The International Journal of Advanced Manufacturing Technology
https://doi.org/10.1007/s00170-024-14097-3

ORIGINAL ARTICLE

Machine learning-based digital twin of a conveyor belt for predictive maintenance

Valerio Pulcini[1] · Gianfranco Modoni[1]

Received: 29 February 2024 / Accepted: 2 July 2024

Abstract
The problem of achieving a good maintenance plan is well-known in the modern industry. One of the most promising approaches is predictive maintenance, which schedules interventions based on predictions made by collecting and analyzing data from the process. However, to the best of the authors' knowledge, this approach is still not widespread and known enough, and particularly, the real-case scenarios of its application appear not exhaustive. To contribute to fill this gap, this work proposes a digital twin (DT), which performs a predictive maintenance approach for a conveyor belt within a real-case scenario with the overall goal of predicting faults during normal belt operations. Specifically, the core of the implemented DT is a model that analyzes the data collected by various sensors distributed along the conveyor belt. In turn, this model exploits a machine learning-based algorithm that predicts the insurgence of faults. The tests of the developed solution, conducted within a real scenario, demonstrated good precision and accuracy in identifying the fault status and also in a time deemed acceptable for the involved stakeholders.

Keywords Predictive maintenance · Digital twin · Machine learning · Conveyor belt

1 Introduction

Maintenance is "any activity – such as tests, measurements, replacements, adjustments, and repairs – intended to retain or restore a functional unit in or to a specified state in which the unit can perform its required functions" [19]. Several approaches of maintenance have been developed and used over the years, mainly [22]:

- Reactive maintenance, in which the intervention is decided and carried on after the occurrence of the fault;
- Preventive maintenance, in which the intervention is scheduled over time;
- Condition-based maintenance, in which the intervention is decided by evaluating the state of degradation of the component.

In recent years, the advent of Industry 4.0 technologies is paving the way for the adoption of predictive strategies and, in particular, the Prognostics and Health Management (PHM) paradigm [59] that makes use of different kinds of collected information with the aim of detecting anomalies from the nominal behavior of an equipment, identifying its degradation states, and predicting its remaining useful life (RUL) [61]. Under these conditions, the outcomes of PHM calculation can be used to support the decisions of a new approach of maintenance called predictive maintenance (PdM), which uses data, models, and knowledge to schedule maintenance operations tailor-made on the process characteristics and present status of the equipment [50, 57]. Compared to other maintenance approaches, PdM can be much more efficient and effective [22] as it can allow the scheduling interventions on optimal time, avoiding operating too early (causing too much machine-stop time) or too late (causing damage to machines and longer machine-stop time) [49, 50]. Nevertheless, this approach is not yet widespread on a large scale, and its adoption is mainly hindered by its realization aspects, which still remain difficult to face [24]. Three of the

Valerio Pulcini and Gianfranco Modoni contributed equally to this work.

✉ Valerio Pulcini
valerio.pulcini@stiima.cnr.it

Gianfranco Modoni
gianfranco.modoni@stiima.cnr.it

1 Institute of Intelligent Industrial Systems and Technologies for Advanced Manufacturing, National Research Council, Via Lembo 38F, Bari 70124, Italy

main aspects, which are addressed in the remainder of this work, are the following:

1. Identification of a proper model to perform the forecasts;
2. Finding a trade-off between accuracy and precision of the model and speed of its computational load;
3. Keeping updated the forecasting model leveraging the data collected within the physical world.

Moreover, it should be noted that, while PdM grants several benefits and it is starting to spread, the number of PdM applications in real-case scenarios is still limited. In addition, PdM approaches are often, if not always, tailor-made to the specific case study and can vary sensibly based on the type of process. In this regard, Tiddens et al. stated in [51] that there is a need to investigate the potential of PdM within further cases of study, and Carvalho et al. [14] suggest exploring PdM applications further in particularly regarding the quality of data and the selection of the proper forecasting algorithm.

With this work, conducted within the European research project AI REGIO ("Regions and Digital Innovation Hubs alliance for AI-driven digital transformation of European Manufacturing SMEs") [4], the goal was to try to fill this absence by providing a solution of PdM for a conveyor belt. This solution is based on a faithful digital twin (DT), i.e., a digital replica of a physical entity with which the DT is constantly synchronized [20].

In particular, the case study is set in a packaging company where a curved conveyor belt is used to automatize the process of realization and filling of plastic bags. The main goal was to limit the insurgence of various types of faults during the curved conveyor belt and, if possible, avoid them because the latter implicates the necessity to stop production to fix them, thus causing high costs. The faults to be identified are of two types: the rupture of the loops connecting the belt with the chain and the slack of the chain itself, which would cause undesired behavior during the process.

The paper is structured as follows: Section 2 focuses on the state of the art of the related works, while Sect. 3 explains the approach used, with its steps. Section 4 describes the architecture of the proposed DT solution. Section 5 deals with the assessment of the proposed solution, explaining the process conducted to verify the validity of it and how it will be applied to the process. Finally, Sect. 6 reports the conclusions and future outcomes.

2 Related works

2.1 Predictive maintenance in industrial scenarios

A significant step to realize a PdM approach is the definition of the forecasting model. Precisely, based on the adopted model, the main PdM approaches proposed in the literature can be classified into three different types, which have been analyzed in a systematic review conducted by Zonta et al. [50].

The first type of approach leans on a data-based view of maintenance by building models to perform predictions on the process components. Two of the main pros of this approach are that it does not require deep knowledge of physics and that it wastes a low computational cost once the model is trained. Instead, two of the main cons are that this data-driven approach is unsuitable for limited historical data and requires the model's retraining when the operational conditions change. Some examples of the application of this approach, which is the one used in the herein presented work, as it will be seen further on, are provided in [53] by Wu et al.. In particular, the latter reported a comparative study on different algorithms based on ML for predicting tool wear. Moreover, Roosefert et al. developed an ML-based model to apply the PdM to a molding machine [42], which is proved to be valuable, granting a great increase in the average time between failures. Another interesting study using this approach is proposed in [18] by Deutsch et al., who leveraged deep learning to predict the remaining useful life of components, specifically rotating ones. The method leverages vibrational data and was validated via two case studies; it showed interesting capabilities for predicting the remaining useful life of rotating components.

A second approach revolves around the development of physical models, using mathematical modeling and statistical analysis to measure the condition of the process and failure occurrence. Compared to the data-driven approach, this second approach needs a deeper understanding of the involved phenomenon and it has a high computational cost, but, on the other hand, it requires less calibration data and can be used to estimate unmeasured variables [9, 26]. An example of this second approach is provided in [25] by Kahiomba et al., who developed a model for PdM analyzing vibrational data.

Finally, the third approach is a hybrid one which combines knowledge-based with data-driven techniques. The prediction of the systems based on this approach can be more accurate. However, the development of these systems is more complex as they comprise different kinds of models which must be properly integrated. One example of the hybrid approach is reported in [47] by Steenwinckel et al., who tried to create a synergy between expert knowledge and ML. Similarly, Luo et al. [26] proposed a hybrid DT-based approach for the PdM of a CNC machine tool, building the DT model with both data-based algorithms and physical-based concepts (degradation model, electric and thermodynamic considerations, etc.).

2.2 PdM solutions applied to conveyor belts

Leveraging the three approaches reported in the previous subsection, different solutions have been proposed in the literature to apply maintenance for the conveyor belt equipment, which is this study's target. A solution is proposed in [25] by Kahiomba et al., who developed a PdM schedule for the conveyor belt motor, exploiting an intelligent vibration sensor and a PLC. The aim is to limit the vibration before it reaches the dangerous zone through a maintenance program accounting not only the current wear conditions of machines but also their future predicted status. Nevertheless, this approach is still based on empirical knowledge and on the analysis of vibration waves. Shifting to data-based methods, an interesting approach is proposed by Görür et al. [37], which presents a PdM approach based on ML. It focuses on safety, including a closed-loop system that runs PdM analysis in real-time and reports a safe operating mode. Each time an abnormal behavior is predicted, the entity of the deviation from a nominal behavior is calculated. The approach proposed by Görür et al. gave interesting results, and it can be considered a valid reference for the present work although it is built on a model that uses as input a polynomial approximation of data and not raw data collected from sensors as the model developed in the herein presented solution. Indeed, the latter has to exploit the vibration data collected by sensors placed on the machine since the main problems regarding conveyor belts typically involve vibration phenomena that cause malfunctioning in the production [39]. In particular, the factors inducing vibration are usually identified in the deformation of the belt and its velocity, assembly issues, and, finally, the problems occurring during normal functioning, mainly the slack of the drive chain and the breakage of the connections between the chain and belt. In this regard, many approaches have been developed to solve the vibration issue. Specifically, Samuel et al. [44] illustrate a vibration-based damage detection technique, identifying Prognostic and Health Management Systems by providing a condition-based maintenance schedule that ensures safety and cost efficiency.

All these solutions so far proposed in the literature to apply PdM of a conveyor belt were proven to give a good understanding of the phenomenon and an acceptable amount of fault prediction, but they tend to rely too much on empirical knowledge since they require a detailed study of vibration phenomena involved in the functioning of the conveyor belt and the skill to create and validate mathematical models to enable the good functioning of the approach [60]. On the other hand, the herein presented approach aims to eliminate or reduce this need, proposing a method in which some knowledge about the process is still required but far less specialized. Indeed, once the model is set, there is no need to know the behavior of the vibrations in detail since the model itself can translate the collected data into usable information.

2.3 DT solutions for PdM

The various physics and data-based models used to apply the three different PdM approaches can be exploited in their full potential if they are synchronized with the changes occurring within the physical factory. In this regard, it can be used the emerging technology of DT, which allows the synchronization of the digital replica with its physical counterpart, thus keeping the first updated through the information about the evolutions occurring within the physical world [20]. Therefore, the use of DT can bring the PdM solutions a significant added value [52], by mainly showing how DT can contribute to enlarge the knowledge of the status of the process, which is a key part in developing a PdM approach. Several solutions have been implemented using DT to improve the approach to manufacturing, reporting the advantage brought in different areas by DT providing real-time information to the digital model [32, 54]. In particular, the combined use of the real asset and its digital counterpart enriches the information provided for the maintenance plan, heightening the level of prediction and the approach overall.

Along with many benefits, DT presents important challenges to be reckoned with. The studies above mentioned [32, 52] report challenges mainly related to the collection of the data [12], the development of the model [58], and the computational burden [33]. Moreover, [52] highlighted the correct issue of a limited number of real applications of DT for PdM available in the literature, and they also reported the lack of DT framework that can support the approach, especially regarding their adaptation to the specific case study. Similarly, Tao Shen et al. performed a review on the application of DT in additive manufacturing, highlighting the lack of DT models for real-time prediction and AI implementation [46].

Based on the results of this study conducted on the related works, no application is deemed exhaustive and applicable to our case of study. For this reason, in the context of this work, it was considered the best course of action to develop a new solution to deal with the presented issue, leveraging a DT solution in order to gain useful insights from the monitoring of the conveyor belt through a tailor-made ML model that provides the best possible plan of maintenance for the conveyor belt in analysis.

3 Proposed solution for the case study

According to the typical approach for PHM [11, 36], the proposed solution to apply PdM for the analyzed conveyor belt implements an approach based on three consecutive steps:

1. *Observation*, which consists in acquiring and collecting the necessary data via a well-designed network of sensors and dedicated instrumentation. Later on, the collected data must pre-processed with proper tools in order to clean, resample, and normalize it, thus making them more suitable for the next steps of the approach [36].
2. *Data analysis for the diagnostics*, which allows to perform both a degradation level assessment and fault detection through the processing of the collected data [11] and aims to understand when the system or component will be in a non-acceptable state. For this reason, this step usually involves a model for monitoring the degradation process and understanding how it evolves over time, by leveraging data-based, physic-based, or hybrid-based approaches [36].
3. *Health management through ad-hoc actions*, which consists in using the knowledge gained through the first two steps to make decisions to apply interventions to the system. A particular focus in this phase is devoted to analyzing how to show the results of step 2 to the stakeholders by considering human–machine aspects [11].

The description of how the three steps of the approach have been implemented in the pilot case is reported in detail in the following subsections.

3.1 Step 1: observation of the conveyor through sensors

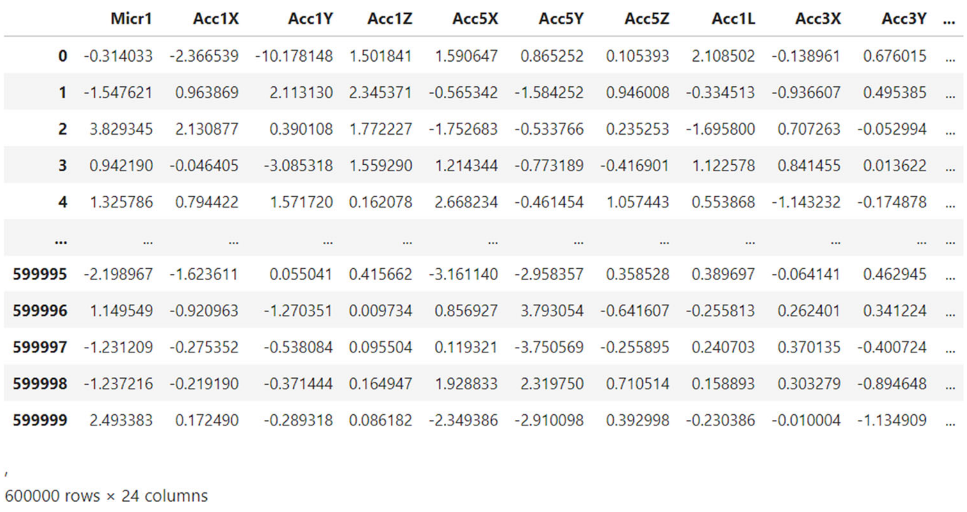
The belt was over-instrumented with various sensors placed on its frame and, in particular, one piezoelectric microphone, several mono-axial and tri-axial accelerometers (X-axis along the direction of motion, Y-axis orthogonal to the fastening surface, and Z-axis transversal to the direction of motion), a photocell, and three current probes. The sensors were installed in 5 fixed points identified along the curve. During the test's sessions, the following conditions of the belt were reproduced:

- Nominal (i.e., no alteration to work conditions);
- Slack in the drive chain (of 1, 2, or 5 mm);
- Removal of some connecting loops (from 1 to 12)
- Removal of a part of some connecting loops (from 1 to 12)

where slack and removal of loops were artificially reproduced.

Each specific condition was monitored in a full load condition and maintained for 10 min, except for the nominal condition, which was maintained for more than 200 min. In addition, in order to monitor each occurring event, the data was acquired at the maximum rate supported by the adopted technology, comprising sensors and acquisition software, i.e., 51,200 Hz. This rate contributed to produce several telemetry files, each corresponding to a 1-min length measurement and a size equal to 588 megabytes. Each telemetry file was saved in the *TDMS format*, a national instrument native format specifically designed to contain the data of the sensor measurements organized in channels [7]. In particular, a TDMS file contains data of the sensor measurements (samples) organized in channels. For the analyzed case study, the telemetry file comprised 24 channels, each containing numeric values. Each of the first 23 channels, some of which are represented in Fig. 1, corresponded to one different measurement performed by a sensor. The last channel is a *virtual* channel, called *Error*, which was added to the set of already acquired samples. Specifically, the value of this channel is 0 if the telemetry file corresponded to a nominal case and 1 if it corresponded to a fault case.

According to some studies on conveyor belts [39] available in the literature, the typical frequency of transverse vibrations interesting to the belts is a maximum of 200 Hz. Thus, according to the Nyquist-Shannon Theorem [45], the



	Micr1	Acc1X	Acc1Y	Acc1Z	Acc5X	Acc5Y	Acc5Z	Acc1L	Acc3X	Acc3Y	...
0	-0.314033	-2.366539	-10.178148	1.501841	1.590647	0.865252	0.105393	2.108502	-0.138961	0.676015	...
1	-1.547621	0.963869	2.113130	2.345371	-0.565342	-1.584252	0.946008	-0.334513	-0.936607	0.495385	...
2	3.829345	2.130877	0.390108	1.772227	-1.752683	-0.533766	0.235253	-1.695800	0.707263	-0.052994	...
3	0.942190	-0.046405	-3.085318	1.559290	1.214344	-0.773189	-0.416901	1.122578	0.841455	0.013622	...
4	1.325786	0.794422	1.571720	0.162078	2.668234	-0.461454	1.057443	0.553868	-1.143232	-0.174878	...
...	...	...	...	...	...	...	...	...	...	...	...
599995	-2.198967	-1.623611	0.055041	0.415662	-3.161140	-2.958357	0.358528	0.389697	-0.064141	0.462945	...
599996	1.149549	-0.920963	-1.270351	0.009734	0.856927	3.793054	-0.641607	-0.255813	0.262401	0.341224	...
599997	-1.231209	-0.275352	-0.538084	0.095504	0.119321	-3.750569	-0.255895	0.240703	0.370135	-0.400724	...
599998	-1.237216	-0.219190	-0.371444	0.164947	1.928833	2.319750	0.710514	0.158893	0.303279	-0.894648	...
599999	2.493383	0.172490	-0.289318	0.086182	-2.349386	-2.910098	0.392998	-0.230386	-0.010004	-1.134909	...

,

600000 rows × 24 columns

Fig. 1 Example of the adopted data set with the visualization of a set of channels (columns) and related samples

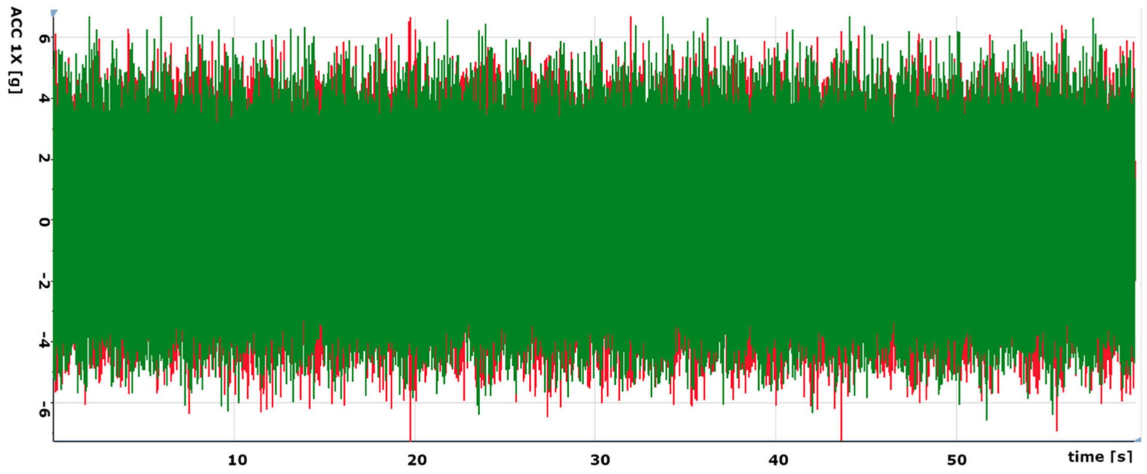


Fig. 2 Superposition of Acc1X acquired in the condition of nominal (green) and one loop removed (red)

sampling rate to use must be at least 400 Hz [27]. Under these conditions, the much higher sampling rate adopted for the telemetry in this case study (51,200 Hz) allows accurate monitoring, granting an excellent representation of the phenomenon. On the other hand, it brings a series of complications, such as a high computational cost to process, transfer, store, and visualize telemetry files. In this regard, to visualize these files, it was decided to use the proprietary tool DiAdem [5], which allows high precision in analysis and manipulation. At first, this tool was used for visual comparison of superposed signals acquired in different conditions (i.e., nominal versus error conditions) in order to identify visible differences. In this section, for the sake of brevity, it is chosen to limit to report the analysis of the acceleration signal corresponding to one of the used sensors along the X-axis. Specifically, as shown in Fig. 2, the first minute of measurement for a fault case and a nominal case is reported. The red signal represents the acceleration Acc1X measured in the fault case (12 loops missing), while the green signal is the same acceleration measured in the nominal case. In this comparison, some slight differences between the nominal case and the loop removal case can be visualized, seeing in particular that the peaks appear different and spread differently, and the waveform seems visually not the same, but it should be noted that the mentioned differences are not evident. Also, the same visual comparison between nominal and fault cases performed for other signals did not show evident differences. Since this performed comparison proved to be non-significant, to better underline the signal pattern for each belt condition, it was decided to focus the analysis on the zoomed images of the superposed plots as the example reported in Fig. 3. This zoomed plot highlighted the difference between the two signals, and thus, it can be considered a valid preliminary step in understanding the failure impact on the process. In particular, it is interesting to notice in Fig. 3 the visible difference in the waveform of the nominal (green)

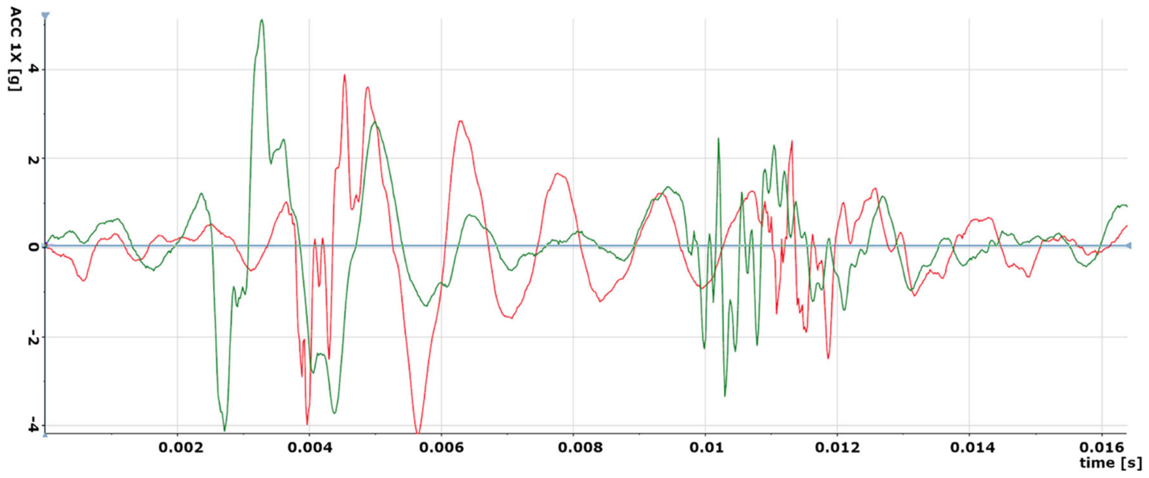


Fig. 3 Zoomed superposition of Acc1X, nominal (green) and one loop removed (red)

and the failure test (red), and some notable differences in some selected moments, for instance, in the range of values from 0.002 to 0.006, or from 0.01 to 0.012 of the x-axis, both reporting a clear change in pattern.

However, despite the positive results achieved with the above evaluations, it should noted that these evaluations are limited to the analysis of a portion of a signal acquired under two different conditions, while the analysis should be applied to the totality of the data acquired from all the sensors during the various conducted tests, which is a much more difficult activity to bear. In this regard, a new strategy has been investigated, which considers the issue of the large size of the telemetry files by reducing it before analyzing and passing this data to the forecasting model. Specifically, it was decided to apply a resampling to reduce each file's dimension while maintaining the relevant information conveyed by the original data. The resampling was performed through the linear interpolation function provided by the already-mentioned DiAdem tool, and this function allowed the reconstruction of each signal exploiting a set of selected points and estimating the adjoining ones [1]. In particular, a resampling with a frequency of 10,000 Hz was applied from the original signal frequency (51,200 Hz), still granting compliance to the Shannon-Nyquist theorem, considering the maximum sample rate suggested in the literature is 400 Hz. This operation reduced each file dimension from 588 to 151 MB. Afterward, in order to evaluate the data loss after the resampling, a set of resampled signals was compared with the corresponding original. As an example, a superposition of a portion of the original signal Acc1X of the nominal test and the corresponding resampled is reported in Fig. 4. The green curve is the resampled plot, and the red is the original signal plot. The pattern shown is similar in both the signals, and the peaks were (min=−7.057;max=8.184) and (min=−5.895;max=6.988), respectively, for the original and resampled signal, with a maximum loss of 1.196 in amplitude. The data loss was considered acceptable, and the resampled signal was still significant. Similar results were also obtained for other resampled signals.

Since the files' dimensions were still considered big after the first resampling, a stricter resampling was tried, applying a sampling rate of 5000 Hz. It should be noted that this rate still adheres to the Nyquist-Shannon sampling theorem, being much higher than the frequency suggested in the literature (i.e., 400 Hz). The result of this latest resampling is that the data size was reduced to 99.5 MB, while the superposition of the resampled and the original signals (considering the same signal portion reported in Fig. 4) is shown in Fig. 5. In this figure, the green curve is the resampled plot, while the red is the original signal plot. Compared to the previous resampling, the pattern shown between the two signals is much less similar, losing the wave's peaks and also changing its structure. Thus, it is clear that the second resampling is much more impacting on the meaningfulness of the signal, and for this reason, it was decided to adopt for the subsequent data preprocessing the resampling rate of 10,000 Hz evaluated in the first conducted experiment.

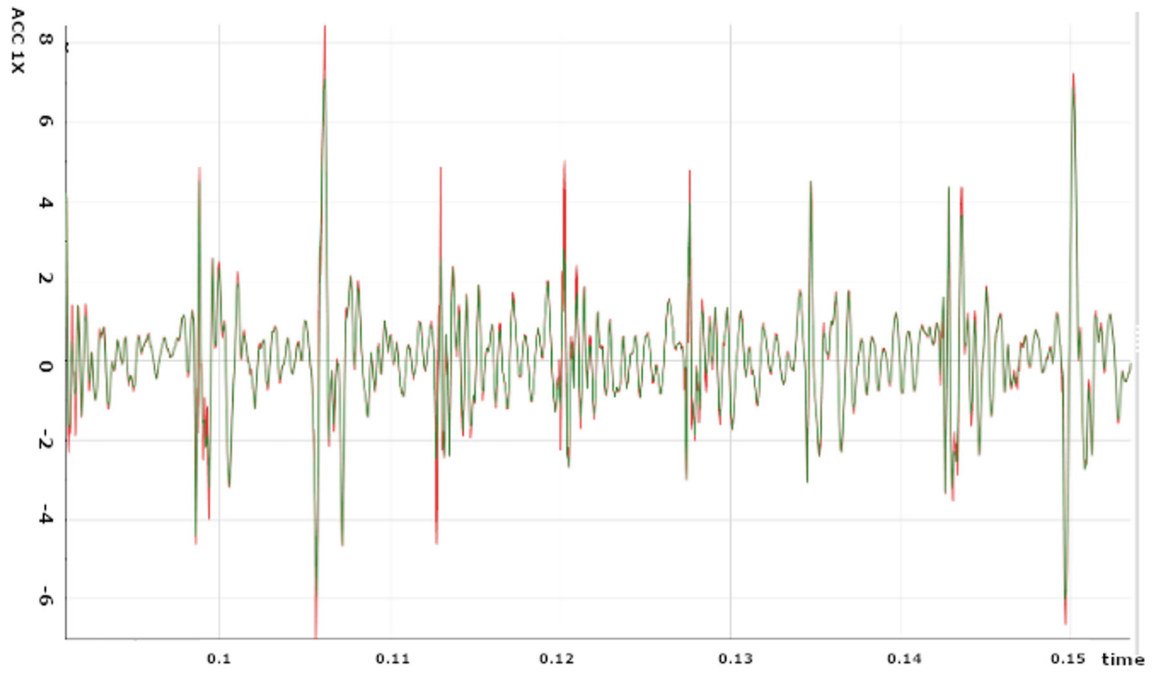


Fig. 4 Superposition of Acc1X of nominal test, original (red) and resampled 10,000 Hz (green)

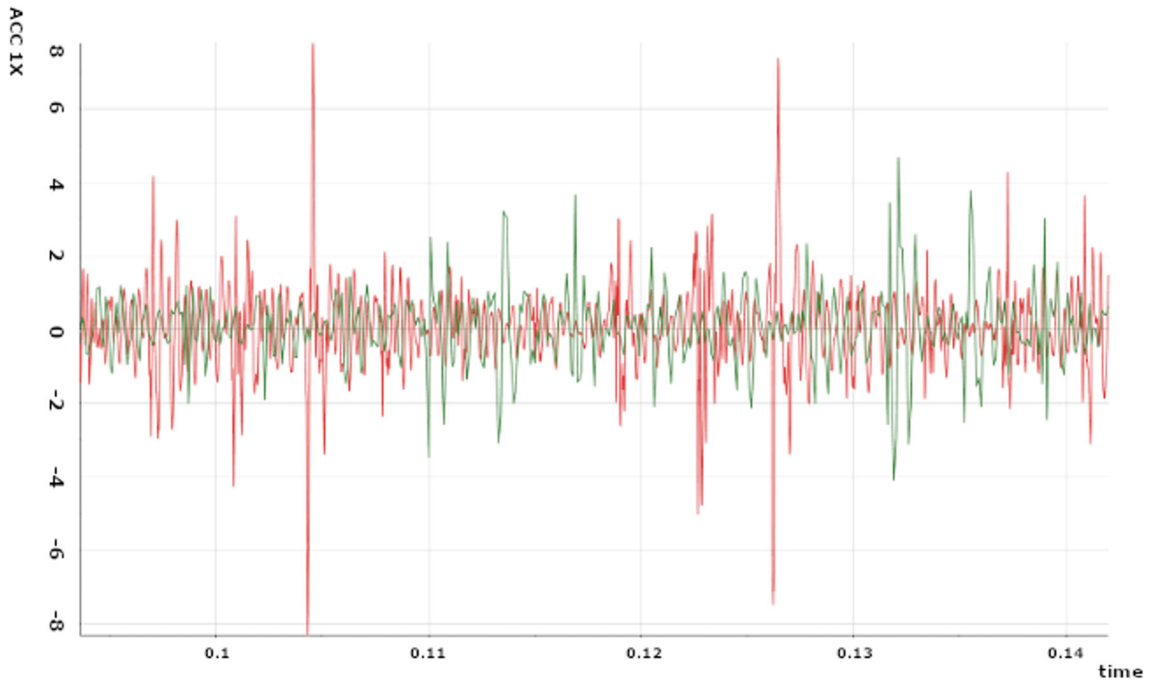


Fig. 5 Superposition of the original Acc1X channel corresponding to a nominal test (red) and its resampling at 5000 Hz (green)

3.2 Step 2: analysis of the conveyor's telemetry data leveraging the ML model

This step concerns the identification of faults during the normal functioning of a conveyor belt, exploiting the data collected through Step 1. In this regard, an ML model was developed to monitor and predict the status of the conveyor belt. The following subsections illustrate the activities conducted to realize this model.

3.2.1 Selection of the ML method to evaluate

A first comparison of various methods was conducted before proceeding with the evaluation. According to literature [29, 40], various methods can be applied to fault identifications, each with different characteristics. The most common ones have been evaluated and compared to understand which is best suited to the problem under analysis. They are reported in the following, accompanied by a brief description:

- **Binary classification**, which consists in a supervised learning approach that allows the classification of the given data in one or two predefined classes [23, 56].
- **Multi-class classification**, which is similar to the binary approach but involves more classes for a more precise differentiation of the cases [10].
- **Regression**, which uses determined independent variables, called predictors, to foresee some decided dependent variables [17].
- **Anomaly detection**, an anomaly system that defines a *normal status* in which the system should be during normal functioning. Then, it can identify and notify when the system deviates from this status [38, 43].

Every method has pros and cons, which can be more suitable in some cases and less in others. For this reason, various approaches based on one or more of the above-mentioned methods were evaluated in this study. In particular, Random Forest (binary classification) [41], logistic regression (regression) [21], linear SVM (binary classification) [48], and decision tree (regression) [55] were taken into account, evaluated, and compared in order to implement a binary classification that distinguishes between nominal and fault cases. It is noteworthy to mention that other methods, specifically based on Recurrent Neural Network (RNN), proved to be a valid option for cases similar to the one analyzed [15], but they have not been yet evaluated mainly for the required high computational cost. However, it is intended to include them when investigating more models as part of the future steps (Sect. 6).

3.2.2 Evaluation and comparison of different ML methods

The evaluation stage started with the training of each ML model, which was conducted using as input of the model data sets acquired in nominal conditions and data sets acquired in a condition of fault. Specifically, one nominal case and six fault cases were used (three files with slack in the chain, and three files with loops removed).

For each algorithm to be evaluated, it was used a *test size* of 0.2, meaning it was taken 80% of the available data to train the algorithm and 20% of data to test it. This ratio was chosen based on the evidence in literature [16] to guarantee an adequate amount of data for training, thus performing an accurate training of the model; the data was divided randomly taking 20% of data from each channel for each data set (i.e., for every considered case was taken 20% of each channel randomly). The evaluation of the four different methods was conducted by comparing their *Receiver Operating Characteristic (ROC)* curve and their *Area Under the Curve (AUC)*. The latter are well-established methods to evaluate the quality of the ML algorithms and, in particular, their capacity to correctly identify the classes [13, 28]. The ROC curve has been designed in Fig. 6 for the four different algorithms using the same overall dataset for training and testing as input.

It should be noted that the perfect score is AUC = 1, while a random approach (which randomly assigns a label of 0 or 1 to the sample) has an AUC = 0.5. The Random Forest algorithm (RanFor) has an AUC of 0.87, by giving the best performance among all the evaluated models. In addition, to evaluate more accurately the four developed algorithms, the following indexes were calculated for each algorithm [8]:

- **Precision score**, i.e., the percentage of occurrences correctly predicted, which represents the model's ability to not label a negative sample as positive. It is calculated as the fraction between the true positives (t_p) and the total given by true positives plus false positives ($t_p + f_p$) $\frac{t_p}{t_p+f_p}$;

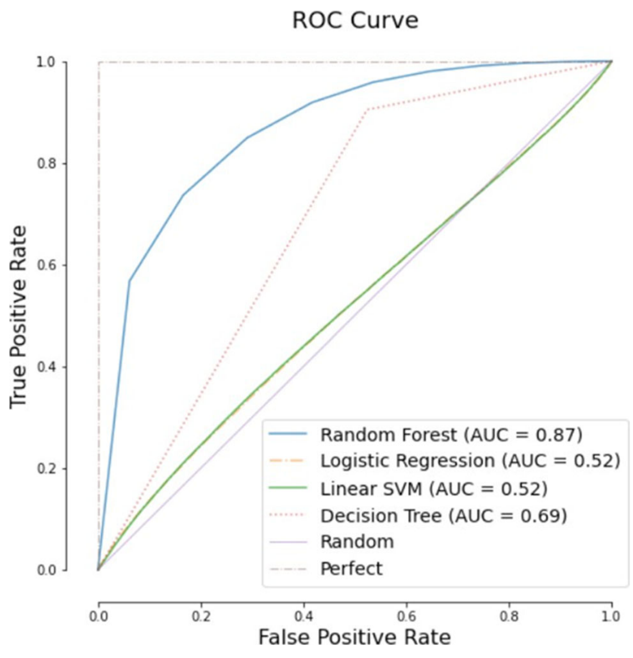


Fig. 6 ROC curve for four different developed algorithms

Table 1 Algorithms indexes

Indexes	RanFor	LogReg	SVM	DecTree
Precision score	0.88	0.74	0.76	0.85
Recall score	0.89	0.86	0.63	0.84
F1-score	0.88	0.79	0.68	0.84

- **Recall score**, i.e., the percentage of positive occurrences correctly predicted over the total of positive occurrences, which represents the model's ability to find all the positive samples. It is calculated as the fraction between the true positives (t_p) and the sum of true positives plus false negatives ($t_p + f_n$) $\frac{t_p}{t_p+f_n}$;
- **F1-score**, a weighted mean of precision and recall that gives an overall algorithm evaluation.

The calculated indexes for the four algorithms, reported in Table 1, highlighted the better performance of the RanFor, thus confirming the evaluation leveraging the AUC index.

Based on the results, the Random Forest algorithm was selected to continue this study.

3.2.3 The code implementation

The implemented code has been developed by using the Python language and the native libraries of the Jupyter Lab platform [6], and it was made available by the authors on Zenodo's platform [3]. Some examples representing the developed code are shown below. In particular, in Listing 1 below, it is reported the code to read the telemetry data sets. In this regard, a Pandas Data frame was created by exploiting the Python library Pandas [31], and it was then filled with a set of imported files.

In Listing 2, the import of the libraries and the related functions needed to develop the model is shown.

Moreover, to randomly divide the input data between input for training and for testing based on the established *test_size*, the Python function *train_test_split* was exploited, as shown in Listing 3. In particular, a ratio of 80% for training and the remaining 20% samples for testing was adopted, thus obtaining the two couples of values (X_train, Y_train) and (X_test, Y_test), where the *X* is the value provided by the sensors, while the *Y* is the predicted value and it represents the presence or absence of the error.

In Listing 4, it is shown the snippet code which performs the training of the model and the calculation of the accuracy, showing the elapsed time.

Moreover, to store the trained model in a specific archive (locally or on a cloud), it is exploited the *dump* function, as shown in Listing 5. The stored model can then be used in subsequent processing without the need to train it again, thus reducing processing time, shown in Listing 6.

Listing 1 Read telemetry data sets

```
      # Errror Tdms files to be loaded
file_paths = [
    "data_Test41_0009_catena_1mm_sampled.tdms",
    "data_Test41_0003_cat1_sampled.tdms",
    "data_Test43_0002_cat5_sampled.tdms",
    "data_Test42_0013_cat2_sampled.tdms",
    "data_Test42_0004_catena_2mm_sampled.tdms",
    "data_Test43_0010_catena_5mm_sampled.tdms",
]

# Create empty list for the Dataframe
df_err = []

# Run every tdsm file and prepare corresponding database
for file_path in file_paths:
    full_path = '/Users/AIRegio/Sampled_Data/Chain/' + file_path
    tdms_file = TdmsFile(full_path)
    df = PrepareTrainingSet_Error(tdms_file)
    df_err.append(df)

df_error = pd.concat(df_err, ignore_index=True)
df_error
```

Listing 2 Libraries and functions import

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay
import matplotlib.pyplot as plt
from sklearn.decomposition import PCA
from scikitplot.metrics import plot_roc, plot_precision_recall,
plot_cumulative_gain, plot_lift_curve
from numpy import argmax
import numpy as np
from sklearn.metrics import precision_score, recall_score, f1_score,
accuracy_score, roc_curve, auc, roc_auc_score
from sklearn.metrics import classification_report
import pandas as pd
from nptdms import TdmsFile
import time
import joblib
```

Listing 3 Divide data for training and testing

```
from sklearn.model_selection import train_test_split
X_train, X_test, Y_train, Y_test = train_test_split(X,Y, test_size=0.2,
random_state =1984)
```

It was created a specific function for all the evaluation metrics (explained previously) to be loaded and used in the code, shown in Listing 7.

3.3 Step 3: action

The outcomes of the implemented ML model can then be exploited to inform the stakeholders about the status of the belt by providing predictions of the eventual fault cases, thus allowing to make actual interventions in the system (e.g., emergency stops) and to enable the decision-making process about its maintenance. Specifically, in this phase of the research project and as asked by the company itself, it was thought to leave the given information to a very core state to let the related stakeholders analyze it and produce an operational plan based on the knowledge of the process itself.

This information differs in form and content according to the specific stakeholder's needs. Specifically, for this scenario, the solution was conceived for two main types of

Listing 4 Model training

```
#Training the model

start_time = time.time()

model = RandomForestClassifier(criterion = 'entropy', min_samples_leaf = 3,
min_samples_split = 5, max_depth = 60 )
model.fit(X_train, Y_train)


train_accuracy = model.score(X_train, Y_train)
print(f"Training Accuracy: {train_accuracy}")

test_accuracy = model.score(X_test, Y_test)
print(f"Test Accuracy: {test_accuracy}")

end_time = time.time()
elapsed_time = end_time - start_time
print(f"The model took {elapsed_time} seconds to train.")
```

Listing 5 Store the trained model

```
#Saving model locally
joblib.dump(model, 'Trained_Model.pkl', compress='zlib')
```

Listing 6 Load the saved model

```
#loading model from local
model = joblib.load('/Users/AIRegio/Trained_Model.pkl')
```

users (system operators and process engineers), each with different duties and needs. In particular, the system operator is responsible for performing small adjustments to the system, monitoring the regular functioning of the conveyor belt, and reporting faults and problems of the system. To accomplish these tasks, the system operator can access, through a proper dashboard, a simplified version of all the ML model outputs, particularly the count of the values predicted as nominal or faults. Indeed, the proposed solution only provides raw information, which is analyzed by the system operator, who identifies potential problems based on his experience. Instead, in the future, an evolution of the solution is foreseen to set a threshold for the number of predicted faults in order to discern between the status of *regular functioning* and *action needed*.

In the workflow supported by the current solution, if the operator identifies potential problems, he communicates them to the *process engineer*, who can decide the appropriate corrective action to apply to the belt. In fact, the process engineer is responsible for ensuring the system is functioning well by scheduling corrective interventions and deciding on a maintenance plan. In addition, the process engineer controls the quality of the ML model by analyzing its performance. In this regard, the process engineer can visualize various information such as the prediction count (as for the operator), the main indexes, the principal component analysis, and the confusion matrix calculated as it is later reported in Sect. 5, and this information is essential for deciding about possible improvements to the ML model.

4 The DT-based solution

A DT-based solution is well suited to support the three steps of the approach described in Sect. 3 and, in particular, to integrate these steps into an overall approach. Under these conditions, the authors conceived and realized a DT of the conveyor belt that provides the telemetry data acquired on the real conveyor belt as input to the ML model. In turn, the ML model uses telemetry data to monitor and predict the status of the conveyor belt, while the model outputs are exploited to make decisions to apply some corrective actions on the belt. This implemented DT, whose code is available on Zenodo's platform [3], represents one of the assets included in the assets catalog of the AI REGIO project [4], and it is also included in the AI4EU portal [2].

According to the typical architecture for DT [35], the proposed solution comprises the following three main components (Fig. 7):

Listing 7 Evaluation metrics for the trained model

```python
#function to report all the evaluation metrics for the trained model
def evaluate_model_first(model, X_train, Y_train, X_Test, Y_Test):

    # Test the model
    Y_pred = model.predict(X_Test)
    print('Precision score: %s' % precision_score(Y_Test, Y_pred))
    print('Recall score: %s' % recall_score(Y_Test, Y_pred))
    print('F1-score: %s' % f1_score(Y_Test, Y_pred))
    print('AccuracY score: %s' % accuracy_score(Y_Test, Y_pred))

    # Confusion Matrix
    print('Confusion Matrix training set')
    confusion_matrix = ConfusionMatrixDisplay.from_estimator(model, X_train,
    Y_train, display_labels=['nominal', 'error'], cmap='Blues')

    print('Confusion Matrix test set')
    confusion_matrix = ConfusionMatrixDisplay.from_estimator(model, X_Test,
    Y_Test, display_labels=['nominal', 'error'], cmap='Blues')

    # Build and Plot PCA
    pca = PCA(n_components=2)
    pca.fit(np.array(X_train))
    X_pca = pca.transform(np.array(X_train))
    plt.scatter(X_pca[:, 0], X_pca[:, 1], c=y_train, cmap=plt.cm.prism,
    edgecolor='k', alpha=0.7)
    plt.show()

    Y_score = model.predict_proba(X_Test)
    fpr0, tpr0, thresholds = roc_curve(Y_Test, Y_score[:, 1])
    roc_auc0 = auc(fpr0, tpr0)

    # Plot metrics
    plot_roc(Y_Test, Y_score)
    plt.show()

    plot_precision_recall(Y_Test, Y_score)
    plt.show()

    plot_cumulative_gain(Y_Test, Y_score)
    plt.show()

    plot_lift_curve(Y_Test, Y_score)
    plt.show()

    # Print a classification report
    print(classification_report(Y_Test, Y_pred))
```

1. The *ML model*, which uses the Random Forest algorithm selected within Step 2;
2. The *telemetry* acquired through the sensors distributed across the belt;
3. The *database* used to persist the information and where the trained model is stored. In the first implementation of the proposed solution, a file archive was used to store the telemetry files.

Figure 7 also reports the main interactions among the components under the form of arrows described below. First, the *Arrow 1* represents the telemetry data acquired from the belt and given as input to the implemented model to train it, while *Arrow 2* represents the trained model, which is stored and persisted within a proper a file archive. The *Arrow 3*, *Arrow 4*, and *Arrow 5* embody the input of the ML model.

Specifically, *Arrow 3* represents the new data provided by the telemetry to be processed in the form of samples, each containing 24 channels as described in Sect. 3.1, while *Arrow 4* represents the loading of the trained model from the database and *Arrow 5* represents the algorithm selected through Step 2 of the approach and then given as input to

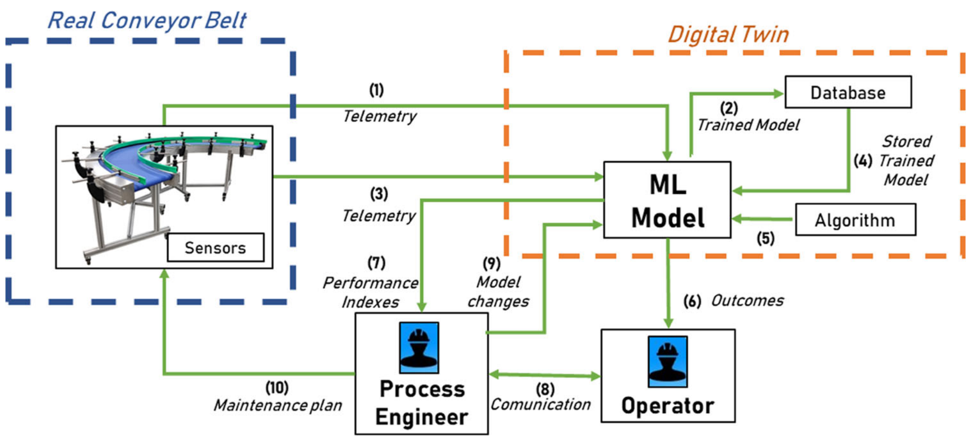


Fig. 7 The proposed DT solution

the ML model. Regarding this algorithm, for the benefit of the readers, the snippet code implemented to perform the new prediction and calculate the elapsed time is reported in Listing 8, while in Fig. 8, it is reported the results of the calculation. In particular, the reported results consist in a count of the predicted values, reporting the number of values predicted as *0* (nominal) and the number of values predicted as *1* (fault cases). These results, which are shown to the operators, represent one of the two outputs of the ML model and are reported in Fig. 7 through the *Arrow 6*, while the other output of the ML model (*Arrow 7*) represents the performance indexes shown to the process engineer. In addition, the bidirectional *Arrow 8* represents the communication exchanged between the operator and the process engineer. Finally, the *Arrow 9* represents the changes applied by the process engineer to the model, while the *Arrow 10* is an update of the maintenance plan to be applied to the belt.

5 Assessment of the proposed solution

This section reports the test session results conducted to assess the proposed solution. In this evaluation, it is taken into account the results of the predictions (number of faults and nominal cases predicted), but also the indexes (precision score, recall score, f1 score), and the confusion matrix to clearly understand the accuracy of the prediction.

In the first executed test, eight nominal files and six error files were used to train and test the model (*test size* = 0.2). The results are shown in Fig. 9, which reports the model evaluation through a confusion matrix. Among the total labels (1,44e6), the matrix reports that 1,40e6 were correctly predicted as nominal, and 2,23e5 were correctly predicted as faults. Regarding incorrect predictions, 8,00e4 were predicted as bad while being nominal, and 1,37e5 were predicted as nominal while being faults. This gives a total accuracy score of 0.88, a recall score of 0.62, and a precision score of 0.74, showing an acceptable response from the model for the performed prediction.

In the second test, it proceeded to give as input to the previously trained model a single file corresponding to an error case. The results of this test, including a confusion matrix, are reported in Fig. 10. They show an accuracy score of 0.83, a recall score of 0.83, and a precision score of 1, showing a even better response than the first test.

The third test aims to evaluate the impact of resampling on the prediction. In this regard, the ML model input is not a resampled file corresponding to a nominal case prediction.

Listing 8 Perform new prediction and report valuer count

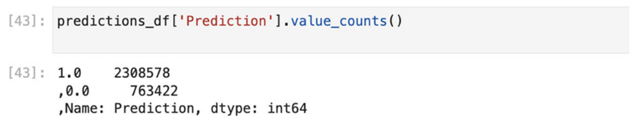
```

predictions_df['Prediction'].value_counts()
start_time = time.time()
new_prediction = model.predict (X_Test_err2) #making predictions

end_time = time.time()
elapsed_time = end_time - start_time

print(f"The model took {elapsed_time} seconds to make predictions.")

# Create a new DataFrame to hold the predictions
predictions_df = pd.DataFrame({'Prediction': new_prediction})
print(predictions_df)
```



```
[43]: predictions_df['Prediction'].value_counts()

[43]: 1.0    2308578
      ,0.0     763422
      ,Name: Prediction, dtype: int64
```

Fig. 8 Code and results for the prediction of a new file

The results, reported in Fig. 11, show a 98% accuracy, which is very high, and thus, it demonstrates an excellent prediction of nominal cases. This proves that a certain loss is caused by the resampling, but it is a necessary trade-off due to the calculation times required for not resampled data sets.

Despite the results obtained in all three tests were improvable, the performed analysis of the evaluation demonstrated that these results were consistent with a high number of true predictions and a low rate of incorrect ones. Under these conditions, it is possible to understand the presence of nominal and fault cases, and the operators can act accordingly.

Concerning the time requested for the prediction, the model took 56 s to predict 1 min of acquisition from the sensors (resampled to 10,000 Hz), while the not-resampled case took 241 s to predict a minute of acquisition. This gives an acceptable response time, allowing monitoring in-line of the belt during its normal use by predicting every minute of a resampled acquisition or, alternatively, 1 min every 4 min of a not-resampled one.

```
[27]: evaluate_model(model, X_train, y_train, X_test, y_test)
      Precision score: 0.7368466245917178

      Recall score: 0.618768962820392

      F1-score: 0.6726653571853891

      Accuracy score (test set): 0.8806821366361334
```

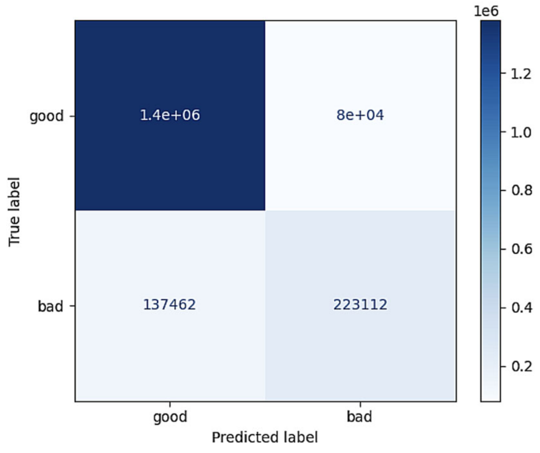


Fig. 9 Evaluation of the results in the second test (the inputs are eight nominal files and six error files)

```
Precision score: 1.0

Recall score: 0.8270345052083333

F1-score: 0.9053299243672775

Accuracy score: 0.8270345052083333

Matrice di confusione test set
```

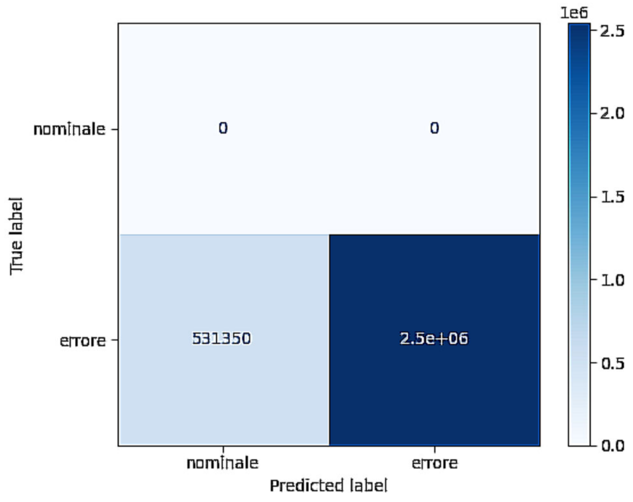


Fig. 10 Evaluation of the results in the second test (the input is a single file corresponding to an error case)

6 Conclusion and future outcomes

The contribution of this study is a DT solution-based ML model that can predict the occurrence of faults in a curved conveyor belt. The results obtained in the study demonstrated

```
Accuracy score: 0.9812587890625

Matrice di confusione test set
```

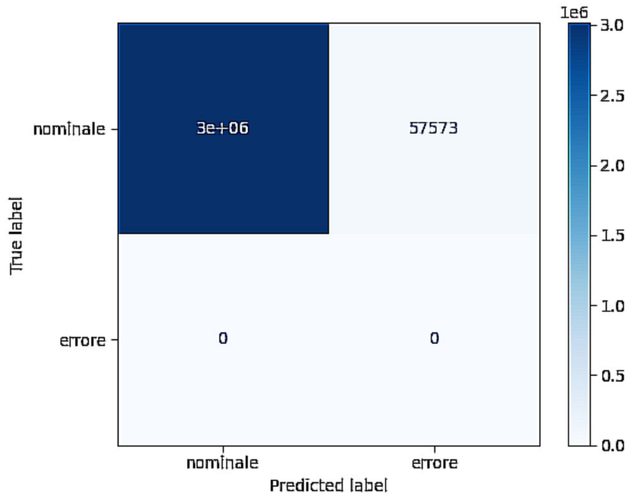


Fig. 11 Evaluation of the results in the third test (the input is a single file corresponding to an error case)

that the proposed solution, developed and tested within a real-case scenario, is valid in terms of precision, accuracy, and response time.

6.1 Future steps

Future studies and developments related to this work will mainly address the following eight goals:

1. The solution will be deployed in production, thus allowing the evaluation of its real impact on the company's processes;
2. The ML model will be extended by integrating a multi-classification approach to the already implemented binary classification. The latest aims to detect the presence of faults, while the multi-classification will identify the kind of faults.
3. In collaboration with the stakeholders' company, specific thresholds for the number of predicted faults will be identified. These thresholds will help to choose which conditions are acceptable and which are not.
4. Visual/acoustic signals and augmented reality applications will be implemented to communicate to the operators the results elaborated by the model.
5. A new method based on statistics (and in particular using *binning* [30]) will be studied to pre-process the input dataset and extrapolate values of interest to be fed to the model, instead of resampling the data (this activity is ongoing). In addition, a new approach to extract features will be studied, using a Recurrent Neural Network model.
6. The possibility to incrementally train the ML model through different steps will be implemented (in the proposed solution, the model is trained in a single step).
7. The potential of another different ML model implementation approach will be investigated. The idea is to implement a different model for each specific condition of faults, and all these models will be run in parallel, thus improving the precision and accuracy of the overall model.
8. The proposed approach will be revised to be compliant with the human-centricity of Industry 5.0 by analyzing a collaborative model between humans and (physical and virtual) machines [34].

Acknowledgements This research work was carried out under the framework of the "Regions and Digital Innovation Hubs alliance for AI-driven digital transformation of European Manufacturing SMEs" (AI REGIO) project, funded by the European Union Framework Programme for Research and Innovation Horizon 2020 under Grant Agreement n°952003.

Author Contributions All authors contributed equally to this work.

Funding Open access funding provided by Consiglio Nazionale Delle Ricerche (CNR) within the CRUI-CARE Agreement.

Declarations

Conflict of Interest The authors declare no competing interests.

References

1. (2015) An Introduction to MATLAB® Programming and Numerical Methods for Engineers. In: Bayen AM, Siauw T (eds) An introduction to MATLAB® programming and numerical methods for engineers. Academic Press, Boston, p iii. https://doi.org/10.1016/B978-0-12-420228-3.00021-X
2. (2022) AI REGIO digital twin for predictive maintenance; Ai4europe. https://www.ai4europe.eu/research/ai-catalog/ai-regio-digital-twin-predictive-maintenance. Accessed 01 Oct 2023
3. (2022) Digital twin for industrial faults predictive maintenance zenodhttps://doi.org/10.5281/zenodo.6403216. Accessed 01 Oct 2023
4. (2023) AI REGIO Project. https://www.airegio-project.eu/. Accessed 01 Oct 2023
5. (2023) DIAdem homepage. https://www.ni.com/it-it/shop/software/products/diadem.html. Accessed 01 Oct 2023
6. (2023) Jupyter homepage. https://jupyter.org. Accessed 01 Oct 2023
7. (2023) Ni TDMS file format - what is a TDMS file? https://www.ni.com/en/support/documentation/supplemental/06/the-ni-tdms-file-format.html. Accessed 01 Oct 2023
8. (2023) sklearn metrics. https://scikit-learn.org/stable/modules/generated/sklearn.metrics.precision_recall_fscore_support.html. Accessed 01 Oct 2023
9. Aivaliotis P, Georgoulias K, Chryssolouris G (2017) A RUL calculation approach based on physical-based simulation models for predictive maintenance. In: 2017 International conference on engineering, technology and innovation (ICE/ITMC), pp 1243–1246. https://doi.org/10.1109/ICE.2017.8280022
10. Aly M (2005) Survey on multiclass classification methods
11. Atamuradov V, Medjaher K, Dersin P, Lamoureux B, Zerhouni N (2017) Prognostics and health management for maintenance practitioners - review, implementation and tools evaluation. Int J Progn Health Manag 8(3):1–31. https://doi.org/10.36001/ijphm.2017.v8i3.2667
12. Booyse W, Wilke DN, Heyns S (2020) Deep digital twins for detection, diagnostics and prognostics. Mech Syst Signal Process 140:106612. https://doi.org/10.1016/j.ymssp.2019.106612

13. Bradley AP (1997) The use of the area under the roc curve in the evaluation of machine learning algorithms. Pattern Recognit. https://doi.org/10.1016/S0031-3203(96)00142-2
14. Carvalho TP, Soares FA, Vita R, Francisco RDP, Basto JP, Alcalá S (2019) A systematic literature review of machine learning methods applied to predictive maintenance. Comput Ind Eng 137:106024. https://doi.org/10.1016/j.cie.2019.106024
15. Ciaburro G (2022) Machine fault detection methods based on machine learning algorithms: a review. Math Biosci Eng 19(11):11453–11490. https://doi.org/10.3934/mbe.2022534
16. Crowther PS, Cox RJ (2005) A method for optimal division of data sets for use in neural networks. In: International conference on knowledge-based and intelligent information and engineering systems, Springer, pp 1–https://doi.org/10.1007/11554028_1
17. Maulud DH, Abdulazeez AM (2020) A review on linear regression comprehensive in machine learning. J Appl Res Sci Tech Trend. https://doi.org/10.38094/jastt1457
18. Deutsch J, He D (2018) Using deep learning-based approach to predict remaining useful life of rotating components. IEEE Trans Syst Man Cybern Syst 48(1):11–20. https://doi.org/10.1109/TSMC.2017.2697842
19. Federal Standard 1037C (1996) Glossary of telecommunication terms. Standard
20. Grieves M, Vickers J (2017) Digital twin: mitigating unpredictable, undesirable emergent behavior in complex systems, Springer, Cham, pp 85–113. https://doi.org/10.1007/978-3-319-38756-7_4
21. Hilbe JM (2009) CRC press
22. Errandonea I, Beltran S, Arrizabalaga S (2020) Digital twin for maintenance: a literature review. Comput Ind. https://doi.org/10.1016/j.compind.2020.103316
23. Li JJ, Tong X (2020) Statistical hypothesis testing versus machine learning binary classification: distinctions and guidelines. Patterns. https://doi.org/10.1016/j.patter.2020.100115
24. Dalzochioa J, Pignaton E, Kunsta R (2020) Machine learning and reasoning for predictive maintenance in industry 4.0: current status and challenges. Comput Ind. https://doi.org/10.1016/j.compind.2020.103298
25. Kiangala KS, Wang Z (2018) Initiating predictive maintenance for a conveyor motor in a bottling plant using industry 4.0 concepts. Int J Adv Manuf Tech 97:3251–3271. https://doi.org/10.1007/s00170-018-2093-8
26. Luo W, Hu T, Ye Y, Zhang C, Wei Y (2020) A hybrid predictive maintenance approach for CNC machine tool driven by digital twin. Robot Comput-Integr Manuf 65:101974. https://doi.org/10.1016/j.rcim.2020.101974
27. Lévesque L (2014) Nyquist sampling theorem: understanding the illusion of a spinning wheel captured with a video camera. Phys Educ 49(6):697. https://doi.org/10.1088/0031-9120/49/6/697
28. Majnik B (2013) Matjaž: ROC analysis of classifiers in machine learning: a survey. Intell Data Anal. https://doi.org/10.3233/IDA-130592
29. Malhotra R (2015) A systematic review of machine learning techniques for software fault prediction. Appl Soft Comput 27:504–518. https://doi.org/10.1016/j.asoc.2014.11.023
30. Martin BR (2012) Chapter 1 - statistics, experiments, and data. In: Martin BR (ed) Statistics for Physical Science, Academic Press, Boston, pp 1–2https://doi.org/10.1016/B978-0-12-387760-4.00001-9. https://www.sciencedirect.com/science/article/pii/B9780123877604000019
31. McKinney W (2011) Pandas: a foundational python library for data analysis and statistics. Python for high performance and scientific computing
32. Melesse TY, Pasquale VD, Riemma S (2020) Digital twin models in industrial operations: a systematic literature review. Procedia Manuf 42:267–272. https://doi.org/10.1016/j.promfg.2020.02.084. International Conference on Industry 4.0 and Smart Manufacturing (ISM 2019)
33. Meraghni S, Terrissa LS, Yue M, Ma J, Jemei S, Zerhouni N (2021) A data-driven digital-twin prognostics method for proton exchange membrane fuel cell remaining useful life prediction. Int J Hydrogen Energy 46(2):2555–2564. https://doi.org/10.1016/j.ijhydene.2020.10.108
34. Modoni GE, Sacco M (2023) A human digital-twin-based framework driving human centricity towards industry 5.0. Sensors 23(13). https://doi.org/10.3390/s23136054
35. Modoni GE, Caldarola EG, Sacco M, Terkaj W (2019) Synchronizing physical and digital factory: benefits and technical challenges. Procedia Cirp 79:472–477. https://doi.org/10.1016/j.procir.2019.02.125
36. Moradi R, Groth KM (2020) Modernizing risk assessment: a systematic integration of PRA and PHM techniques. Reliab Eng Syst Saf 204:107194. https://doi.org/10.1016/j.ress.2020.107194
37. Gorur OC, Sivrikaya F, Yu X (2021) Integrating predictive maintenance in adaptive process scheduling for a safe and efficient industrial process. Appl Sci. https://doi.org/10.3390/app11115042
38. García-Teodoro P, Maciá-Fernández G, Díaz-Verdejo J (2008) Anomaly-based network intrusion detection: techniques, systems and challenges. Comput Secur. https://doi.org/10.1016/j.cose.2008.08.003
39. Bortnowski P, Król R, Kawalec W (2022) Identification of conveyor belt tension with the use of its transverse vibration frequencies. Measurement. https://doi.org/10.1016/j.measurement.2022.110706
40. Rauber TW, da Silva Loca AL, de Assis Boldt F et al (2021) An experimental methodology to evaluate machine learning methods for fault diagnosis based on vibration signals. Expert Syst App 167:114022. https://doi.org/10.1016/j.eswa.2020.114022
41. Rigatti SJ (2017) Random forest. J Insur Med 47(1):31–3. https://doi.org/10.17849/insm-47-01-31-39.1
42. Roosefert Mohan T, Preetha Roselyn J, Annie Uthra R, Devaraj D, Umachandran K (2021) Intelligent machine learning based total productive maintenance approach for achieving zero downtime in industrial machinery. Comput Ind Eng 157:107267. https://doi.org/10.1016/j.cie.2021.107267
43. Omar S, Jebur HH, Ngadi A (2013) Machine learning techniques for anomaly detection: an overview. Int J Comput Appl
44. Samuel PD, Pines DJ (2005) A review of vibration-based techniques for helicopter transmission diagnostics. J Sound Vib 282(1):475–508. https://doi.org/10.1016/j.jsv.2004.02.058
45. Shannon CE (1948) A mathematical theory of communication. Bell Syst Tech J 27(3):379–423. https://doi.org/10.1002/j.1538-7305.1948.tb01338.x, conference Name: The Bell System Technical Journal
46. Shen T, Li B (2024) Digital twins in additive manufacturing: a state-of-the-art review. Int J Adv Manuf Tech 1–3. https://doi.org/10.1007/s00170-024-13092-y
47. Steenwinckel B, De Paepe D, Vanden Hautte S (2021) Flags: a methodology for adaptive anomaly detection and root cause analysis on sensor data streams by fusing expert knowledge with machine learning. Future Gener Comput Syst 116:30–48. https://doi.org/10.1016/j.future.2020.10.015
48. Suthaharan S, Suthaharan S (2016) Support vector machine. Machine learning models and algorithms for big data classification: thinking with examples for effective learning, pp 207–235. https://doi.org/10.1007/978-1-4899-7641-3_9
49. Swanson L (2001) Linking maintenance strategies to performance. Int J Prod Econ. https://doi.org/10.1016/S0925-5273(00)00067-0
50. Zonta T, da Costa CA (2020) Predictive maintenance in the industry 4.0: a systematic literature review. Comput Ind Eng. https://doi.org/10.1016/j.cie.2020.106889

51. Tiddens W, Braaksma J, Tinga T (2022) Exploring predictive maintenance applications in industry. J Qual Maint Eng 28(1):68–85. https://doi.org/10.1108/JQME-05-2020-0029
52. van Dinter R, Tekinerdogan B, Catal C (2022) Predictive maintenance using digital twins: a systematic literature review. Inf Softw Technol 151:107008. https://doi.org/10.1016/j.infsof.2022.107008
53. Wu D, Jennings C, Terpenny J, Gao RX, Kumara S (2017) A comparative study on machine learning algorithms for smart manufacturing: tool wear prediction using random forests. J Manuf Sci Eng 139(7). https://doi.org/10.1115/1.4036350
54. Xiong M, Wang H, Fu Q, Xu Y (2021) Digital twin-driven aero-engine intelligent predictive maintenance. Int J Adv Manuf Technol 114(11–12):3751–3761. https://doi.org/10.1007/s00170-021-06976-w
55. Xu M, Watanachaturaporn P, Varshney PK, Arora MK (2005) Decision tree regression for soft classification of remote sensing data. Remote Sens Environ 97(3):322–336. https://doi.org/10.1016/j.rse.2005.05.008
56. Chandola Y, Virmani J, Bhadauria HS, Kumar P (2021) Deep learning for chest radiographs. Comput Aided Classif Acad Press. https://doi.org/10.1016/C2020-0-03809-0
57. Ran Y, Lin P, Zhou X (2019) A survey of predictive maintenance: systems, purposes and approaches. Electr Eng Syst Sci. https://doi.org/10.48550/arXiv.1912.07383
58. Yu J, Song Y, Tang D, Dai J (2021) A digital twin approach based on nonparametric Bayesian network for complex system health monitoring. J Manuf Syst 58:293–304. https://doi.org/10.1016/j.jmsy.2020.07.005
59. Zhang W, Yang D, Wang H (2019) Data-driven methods for predictive maintenance of industrial equipment: a survey. IEEE Syst J 13(3):2213–2227. https://doi.org/10.1109/JSYST.2019.2905565
60. Jia Z, Sharma A (2021) Review on engine vibration fault analysis based on data mining. J Vibro Eng. https://doi.org/10.21595/jve.2021.21928
61. Zio E (2022) Prognostics and health management (PHM): where are we and where do we (need to) go in theory and practice. Reliab Eng Syst Saf 218:108119