results for the base model are given in Table 9.

As it can be seen from the results of the TP-FP task, the method had to be rearranged because meaningful outputs could not be obtained. The high figures in TP and FP values were inconsistent. When the algorithm was checked, the flaws in the approach were fixed.

4.4.3. True Positive – False Positive approach with median filtering

As noted in the methodology, this approach was a new observational approach in our study, where our concern was the overall accuracy of the model for RUL prediction rather than the correct prediction of each sample. This new approach also brings implementation details into focus. Since the dataset used in this study was from real-world production lines, the approach had to be implemented and useful in real-time as well.

The evaluation results reported in Tables 10, 11, 12, and 13 were obtained using RF models only. The MLP and XGB models failed to achieve as promising results as the RF models. This was not surprising given the previously observed estimates for the new validation set. Four different modeling approaches were used for this new methodology to observe model performance results. Each model was presented in its respective table (Tables 10, 11, 12, and 13).

The tables show four different window sizes of the median values used to calculate TP-FP values. Each model was tested for four different RUL values as a time interval. These values represent the actual behavior of the model running on the production line. The minimum time interval selected for model alarm was 1 h, and the maximum was 3 h. Unique values represent the amount of unique stops detected from the original dataset. The dataset tested above contains 29 unplanned stops.

At first glance, it was observed how the stops-removed dataset, to which cluster labels were added, performed poorly, detecting four TP values as the best prediction while producing a high number of FP values. Using the confusion matrix metrics and calculating the Precision and Sensitivity of all results, it was observed that the predictions of RF model generated from the stops-removed dataset outperformed the other alternatives.

The best results were obtained using the median size of 50. In terms of time intervals, 1 h and 1,5 h results were similar to each other and produced the most accurate predictions.

5. Discussion

In this study, a predictive maintenance approach is proposed for predicting RUL of a real-life production line. By using machine learning algorithms trained on real-world datasets collected from integrated IoT sensors, we aimed to detect possible future failures on assembly-lines, and an early maintenance task was scheduled for preventive maintenance.

The first decision to be made in our study was to decide how to make the prediction approach. In our real-world scenario, where the model would make predictions based on a data instance, it would not be appropriate to choose a classification approach as it does not allow sufficient time to take preventive action before the error occurs. To obtain a suitable preventive model, we focused on the regression task and estimated RUL in the time domain.

As a result of our discussions with the experts on the production line, it was decided that it is important to know an error up to 4 h before it occurs. Therefore, for the "True Positive" (TP) metric, we considered the 4 h threshold for prediction accuracy.

While a large body of the literature reviewed has focused on time series data analysis in this area, this was not a viable approach in our case. Because it is not possible for the model to verify the occurrence of a failure in real-time. Four different machine learning algorithms including two ensemble and two constituent algorithms, namely RF,

Table 10
Main dataset RF model prediction results on validation set.

RUL alarm	Median size	TP	FP	Unique TP	Unique FP	Prec.	Sens.
3600 s (1 h)	50	73	1	11	1	0,917	0,379
	100	36	1	10	1	0,909	0,345
	150	20	1	7	1	0,875	0,241
	200	19	0	8	0	1,000	0,276
5400 s (1,5 s)	50	113	3	12	3	0,800	0,414
	100	58	2	10	2	0,833	0,345
	150	34	2	7	2	0,778	0,241
	200	28	1	8	1	0,889	0,276
7200 s (2 h)	50	155	5	13	5	0,722	0,448
	100	79	2	10	2	0,833	0,345
	150	49	3	7	3	0,700	0,241
	200	39	1	8	1	0,889	0,276
10800 s (3 h)	50	222	5	14	5	0,737	0,483
	100	113	3	11	3	0,786	0,379
	150	71	2	7	2	0,778	0,241
	200	57	1	9	1	0,900	0,310

Table 11
Stops removed dataset RF model prediction results on validation set.

RUL alarm	Median size	TP	FP	Unique TP	Unique FP	Prec.	Sens.
3600 s (1 h)	50	47	1	12	1	0,923	0,414
	100	21	1	8	1	0,889	0,276
	150	13	0	6	0	**1,000**	0,207
	200	11		5	0	**1,000**	0,172
5400 s (1,5 s)	50	84	1	12	1	0,923	0,414
	100	40	1	8	1	0,889	0,276
	150	25	0	7	0	1,000	0,241
	200	18	0	6	0	1,000	0,207
7200 s (2 h)	50	123	2	12	2	0,857	0,414
	100	58	1	8	1	0,889	0,276
	150	39	0	8	0	1,000	0,276
	200	28	0	7	0	1,000	0,241
10800 s (3 h)	50	182	4	13	4	0,765	0,448
	100	89	1	9	1	0,900	0,310
	150	58	1	8	1	0,889	0,276
	200	43	0	7	0	1,000	0,241

Table 12
Correlations removed dataset RF model prediction results on validation set.

RUL alarm	Median size	TP	FP	Unique TP	Unique FP	Prec.	Sens.
3600 s (1 h)	50	75	2	13	2	0,867	0,448
	100	36	1	9	1	0,900	0,310
	150	22	2	8	2	0,800	0,276
	200	19	1	8	1	0,889	0,276
5400 s (1,5 s)	50	116	2	13	2	0,867	0,448
	100	57	1	9	1	0,900	0,310
	150	36	2	8	2	0,800	0,276
	200	27	1	8	1	0,889	0,276
7200 s (2 h)	50	154	4	13	4	0,765	0,448
	100	75	1	9	1	0,900	0,310
	150	50	3	8	3	0,727	0,276
	200	37	1	8	1	0,889	0,276
10800 s (3 h)	50	224	10	14	10	0,583	0,483
	100	111	5	11	5	0,688	0,379
	150	72	4	8	4	0,667	0,276
	200	54	3	8	3	0,727	0,276

XGB, MLP and SVR, were selected and trained to predict possible failures.

To assess the performance of the models, four datasets were collected for different evaluation cases. An additional validation dataset was created to test the model implemented on the real assembly line. The model prediction results showed the performance of the RF algorithm, which produced the highest R^2 value in all cases. We think that this was due to the complex nature of the model, which in our case created multiple decision trees resulting in better data fitting.

With the introduction of the newly created dataset, we had the opportunity to test our models on the new validation set. The performance results were unexpectedly low when compared to the model evaluations on test sets. We further investigated the prediction results to understand the problem. While the problem of overfitting in models is often the first thing to consider for the issue, examining these results revealed that this was not the case.

When the prediction plots were analyzed for the new validation set, a remarkable model fit was observed for the RF models. Generating linearly decreasing RUL values until the next error date, as in our RUL

Table 13
Cluster added dataset RF model prediction results on validation set.

RUL alarm	Median size	TP	FP	Unique TP	Unique FP	Prec.	Sens.
3600 s (1 h)	50	24	19	3	19	0,136	0,103
	100	11	9	2	9	0,182	0,069
	150	7	7	2	7	0,222	0,069
	200	5	5	2	5	0,286	0,069
5400 s (1,5 s)	50	39	40	3	39	0,071	0,103
	100	19	20	3	20	0,130	0,103
	150	13	13	3	13	0,188	0,103
	200	9	9	2	9	0,182	0,069
7200 s (2 h)	50	58	59	3	58	0,049	0,103
	100	28	30	3	30	0,091	0,103
	150	18	21	4	21	0,160	0,138
	200	14	14	3	14	0,176	0,103
10800 s (3 h)	50	92	86	3	84	0,034	0,103
	100	46	41	3	41	0,068	0,103
	150	31	29	4	29	0,121	0,138
	200	22	21	3	21	0,128	0,103

generation method, does not always reflect the real-world randomness of errors. Actual faults that occur on the production lines are sometimes caused by a raw material feed instance and have nothing to do with equipment defects. Considering the problem in this way, it can be said that our previously trained models learned the data as well as the error states.

The reason for the unsuccessful results from the RF model can be explained by the high amount of variance in the prediction results observed from the plots. Given that the model fits the plot, our focus shifted to assessing the real-world value of the prediction results and we introduced the TP-FP approach, a new layer in RUL prediction. In other words, how many of real errors could be avoided by using alarms generated by models.

This approach has two important aspects. First, we try to predict the RUL of the assembly line by providing the relevant results to take the necessary actions. It was observed that constituent ML algorithms were not sufficient for such real-world production datasets, where unpredictable effects were present in the dataset compared to a synthetic dataset. Second, it is not enough to simply present a prediction model in situations where human–machine interaction and communication is vital, which is one of the most important findings of this study. Our new approach can be seen as a framework definition where we try to minimize the effects caused by the real-world system and provide an information technique based on ML results.

Our initial approach resulted in inconsistent values as many true or false alarms made no sense. The problem with the first approach was caused by the filtering method, where we scan the dataset line by line after each FP value obtained. Although this detail was mentioned and corrected, we had not yet achieved practical results. Based on our observations of the predictions, variation in the prediction distribution led the approach to filtering techniques, but models with filtering were simply not sufficient to yield promising results.

Our final approach was to combine these two methods described above to create a final evaluation method. Unlike the previous TP-FP approach, we introduced median window sizes in this new method. While we could not obtain promising results from the median filter alone, we decided to use the median filter for prediction result smoothing before the TP-FP control. This new method showed promising results, where we were able to obtain a value of 0.923 for the precision of the results.

A key finding of the study was that we realized that AI models were able to capture meaningful, actionable inferences in the IoT data obtained from the production lines. After implementing the model in the production facility, it was observed that the model was able to accurately predict more than 50 percent of unplanned production stops. This means that half of the unplanned downtimes that occur on the lines due to equipment failures were prevented by the system. Consequently, 50 percent savings were achieved in product discards, which is a significant cost. The proposed approach not only helps to reduce costs and increase production efficiency but also contributes to sustainability by reducing the hundreds of thousands of discarded products per year.

Although the machine learning models are data dependent and cannot be directly used in all manufacturing conditions, we think that the proposed methodology can be generalized and applied to similar production environments.

Considering the size of our datasets and the computational power required for training, complex deep learning models were omitted from the study due to limitations on available computational resources. For future work, more computationally complex deep learning models will be investigated to compare with the current work.

6. Conclusion

With the continuous advancements in machine learning, more and more industry 4.0 applications are expected to emerge. Due to the integration of artificial intelligence into the industry, it has begun to change the characteristics of previously implemented human–machine interactions.

The aim of this study was to propose a prediction model for a real-life production line. Four different ML algorithms, RF, XGB, MLP and SVR, were used to create models with different approaches to the existing data set. It is very important to note that the problem for the ML application begins with the data cleaning and preprocessing procedures. Using real-world data collected from the production lines, we developed four different predictive approaches and compared the results for all possible variations. Among all the proposed methods, RF, an ensemble bagging method, turned out to perform best with our dataset followed by XGB. Although remarkable results have been obtained from other ML algorithms, previously created models gave insufficient results when implementing and testing models with the new validation dataset.

With this new approach, we tried to filter the results from the prediction model and eliminate the effects of high variability in the prediction results. This layer, added to the prediction model, yielded reasonable results and, based on the dataset, prevented approximately 42 percent of production line errors. In future work, we plan to test our proposed method on a long-term dataset for comparison and extend our hybrid approach with additional methods for RUL prediction.

Funding

The authors did not receive support from any organization for the submitted work.

Ethical statement

The authors consciously assure that this material is the authors' own original work, which is not currently being considered for publication elsewhere. This article does not contain any studies with human participants or animals performed by any of the authors.

Informed consent

This article does not contain any studies with human participants or animals performed by any of the authors. The consent is not a requirement for this study.

CRediT authorship contribution statement

Bernar Taşcı: Conceptualization, Data curation, Methodology, Programming, Writing – original draft, Writing – review & editing. **Ammar Omar:** Conceptualization, Data curation, Methodology, Writing – review & editing. **Serkan Ayvaz:** Conceptualization, Methodology, Programming, Supervision, Writing – review & editing.

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.

Data availability

The authors do not have permission to share data.

References

Ayvaz, S., & Alpay, K. (2021). Predictive maintenance system for production lines in manufacturing: A machine learning approach using IoT data in real-time. *Expert Systems with Applications*, *173*, Article 114598.

Cakir, M., Guvenc, M. A., & Mistikoglu, S. (2021). The experimental application of popular machine learning algorithms on predictive maintenance and the design of IIoT based condition monitoring system. *Computers & Industrial Engineering*, *151*, Article 106948.

Canziani, A., Paszke, A., & Culurciello, E. (2017). An analysis of deep neural network models for practical applications. arXiv:1605.07678 [cs].

Carvalho, T. P., Soares, F. A., Vita, R., Francisco, R. d. P., Basto, J. P., & Alcalá, S. G. (2019). A systematic literature review of machine learning methods applied to predictive maintenance. *Computers & Industrial Engineering*, *137*, Article 106024.

Chen, C., Liu, Y., Wang, S., Sun, X., Di Cairano-Gilfedder, C., Titmus, S., et al. (2020). Predictive maintenance using cox proportional hazard deep learning. *Advanced Engineering Informatics*, *44*, Article 101054.

Chen, C., Shi, J., Lu, N., Zhu, Z. H., & Jiang, B. (2022). Data-driven predictive maintenance strategy considering the uncertainty in remaining useful life prediction. *Neurocomputing*, *494*, 79–88.

Coelho, D., Costa, D., Rocha, E. M., Almeida, D., & Santos, J. P. (2022). Predictive maintenance on sensorized stamping presses by time series segmentation, anomaly detection, and classification algorithms. *Procedia Computer Science*, *200*, 1184–1193.

Del Buono, F., Calabrese, F., Baraldi, A., Paganelli, M., & Guerra, F. (2022). Novelty detection with autoencoders for system health monitoring in industrial environments. *Applied Sciences*, *12*(10), 4931.

Francis, F., & Mohan, M. (2019). ARIMA model based real time trend analysis for predictive maintenance. In *2019 3rd international conference on electronics, communication and aerospace technology* (pp. 735–739). Coimbatore, India: IEEE.

Gensler, A., Henze, J., Sick, B., & Raabe, N. (2016). Deep learning for solar power forecasting — An approach using AutoEncoder and LSTM neural networks. In *2016 IEEE international conference on systems, man, and cybernetics* (pp. 002858–002865). Budapest, Hungary: IEEE.

Gupta, V., Mitra, R., Koenig, F., Kumar, M., & Tiwari, M. K. (2023). Predictive maintenance of baggage handling conveyors using IoT. *Computers & Industrial Engineering*, *177*, Article 109033.

Ho, S., Xie, M., & Goh, T. (2002). A comparative study of neural network and Box-Jenkins ARIMA modeling in time series prediction. *Computers & Industrial Engineering*, *42*(2–4), 371–375.

Hyndman, R. J., & Koehler, A. B. (2006). Another look at measures of forecast accuracy. *International Journal of Forecasting*, *22*(4), 679–688.

Jiang, Y., Dai, P., Fang, P., Zhong, R. Y., Zhao, X., & Cao, X. (2022). A2-LSTM for predictive maintenance of industrial equipment based on machine learning. *Computers & Industrial Engineering*, *172*, Article 108560.

Kanawaday, A., & Sane, A. (2017). Machine learning for predictive maintenance of industrial machines using IoT sensor data. In *2017 8th IEEE international conference on software engineering and service science* (pp. 87–90). Beijing, China: IEEE.

Karpathy, A., Johnson, J., & Fei-Fei, L. (2015). Visualizing and understanding recurrent networks. arXiv:1506.02078 [cs].

Lee, J., Davari, H., Singh, J., & Pandhare, V. (2018). Industrial artificial intelligence for industry 4.0-based manufacturing systems. *Manufacturing Letters*, *18*, 20–23.

Lee, G., Kim, J., & Lee, C. (2022). State-of-health estimation of Li-ion batteries in the early phases of qualification tests: An interpretable machine learning approach. *Expert Systems with Applications*, *197*, Article 116817.

Lee, W. J., Wu, H., Yun, H., Kim, H., Jun, M. B., & Sutherland, J. W. (2019). Predictive maintenance of machine tool systems using artificial intelligence techniques applied to machine condition data. *Procedia CIRP*, *80*, 506–511.

Li, Z., Wang, Y., & Wang, K.-S. (2017). Intelligent predictive maintenance for fault diagnosis and prognosis in machine centers: Industry 4.0 scenario. *Advances in Manufacturing*, *5*(4), 377–387.

Mohammadi, M., Al-Fuqaha, A., Sorour, S., & Guizani, M. (2018). Deep learning for IoT big data and streaming analytics: A survey. arXiv:1712.04301 [cs].

Nasir, V., & Sassani, F. (2021). A review on deep learning in machining and tool monitoring: methods, opportunities, and challenges. *International Journal of Advanced Manufacturing Technology*, *115*(9), 2683–2709.

Pacheco, A. L., Flesch, R. C., Flesch, C. A., Iervolino, L. A., & Barros, V. T. (2022). Tool based on artificial neural networks to obtain cooling capacity of hermetic compressors through tests performed in production lines. *Expert Systems with Applications*, *194*, Article 116494.

Rieger, T., Regier, S., Stengel, I., & Clarke, N. (2019). Fast predictive maintenance in industrial Internet of Things (IIoT) with deep learning (DL): A review. *Internet of Things*, 11.

Rivera, D. L., Scholz, M. R., Fritscher, M., Krauss, M., & Schilling, K. (2018). Towards a predictive maintenance system of a hydraulic pump. *IFAC-PapersOnLine*, *51*(11), 447–452.

Rivera-Gómez, H., Gharbi, A., Kenné, J.-P., Ortiz-Zarco, R., & Corona-Armenta, J. R. (2021). Joint production, inspection and maintenance control policies for deteriorating system under quality constraint. *Journal of Manufacturing Systems*, *60*, 585–607.

Shimada, J., & Sakajo, S. (2016). A statistical approach to reduce failure facilities based on predictive maintenance. In *2016 international joint conference on neural networks* (pp. 5156–5160). Vancouver, BC, Canada: IEEE.

Soltanali, H., Khojastehpour, M., Farinha, J. T., & Pais, J. E. D. A. E. (2021). An integrated fuzzy fault tree model with Bayesian network-based maintenance optimization of complex equipment in automotive manufacturing. *Energies*, *14*(22), 7758.

Traini, E., Bruno, G., D'Antonio, G., & Lombardi, F. (2019). Machine learning framework for predictive maintenance in milling. *IFAC-PapersOnLine*, *52*(13), 177–182.

Wang, S., Wan, J., Zhang, D., Li, D., & Zhang, C. (2016). Towards smart factory for industry 4.0: A self-organized multi-agent system with big data based feedback and coordination. *Computer Networks*, *101*, 158–168.

Willmott, C. J., & Matsuura, K. (2005). Advantages of the Mean Absolute Error (MAE) over the root mean square error (RMSE) in assessing average model performance. *Climate Research*, *30*(1), 79–82.

Wright, S. A., & Schultz, A. E. (2018). The rising tide of artificial intelligence and business automation: Developing an ethical framework. *Business Horizons*, *61*(6), 823–832.

Xie, X., Wu, D., Liu, S., & Li, R. (2021). IoT data analytics using deep learning. arXiv:1708.03854 [cs].

Zhang, J., Wang, P., Yan, R., & Gao, R. X. (2018a). Deep learning for improved system remaining life prediction. *Procedia CIRP*, *72*, 1033–1038.

Zhang, J., Wang, P., Yan, R., & Gao, R. X. (2018b). Long short-term memory for machine remaining life prediction. *Journal of Manufacturing Systems*, *48*, 78–86.

Zonta, T., Da Costa, C. A., da Rosa Righi, R., de Lima, M. J., da Trindade, E. S., & Li, G. P. (2020). Predictive maintenance in the Industry 4.0: A systematic literature review. *Computers & Industrial Engineering*, *150*, Article 106889.