

Data-Driven Predictive Maintenance for Component Life-Cycle Extension

Margarida Moreira¹ ^a, Eliseu Pereira^{1,2} ^b and Gil Gonçalves^{1,2} ^c

¹Faculty of Engineering, University of Porto, Portugal

²SYSTEC-ARISE, Faculty of Engineering, University of Porto, Portugal

Keywords: Predictive Maintenance, Industry 4.0, Remaining Useful Life (RUL), Data-Driven Methods, Survival Analysis.

Abstract: In the era of Industry 4.0, predictive maintenance is crucial for optimizing operational efficiency and reducing downtime. Traditional maintenance strategies often cost more and are less reliable, making advanced predictive models appealing. This paper assesses the effectiveness of different survival analysis models, such as Cox Proportional Hazards, Random Survival Forests (RSF), Gradient Boosting Survival Analysis (GBSA), and Survival Support Vector Machines (FS-SVM), in predicting equipment failures. The models were tested on datasets from Gorenje and Microsoft Azure, achieving C-index values on test data such as 0.792 on the Cox Model, 0.601 using RSF, 0.579 using the GBSA model and 0.514 when using the FS-SVM model. These results demonstrate the models' potential for accurate failure prediction, with FS-SVM showing significant improvement in test data compared to its training performance. This study provides a comprehensive evaluation of survival analysis methods in an industrial context and develops a user-friendly dashboard for real-time maintenance decision-making. Integrating survival analysis into Industry 4.0 frameworks can significantly enhance predictive maintenance strategies, paving the way for more efficient and reliable industrial operations.

1 INTRODUCTION

In the industrial setting, maintaining system components is a critical yet challenging aspect of operations management. Traditional maintenance approaches, often reactive or time-based, have limitations in terms of cost efficiency and operational effectiveness. With the advancement of Industry 4.0 development, the integration of digital technologies into manufacturing processes has revolutionized maintenance strategies, making predictive maintenance not just beneficial but essential. Predictive maintenance utilizes advanced data analytics and machine learning to predict component failures before they occur, allowing for timely and cost-effective interventions.


Traditional maintenance methods, such as reactive maintenance and preventive maintenance, often result in inefficiencies. Reactive maintenance can lead to unexpected downtime and high repair costs, while preventive maintenance may cause over-maintenance


and unnecessary expenses. Although there are some attempts in this area (Rossini et al., 2021), there is a critical need for predictive maintenance models that can optimize maintenance schedules, reduce costs, and improve equipment uptime.


This work focuses on implementing and evaluating various predictive maintenance models using survival analysis techniques. The methods include Cox Proportional Hazards, Random Survival Forest (RSF), Gradient Boosting Survival Analysis (GBSA), and Survival Support Vector Machines (FS-SVM). The data used for this analysis comes from two sources: a dataset from Gorenje company related to a spot welding robot and a Microsoft Azure predictive maintenance dataset from Kaggle. It will show the preprocessing of data ingested, the training and evaluation of different survival analysis models and a simulation of a maintenance scenario.

The work presented in this paper aims to provide a comprehensive evaluation of predictive maintenance models based on survival analysis and their integration within an Industry 4.0 framework.

This paper is organised as follows: Section 2 presents the literature review. Section 3 presents the

^a  <https://orcid.org/0009-0000-7290-6467>

^b  <https://orcid.org/0000-0003-3893-3845>

^c  <https://orcid.org/0000-0001-7757-7308>

system's architecture implementation. Section 4 discusses the main results obtained after the implementation. Finally, Section 5 provides the conclusions and future work that can still be done.

2 LITERATURE REVIEW

The following literature review summarises Industry 4.0 and the different types of industrial maintenance that are used, the different predictive maintenance models, survival analysis and its performance metrics and in the end, the concept of data preparation and feature engineering and ends with the presentation of different data-driven methods. It concludes with a summary of the primary findings from the reviewed studies.

2.1 Types of Industrial Maintenance

Predictive Maintenance (PdM) has been increasing in popularity in the last few years, mainly in Industry 4.0. It is estimated that "The impact of maintenance represents a total of 15 to 60% of the total costs of operating of all manufacturing" (Zonta et al., 2020).

Nowadays, the three main types of industrial maintenance are reactive maintenance, preventive maintenance, and predictive maintenance.

Reactive maintenance is characterized by responding to equipment failures, in other words, without prior planning. Preventive maintenance refers to activities performed on a fixed schedule, normally in regular intervals. This technique reduces the likelihood of equipment failure. However, it may lead to over-maintenance, where components are often fixed more than necessary. This will cause an increase in costs, which will cause the companies to lose more than necessary.

Predictive maintenance relies on the real-time monitoring of equipment conditions to predict when maintenance should be performed. It uses a combination of data analysis, machine learning, and sensor technologies to indicate potential failures before they happen. The main goal is to optimize maintenance frequency, reduce costs, and increase equipment uptime. It predicts and prevents component failures. It has been widely adopted due to its superior results compared to the previously mentioned techniques (Moat and Coleman, 2021).

In the last years, a new kind of maintenance has been developed. This type, called proactive maintenance, is considered the latest type in this evolution process. It consists of identifying the source

of the fault and remedying conditions that can fail (Ramezani et al., 2023).

2.2 Predictive Maintenance Models

There are three main models, as presented below:

- **Condition-Based Maintenance (CBM)**
- **Prognostics and Health Management (PHM)**
- **Remaining Useful Life (RUL)**

The main focus of this article will be on the Remaining Useful Life (RUL) strategy. RUL refers to the estimated lifespan of a device before it requires maintenance or replacement and also the remaining time until the equipment's health conditions reach the failure threshold (Gupta et al., 2024). The accuracy of the RUL estimation depends on the quality and quantity of available health monitoring data and the methods used for such analysis (Ferreira and Gonçalves, 2022).

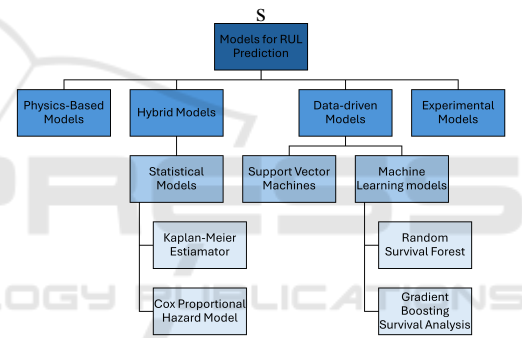


Figure 1: Models for RUL prediction, based on (Achouch et al., 2022).

RUL uses different models for its prediction-making, as shown in figure 1.

2.3 Survival Analysis

Survival analysis encompasses various methods for analyzing data related to time-to-event outcomes. It is also comprehended as a reliability theory and an event history analysis. The term 'survival' is equivalent to probabilities in survival analysis, implying that survival can be 'measured'. The survival function can be expressed as (Moat and Coleman, 2021):

$$S(t) = P(T > t) = 1 - F(t) \quad (1)$$

Where $S(t)$ is the survival function representing the probability that a system or component survives beyond a certain time t . The T is the number of cycles until the component is worn out. The whole function describes the probability that the component is not worn out by t cycles.

The survival function present on 1 can be estimated using the Kaplan-Meier Estimator (KM). It is a non-parametric estimator used to measure the fraction of living subjects after a certain amount of time with respect to one or more specific events (Yang et al., 2022). It is defined as:

$$\hat{S}(t) = \prod_{t_i \leq t} \left(1 - \frac{d_i}{n_i}\right) \quad (2)$$

In the equation 2 (Frumosu et al., 2020), the t_i is the time when at least one event (worn-out) occurred, the d_i is the total number of events (worn-out) at t number of shots and n_i is the number of components at risk at t number of shots.

The Kaplan-Meier survival curve starts from 1 when all components work correctly and decreases to 0 as events (worn-outs) occur.

To evaluate and compare the performance of survival analysis models, the scikit-survival module from Python can be used, which offers comprehensive tools and metrics tailored for survival data. The bigger difference between survival analysis and traditional machine learning is the fact that parts of the training data can only be partially observed. (Pölsterl, 2024). Consequently, the performance metrics used are the Concordance index (C-index), Brier Score, integrated Brier score and Time-Dependent ROC.

2.4 Performance Metrics

Performance metrics are used to evaluate and estimate the survival models. Some of the most common include the Concordance Index (C-index), Brier Score and the integrated Brier score (IBS) (Zeng et al., 2023).

The C-index, also called Harrell's Index (Frumosu et al., 2020), is one of the most common evaluation metrics used for measuring the quality and efficiency of survival models. It ranges from 0 to 1, where results closer to 1 indicate a better performance of the model (De Santis et al., 2022).

The Brier score measures the difference between the predicted probability and the true outcome, with higher scores (Shen et al., 2024). The Brier score ranges from 0 to 1, and contrarily to C-index, 0 indicates a perfect prediction while 1 indicates poorer prediction accuracy and calibration.

The IBS extends the Brier score to the context of time-to-event data, such as survival analysis. It reflects calibration over all time points, with a smaller value indicating greater accuracy. Like the Brier score, a smaller value indicates a greater accuracy (Tran et al., 2023).

2.5 Data Preparation and Feature Engineering

The dataset is usually chosen based on the data collected by the sensors, more precisely, temperature, voltage, pressure, vibration, and vibration, among others.

Normally, two different datasets are used for testing, one real and another one already used in other works, to compare the results.

The dataset that would be chosen to be compared with the real one should contain a large number of points in order to have more accurate results.

The first step normally consists of the removal of low-coverage parameters, replacing missing values with the median in order to maintain data integrity, removing non-numerical parameters, and eliminating non-numeric features that are not useful for analysis.

Following that, the aggregation process begins by summarizing and consolidating the data. Following that, the feature selection is realized, which starts by first summarising and consolidating the data and then proceeding with the selection of relevant features and removal of redundant ones.

The use of multiple datasets allows for the validation of the models that are being tested and, consequently, the generalization of the results. It also allows us to compare the efficiency of predictive maintenance tools against established benchmarks. Finally, the comparative dataset can help interpret results from the real dataset and vice-versa.

When these processes have been concluded, some data transformation is performed, where the normalization of the features occurs.

To end this procedure, the data is normally split into training and testing to posteriorly evaluate the model's performance.

Feature engineering is a technique that creates new features based on the existing ones from the dataset (Zhou et al., 2022). Some of the most common techniques are feature creation and feature extraction. Feature creation involves generating new ones by creating new variables used in the machine learning models (Madasamy et al., 2023). Feature extraction is based on extracting the most relevant features from a raw data sample used as input to the models. The most common domains to be used to extract these variables are time, frequency and time-frequency (Gupta et al., 2024).

2.6 Data-Driven Methods

To predict the behaviour of the machines and determine when an intervention should be conducted to re-

store some components, it is necessary to evaluate and analyze the available data from the sensors.

There are three main types of data-driven methods, as shown below (Arena et al., 2021):

1. **Statistical approaches;**
2. **Stochastic approaches;**
3. **Machine Learning Techniques.**

The first two methodologies are best suited for examining complex systems whose evolutionary processes are not straightforward to anticipate.

The third one is the approach that will be considered in this work.

This can be summarised in the following figure 2.

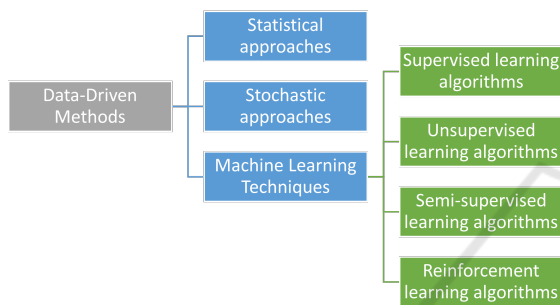


Figure 2: Types of data-driven methods and the fourth main machine learning methodologies.

Regarding machine learning algorithms, they are divided into three main categories, as shown in Figure 3:

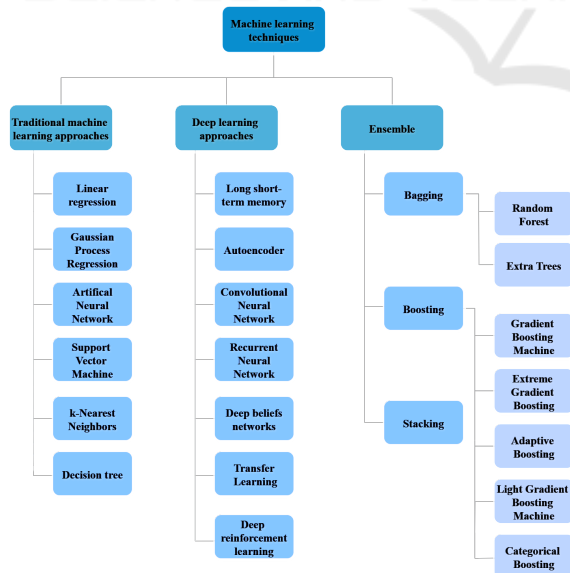


Figure 3: Machine learning algorithms divided into the three main categories.

The first technique involves using simpler algorithms that are more easily interpreted.

The second one uses multilayered artificial neural networks to create a more suitable mapping function between given inputs and outputs. It normally requires a large amount of data to achieve better accuracy.

Ensemble learning methods involve combining two or more ML algorithms to achieve better performance than when using the individual algorithms separately. Instead of depending on a single model, the predictions from the individual algorithms are combined using a specific rule to produce a more accurate single prediction. Ensemble methods are typically categorized into parallel (bagging) and sequential ensembles (boosting) (Mienye and Sun, 2022).

2.7 Gap Analysis

The reviewed studies present different methods and models to predict the remaining useful life of different components. Some of the most common approaches are traditional machine learning techniques, and with the development of technology, there are more studies where deep learning methods are being developed.

Nevertheless, significant gaps prevail when using a survival approach. When searching for articles that used this package in Industry 4.0, the search returned null results on Scopus. However, this technique is widely used in the field of medicine. By leveraging the strengths of survival analysis, industries could improve their predictive maintenance strategies, leading to more efficient operations and reduced downtime.

3 IMPLEMENTATION

In this chapter, the implementation of the Data-Driven Predictive Maintenance system is designed to extend the life cycle of critical components. The implementation is structured into distinct modules, each responsible for a specific aspect of the data processing and maintenance prediction workflow. This modular approach ensures scalability, flexibility, and ease of maintenance.

3.1 Systems' Architecture

The architecture of the system can be categorized into the following modules, as shown in the Figure 4 below:

Each module plays a critical role in the overall functionality of the predictive maintenance system. Figure 1 illustrates the interaction between these modules and their respective components.

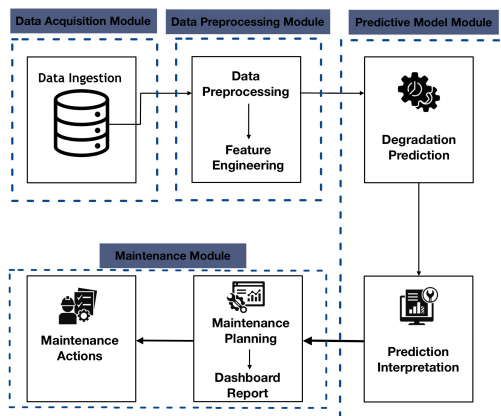


Figure 4: Systems' Architecture Overview.

The data acquisition module is responsible for collecting and ingesting raw data from various sensors and data sources. This data forms the backbone of the predictive maintenance system, providing the necessary inputs for subsequent analysis and modelling.

Once the data is ingested, it undergoes preprocessing to clean, normalize, and transform it into a suitable format for analysis on the data preprocessing module. This module includes feature engineering processes that extract meaningful features from raw data, enhancing the accuracy and efficiency of the predictive models.

In the predictive model module, machine learning algorithms are employed to predict the degradation of components. These predictions help prevent unexpected failures and extend the component life cycle.

The maintenance module interprets the predictions and provides actionable insights for maintenance planning. It generates dashboard reports that aid in decision-making and execution of maintenance actions. This module ensures that the right maintenance activities are performed at the right time based on data-driven insights.

For the validation of the solution, two datasets were used: one provided by Gorenje company that is related to component welding and another that was taken from Kaggle, a Microsoft Azure predictive maintenance dataset created for predictive maintenance model building, being a synthetic dataset.

For the development of the architecture, the open-source scikit-learn package used version 1.3.2, the scikit-survival package used version 0.22.2, the scipy package version used version 1.10.1, and the Python version 3.9.0.

3.2 Data Acquisition and Preprocessing

The data related to the company Gorenje were related to a spot welding robot. It was presented in diverse CSV files, with the minimum, mean and maximum current from 2022-09-30 until 2023-03-08. It presented a total of 629444 samples. In another file, there was information related to the values of a rotary table, with a total of 96869 samples. There was also a file that presented the time of failure when the failure occurred, the error code number, and a total of 248 samples.

Based on these files, the first step was to merge the sensor data with the counter data to align timestamps correctly, using the *pd.concat* function, incorporating sensor readings and counter value.

After merging the data, the difference between successive counter values was calculated in order to determine the machine cycles. For each cycle, statistics such as the mean and standard deviation of current sensor readings were computed. Each cycle was then labelled as either working or resulting in a failure based on historical failure data. This labelling was used to train predictive models.

The Microsoft Azure predictive maintenance dataset was designed for building predictive maintenance models and includes various operational metrics and sensor readings, all collected hourly. The dataset comprises several CSV files such as *PdM_telemetry.csv* that contain hourly averages of voltage, rotation, pressure, and vibration from 100 machines in 2015, totalling 876,101 samples. *PdM_errors.csv* includes 3,920 samples of machine errors, detailing the datetime, machineID, and specific error. These errors occur without causing machine shutdowns. *PdM_maint.csv* contains 3,287 samples with the datetime, machineID, and the component that was replaced. It records both proactive maintenance during scheduled visits and reactive maintenance after component failures. *PdM_failures.csv* is a subset of the maintenance data, showing component replacements due to failures and *PdM_machines.csv* provides each machine's model type and age. Only telemetry data from 2015 was used in the preprocessing phase.

3.3 Predictive Models

For both Gorenje and Microsoft Azure datasets, the preprocessed data obtained in the previous module was used to train various survival analysis models. First, the preprocessed data was loaded, and then relevant features, including cycle time and cumulative metrics, were selected. The dataset was divided into

a training set and a test set in order to evaluate the performance of each model and proceed to the validation of the system. The x_{train} contained the features of each dataset, while the y_{train} included the status of the components (if they have failed) and the time to failure for each component. After dividing the datasets, the first step was the training phase. During training, the model used x_{train} (cycles) and y_{train} as inputs. After the model was trained, the outputs produced were the different models and the associated c_index train for each model. After completing the training phase, the testing phase was conducted to obtain the model predictions. It was used the x_{test} and y_{test} as inputs, and the trained models produced in the testing phase. The outputs obtained included the predictions associated with each model (such as the survival functions and the time to failure for the Fast Survival Support Vector Machine), as well as the c_index test to evaluate the performance. The different survival models that were trained and evaluated included the Cox Proportional Hazards, Random Survival Forest, Gradient Boosting Survival Analysis and Fast Survival Support Vector Machine.

For this, was created the training phase and, after that, the testing phase. In the first one, the system had as inputs the x_{train} . After this, the models were evaluated using the concordance index, and survival functions were plotted to visualize the estimated survival probabilities over time. This procedure was applied to each component individually to create this module.

After this step, the models were stored on the disk using the dump function to be used in the maintenance module.

3.3.1 Cox Proportional-Hazards Model

The Cox proportional hazard model (PH) is a semi-parametric model. It is used to investigate the effect of several variables on the time a specified event takes to happen, and it has the following hazard function (Moat and Coleman, 2021):

$$h(t, X) = h_0(t) \exp(\beta X) \quad (3)$$

In the equation 3, $h_0(t)$ is the baseline hazard function, β is the coefficient, and X is the covariate. The Cox PH model assumes that survival times t are independent, the hazard is proportional, i.e., the hazard ratio is proportional, the hazard function is the linear function of the numerical covariates, and the values of X 's do not change over time.

The Algorithm 1 was implemented as follows:

3.3.2 Random Survival Forests

The Random Survival Forest is a collection of tree-based models that ensures that each tree is built on

Data: Training data x_{train} , y_{train} , Test data x_{test} , y_{test}
Result: Trained Cox model, Feature importance, Survival functions
Initialize: Cox Proportional Hazards model cox_model ;
 $cox_model \leftarrow CoxPHSurvivalAnalysis()$;
Call: $TrainModel(cox_model)$;
Print: "Feature importance:";
Print: $pd.Series(cox_model.coef_, index=x_features)$;

Algorithm 1: Cox Proportional Hazards Model Training.

a different bootstrap sample of the original training data, consequently removing correlations between the trees. At each node, only a randomly selected subset of features and thresholds are used to evaluate the split criterion. The final predictions are made by combining the predictions of each individual tree in the ensemble. The following algorithm 2 demonstrates the implementation of this model.

Data: Training data x_{train} , y_{train} , Test data x_{test} , y_{test}
Result: Trained RSF model, Concordance index
Initialize: Random Survival Forest model rsf ;
 $rsf \leftarrow RandomSurvivalForest(n_estimators=100, min_samples_split=10, min_samples_leaf=15, max_features="sqrt", n_jobs=-1, random_state=20)$;
Fit the model: $rsf.fit(x_{train}, y_{train})$;
Evaluate the model using concordance index;
 $cindex_train \leftarrow rsf.score(x_{train}, y_{train})$;
 $cindex_test \leftarrow rsf.score(x_{test}, y_{test})$;
Print: "RSF Model";
Print: "cindex train: ", $round(cindex_train, 3)$;
Print: "cindex test: ", $round(cindex_test, 3)$;

Algorithm 2: Random Survival Forest Model Training.

3.3.3 Gradient Boosting Survival Analysis

Gradient Boosting leverages the principle of strength in numbers by combining the predictions of multiple base learners to create a robust overall model.

A gradient boosted model shares similarities with a Random Survival Forest in that both rely on multiple base learners to generate an overall prediction. However, they differ in their combination methods. A Random Survival Forest fits several Survival Trees independently and averages their predictions, whereas a gradient boosted model is built sequentially in a greedy stagewise manner. The Algorithm 3 demonstrates the implementation of this model.

Data: Training data x_{train}, y_{train} , Test data x_{test}, y_{test}
Result: Trained Gradient Boosting model, Concordance index
Initialize: Gradient Boosting Survival Analysis model $gbsa$;
 $gbsa \leftarrow$ GradientBoostingSurvivalAnalysis($n_estimators=100$, $learning_rate=0.01$);
Fit the model: $gbsa.fit(x_{train}, y_{train})$;
Evaluate the model using concordance index;
 $cindex_train \leftarrow gbsa.score(x_{train}, y_{train})$;
 $cindex_test \leftarrow gbsa.score(x_{test}, y_{test})$;
Print: "Gradient Boosting Survival Analysis Model";
Print: "cindex train: ", $round(cindex_train, 3)$;
Print: "cindex test: ", $round(cindex_test, 3)$;

Algorithm 3: Gradient Boosting Survival Analysis Model Training.

3.3.4 Survival Support Vector Machines

Survival Support Vector Machines (Survival SVM) are an extension of traditional Support Vector Machines adapted for survival analysis. The Survival SVM method integrates the principles of SVM into survival analysis, aiming to find a hyperplane that maximizes the margin between different survival times. Unlike traditional SVM, which is used for classification or regression, Survival SVM deals with censored data, where the event of interest has not occurred for all subjects during the observation period. The following Algorithm 4 illustrates the implementation of this model.

3.4 Maintenance Module

This module was only applied to the second dataset since this last one presented more features and data samples.

Firstly, a simulation was created where the different models produced in each component would be called using the `load()` function. In this simulation, a machine class was created to simulate a machine with multiple components that may fail and need replacement. Inside this class, diverse methods were created, such as:

- `log_event`: that logs significant events, such as component failures and repairs, for debugging and record-keeping, including timestamps and costs associated with the maintenance of the components;

Data: Training data x_{train}, y_{train} , Test data x_{test}, y_{test}
Result: Trained FS-SVM model, Concordance index
Procedure: TrainSVM();
Initialize: Fast Survival SVM model svm_model ;
 $svm_model \leftarrow$ FastSurvivalSVM($rank_ratio=0.0$, $max_iter=1000$, $tol=1e-5$, $random_state=42$);
Print: "FS-SVM Model";
Fit the model: $svm_model.fit(x_{train}, y_{train})$;
Evaluate the model using concordance index;
 $cindex_train \leftarrow$ $concordance_index_censored(y_{train}['status'], y_{train}['ttf_comp1'], -svm_model.predict(x_{train}))$;
Print: "cindex train: ", $round(cindex_train[0], 3)$;
 $cindex_test \leftarrow$ $concordance_index_censored(y_{test}['status'], y_{test}['ttf_comp1'], -svm_model.predict(x_{test}))$;
Print: "cindex test: ", $round(cindex_test[0], 3)$;
Return: $svm_model, cindex_train, cindex_test$;

Algorithm 4: Fast Survival SVM Model Training.

- `predict_time_to_failure`: where is specified the model to predict the remaining useful life of a component based on its features;
- `run_machine`: It simulates the machine operation and decrements the remaining life of components, checks for failures, and starts repairs when necessary. It also checks component lifetimes at the end of each simulated day;
- `log_remaining_times`: It logs the remaining times for all components at regular intervals for monitoring purposes;
- `update_remaining_times`: Regularly updates the remaining times of components every 8 hours;
- `start_repair`: Manages the repair process of a failed component, including scheduling repairs within working hours and handling repair costs;
- `complete_repair`: Completes the repair of a component and updates its predicted remaining life.

In this simulation, a cost of 40€ per repair of the component was also added, and 40€ was referred to as the displacement of the repairmen. It was also added some restrictions, such as the repayment only

does the maintenance of the components from Monday to Friday, from 08:00 to 17:00. Also, the first component takes four hours to be repaired, and the subsequent on that day takes only one hour.

After the simulation was performed, a dashboard was created with the support of Streamlit. This Dashboard first opens a webpage where the people can select a date and time between 01/01/2015 06:00 and 01/01/2016 06:00. The person can also choose the time of simulation he wants to produce. This time is referred to as the weeks. After the person presses the button of the simulation, a report with the total cost associated with the repairs and displacement is demonstrated, as well as the start and end date of the simulation. It then shows a table with the repair log of each component, the time it started being repaired and the model that was used to predict the failure. It ends with showing graphs, one with the remaining time of each component during the weeks simulated and another with the failures of each component.

4 EXPERIMENTS AND RESULTS

The results can be divided between the performance of each model for each dataset and the dashboard created with the help of the Streamlit application.

4.1 Predictive Models Performance

The first results were obtained after the training of the different models. The results were evaluated using the C-Index for training and for test samples.

The initial findings were performed on Gorenje's dataset and can be observed in the table 1.

Table 1: Evaluation of the different models trained using C-index performance measure for the Gorenje's dataset.

Model	Cox Model	RSF	GBSA	FS-SVM
C-index train	0.763	0.897	0.789	0.722
C-index test	0.974	0.791	0.789	0.865

The Cox Model was observed to perform properly on the training data (0.763), but it performed exceptionally well on the test data (0.974). This unusually high test c-index compared to the training c-index might indicate potential overfitting. Generally, the test performance is expected to be slightly lower or comparable to the training performance.

RSF shows high performance on the training data (0.897), suggesting it has learned it well. However,

its performance drops on the test data (0.791), maintaining a good level of predictive accuracy on the test data.

The GBSA model shows consistent performance on both training and test datasets with identical c-index values (0.789). This consistency suggests that the model generalizes well and has neither overfitted nor underfitted the data.

FS-SVM has the lowest performance on the training data (0.722) but performs significantly better on the test data (0.865). This improvement could imply that the model is better suited to the distribution of the test data.

When testing for Microsoft Azure's dataset, the performance was evaluated for each component of machine 1. The results are represented in table 2.

Based on the results, it can be confirmed that the CoxPH Model presents a pattern of overfitting across all components, with test c-index values significantly higher than training c-index values. On the other hand, the RSF model represents a good generalization on components one and four, but components two and three present a big drop between the train and test c-index, which may suggest an overfitting. The GBSA model presents a consistent performance for components one, two and four and component three shows a moderate drop in test performance, suggesting some overfitting. In the end, FS-SVM does present a consistent performance for components three and four, although the results are not that high.

4.1.1 Simulation and Dashboard Report

When simulating, based on the results of the c-index values, the models for each component were chosen. The models chosen were:

- `model_choice_comp1 = 'rsf'`
- `model_choice_comp2 = 'gbsa'`
- `model_choice_comp3 = 'svm'`
- `model_choice_comp4 = 'svm'`

These models could be changed in the simulation script.

With the created interface, the person can choose the date and the weeks they want to simulate, according to the rules mentioned in section 3.4. For the following results, the data chosen was 2015/01/05 06:00, and the time for simulation was 5 weeks. The first ones to acknowledge are the simulation report and the repair log, present in the figure 5.

Table 2: Evaluation of for the different models trained using C-index performance measure, for the Microsoft Azure’s dataset.

	CoxPH		RSF		GBSA		FS-SVM	
	C-index train	C-index test	C-index train	C-index test	C-index train	C-index test	C-index train	C-index test
Comp1	0.531	0.926	0.709	0.689	0.627	0.645	0.510	0.553
Comp2	0.518	0.858	0.702	0.387	0.627	0.509	0.516	0.544
Comp3	0.480	0.649	0.751	0.458	0.722	0.543	0.511	0.496
Comp4	0.518	0.792	0.687	0.601	0.602	0.579	0.507	0.514



Figure 5: Simulation Report and Repair Log for the simulated dates.

After the report, the following charts were presented. In figure 6, it can be seen on the line chart the remaining time of each component. In figure 7, it is presented with the help of a bar chart, the times that a component failed during the simulated time.

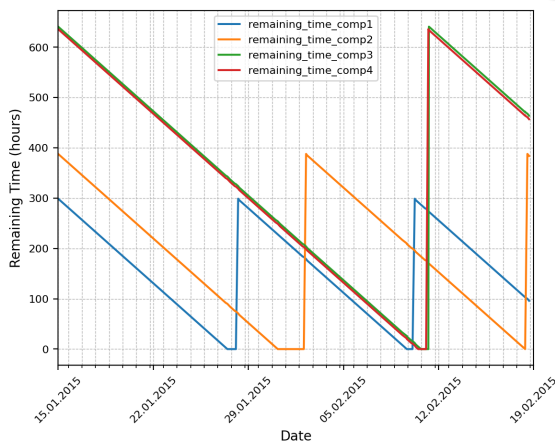


Figure 6: Remaining time for each component.

Overall, the analysis of the predictive models’ performance across different datasets highlights key insights. The CoxPH model, despite its strong performance on test data, exhibited signs of overfitting, ne-

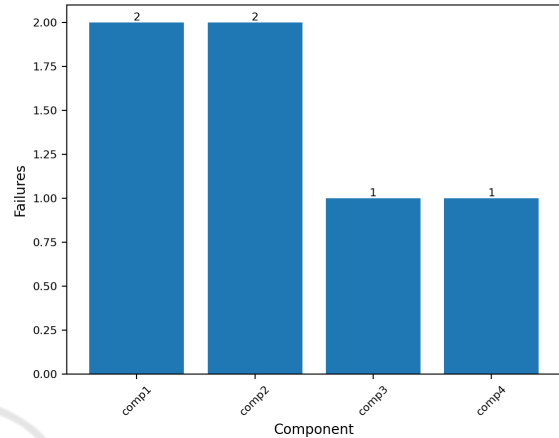


Figure 7: Failures of each component during the simulated time.

cessitating further validation and regularization techniques. The RSF model demonstrated good generalization in some components but struggled with overfitting in others, indicating the need for potential adjustments or alternative strategies for certain data subsets. GBSA consistently delivered reliable results across most components, confirming its robustness and generalizability. Finally, FS-SVM showed varied performance, excelling in some test scenarios while underperforming in others, suggesting it might benefit from further refinement or hybrid approaches.

In the simulation phase, the models selected for each component, based on their c-index values, provided a solid foundation for generating accurate predictions and facilitating the creation of a user-friendly interface for real-time decision-making. This comprehensive approach not only validated the chosen models but also highlighted areas for further enhancement.

5 CONCLUSIONS AND FUTURE WORK

The application of survival analysis methods for predictive maintenance within the framework of Industry 4.0 has demonstrated significant potential and effectiveness. This document evaluated several predictive models, including the Cox Proportional Hazards model, Random Survival Forests (RSF), Gradi-

ent Boosting Survival Analysis (GBSA), and Survival Support Vector Machines (FS-SVM), on datasets from both Gorenje and Microsoft Azure.

The results reveal that survival methods can be highly effective for predicting maintenance needs in an industrial setting. As it was seen, the Cox model showed strong performance on test data, although it exhibited overfitting, which necessitates further investigation and validation. The RSF model displayed good generalization in some components but also signs of overfitting in others, indicating the need for model adjustments. The GBSA model consistently performed well across different datasets, suggesting it is a robust choice for generalizable predictive maintenance tasks. Meanwhile, FS-SVM showed potential, especially in specific test scenarios, though it may benefit from further refinement.

A key advantage of using these survival analysis models is the ability to generate Remaining Useful Life (RUL) estimates, which are strongly important for planning maintenance activities. The simulation and dashboard interface developed using Streamlit provided a practical tool for visualizing these predictions, allowing maintenance schedules to be optimized to minimize costs and downtime. The interface facilitated real-time decision-making by enabling users to simulate various scenarios and view detailed reports on component health and predicted failures.

Despite the promising results, there are several areas for future work to enhance the application of predictive maintenance, which can be highlighted:

- **Real-Time Data Integration:** Implementing models that can handle real-time data streams will improve the timeliness and accuracy of maintenance predictions. This integration will also allow for dynamic updating of RUL estimates and more responsive maintenance planning.
- **Enhanced User Interface:** Developing a more complex and detailed user interface could provide additional insights and functionality, such as more granular control over simulation parameters and detailed visualizations of component health trends over time.
- **Advanced Model Refinement:** Further refinement of models, particularly those showing signs of overfitting or underperformance, could involve hybrid approaches that combine the strengths of multiple models or the integration of additional data sources to improve prediction accuracy. The use of other performance measures, such as the Brier Score mentioned in section 2.4, can help to compare the results obtained in the future.
- **Scalability Testing:** Testing the models and simu-

lation framework in larger, more diverse industrial settings will ensure the scalability and robustness of the solutions proposed.

In conclusion, this study emphasises the potential of survival analysis methods in improving predictive maintenance within Industry 4.0 environments. By using advanced predictive models and user-friendly interfaces, industries can move towards more efficient, cost-effective, and reliable maintenance strategies. Continued research and development in this field will further solidify these methods' role in the next generation of intelligent and efficient maintenance.

ACKNOWLEDGEMENTS

This work was partially supported by the HORIZONCL4-2021-TWIN-TRANSITION-01 openZDM project, under Grant Agreement No. 101058673.

REFERENCES

- Achouch, M., Dimitrova, M., Ziane, K., Sattarpanah Karganroudi, S., Dhoub, R., Ibrahim, H., and Adda, M. (2022). On Predictive Maintenance in Industry 4.0: Overview, Models, and Challenges. *Applied Sciences*, 12(16):8081.
- Arena, F., Collotta, M., Luca, L., Ruggieri, M., and Termine, F. G. (2021). Predictive Maintenance in the Automotive Sector: A Literature Review. *Mathematical and Computational Applications*, 27(1):2.
- De Santis, R. B., Gontijo, T. S., and Costa, M. A. (2022). A Data-Driven Framework for Small Hydroelectric Plant Prognosis Using Tsfresh and Machine Learning Survival Models. *Sensors*, 23(1):12.
- Ferreira, C. and Gonçalves, G. (2022). Remaining Useful Life prediction and challenges: A literature review on the use of Machine Learning Methods. *Journal of Manufacturing Systems*, 63:550–562.
- Frumosu, F. D., Rønsch, G. Ø., and Kulahci, M. (2020). Mould wear-out prediction in the plastic injection moulding industry: a case study. *International Journal of Computer Integrated Manufacturing*, 33(12):1245–1258.
- Gupta, S., Kumar, A., and Maiti, J. (2024). A critical review on system architecture, techniques, trends and challenges in intelligent predictive maintenance. *Safety Science*, 177:106590.
- Madasamy, S., Shankar, B. P., Yadav, R. K., and P. J. K. (2023). A Machine Learning Approach in Predictive Maintenance in the IoT Enabled Industry 4.0. In *2023 4th International Conference on Smart Electronics and Communication (ICOSEC)*, pages 418–423, Trichy, India. IEEE.

- Mienye, I. D. and Sun, Y. (2022). A Survey of Ensemble Learning: Concepts, Algorithms, Applications, and Prospects. *IEEE Access*, 10:99129–99149.
- Moat, G. and Coleman, S. (2021). Survival Analysis and Predictive Maintenance Models for non-sensored Assets in Facilities Management. In *2021 IEEE International Conference on Big Data (Big Data)*, pages 4026–4034, Orlando, FL, USA. IEEE.
- Pölsterl, S. (2024). *scikit-survival: A Python package for time-to-event analysis*. Accessed: 2024-07-18.
- Ramezani, S. B., Cummins, L., Killen, B., Carley, R., Amiratifi, A., Rahimi, S., Seale, M., and Bian, L. (2023). Scalability, Explainability and Performance of Data-Driven Algorithms in Predicting the Remaining Useful Life: A Comprehensive Review. *IEEE Access*, 11:41741–41769.
- Rossini, R., Prato, G., Conzon, D., Pastrone, C., Pereira, E., Reis, J., Goncalves, G., Henriques, D., Santiago, A. R., and Ferreira, A. (2021). AI environment for predictive maintenance in a manufacturing scenario. In *2021 26th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, pages 1–8, Vasteras, Sweden. IEEE.
- Shen, J., Yang, D., Zhou, Y., Pei, J., Wu, Z., Wang, X., Zhao, K., and Ding, Y. (2024). Development of machine learning models for patients in the high intrahepatic cholangiocarcinoma incidence age group. *BMC Geriatrics*, 24(1):553.
- Tran, T. T., Lee, J., Gunathilake, M., Kim, J., Kim, S.-Y., Cho, H., and Kim, J. (2023). A comparison of machine learning models and Cox proportional hazards models regarding their ability to predict the risk of gastrointestinal cancer based on metabolic syndrome and its components. *Frontiers in Oncology*, 13:1049787.
- Yang, Z., Kannianen, J., Krogerus, T., and Emmert-Streib, F. (2022). Prognostic modeling of predictive maintenance with survival analysis for mobile work equipment. *Scientific Reports*, 12(1):8529.
- Zeng, C., Huang, J., Wang, H., Xie, J., and Zhang, Y. (2023). Deep Bayesian survival analysis of rail useful lifetime. *Engineering Structures*, 295:116822.
- Zhou, B., Pychynski, T., Reischl, M., Kharlamov, E., and Mikut, R. (2022). Machine learning with domain knowledge for predictive quality monitoring in resistance spot welding. *Journal of Intelligent Manufacturing*, 33(4):1139–1163.
- Zonta, T., da Costa, C., da Rosa Righi, R., de Lima, M., da Trindade, E., and Li, G. (2020). Predictive maintenance in the Industry 4.0: A systematic literature review. *Computers and Industrial Engineering*, 150.