Contents lists available at ScienceDirect

Reliability Engineering and System Safety

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

Semi-supervised clustering-based method for fault diagnosis and prognosis: A case study

Kamyar Azar [a], Zohreh Hajiakhondi-Meybodi [b], Farnoosh Naderkhani [a,*]

[a] *Concordia Institute for Information System Engineering, Concordia University, Montreal, Canada*
[b] *Department of Electrical and Computer Engineering, Concordia University, Montreal, Canada*

ARTICLE INFO

Keywords:
Condition-based maintenance
Machine learning
Maintenance decision support system
Prognostic and health management
Proportional hazard model

ABSTRACT

Recent increased enthusiasm towards Industrial Artificial Intelligence (IAI) coupled with advancements in smart sensor technologies have resulted in simultaneous incorporation of several Condition Monitoring (CM) technologies within manufacturing/industrial sectors. Smart utilization of CM data leads to enhanced safety, reliability and availability of manufacturing systems. Conventional system monitoring techniques, however, cannot efficiently cope with such rich CM information content. In this regard, the paper proposes a novel hybrid Maintenance Decision Support System (MDSS) for fault diagnostic and prognostics considering event-triggered CM data. The proposed MDSS is a hybrid framework designed by coupling Machine Learning (ML)-based models and statistical techniques. More specifically, the MDSS is a time-dependent Proportional Hazard Model (PHM) augmented with semi-supervised ML approaches and Reinforcement Learning (RL) to find an optimal maintenance policy for systems subject to stochastic degradations with focus on cost minimization. The developed hybrid model is capable of inferring and fusing high-volume CM data sources in an adaptive and autonomous fashion to recommend optimal maintenance decisions without human intervention, which is a step-forward contribution in the maintenance context. To evaluate the structure and performance of the proposed model, comprehensive ML-based solutions are developed based on a dataset provided by NASA containing run-to-failure and CM data associated with aircraft engines.

1. Introduction

In today's competitive business world, every company observes the urgency to work with full potential to keep up with competitors and stay, strongly, in the business. Given advancements and evolution of the Industrial Internet of Things (IIoT), and the advent of smart sensors technologies, Condition Monitoring (CM) data becomes abundantly accessible. Availability of rich and real-time CM data together with advancements in Machine Learning (ML), in particular, Deep Neural Networks (DNNs), have made it possible to intelligently and more efficiently monitor industrial systems and optimally perform maintenance decision making. Rise of IIoT and Industrial Artificial Intelligence (IAI) have encouraged industrial/manufacturing sectors to deploy intelligent and autonomous systems for production. As a result, CM data can be understood and processed better and more efficiently. It is expected that IIoT and IAI soon become integral component of every business seeking to become successful in the market, and keep up or advance against its competitors. Moreover, it is viable to maintain the integrity of complex systems via an efficient and smart Prognostic and Health Management (PHM) system. A smart PHM system is responsible to minimize the cost by reducing failures via monitoring the system continuously, detecting any distinct abnormalities or trends in CM data, and suggesting optimal or near-optimal control and maintenance actions.

Capitalizing on the aforementioned importance of IIoT and IAI, Condition-based Maintenance (CBM) can be considered as the enabling mechanism for companies to maintain a competitive advantage over their competitors. The CBM is considered as the state-of-the-art maintenance policy that uses CM data to track health state of the underlying system; predict potential failure point(s), and; prevent unwanted failure occurrences. Recently, there has been a new trend in using ML/DNN and Deep Learning (DL) based solutions for design of CBM strategies because of their unique capabilities in analyzing and processing High-Dimensional and Multi-model Streaming (HDMS) CM data [1]. Although using ML/DL techniques in predictive maintenance [2–4] is significantly challenging, it has great potential to introduce exceptional benefits such as superior availability and reliability of assets, capital expenditure decrease (fleet size can be decreased with improved asset availability), and operational expense reduction (e.g., replacement costs). Generally speaking, CBM consists of three main components: (i)

[*] Correspondence to: 1515 Saint-Catherine St. W., Montreal, QC, H3G-2W1, Canada.
E-mail addresses: k_zar@encs.concordia.ca (K. Azar), farnoosh.naderkhani@concordia.ca (F. Naderkhani).

https://doi.org/10.1016/j.ress.2022.108405
Received 29 October 2020; Received in revised form 14 February 2022; Accepted 15 February 2022
Available online 25 February 2022

Data acquisition; (ii) Data processing, and; (iii) Maintenance decision-making [5]. Although, the first two components have recently been researched extensively, extracted information rarely translated and efficiently/properly used for maintenance decision making. The paper aims to address this gap by proposing a novel MDSS that uses a semi-supervised clustering-based method for fault diagnosis and prognosis and RL to find the optimal policy.

2. Related work

In this section, we present a literature review on the ML and DL applications in CBM and the maintenance decision-making models alongside the limitations of the existing models. The novelty and contributions of the proposed model are also provided followed by organization of the paper.

2.1. Literature review

Deep learning-based Remaining Useful Life (RUL) prediction has attracted significant attention recently. Using ML/DL models to predict RUL is one of the recent contributions in this line of research. For instance, a deep bidirectional Long Short-Term Memory (BiLSTM) model is used to estimate RUL for gas turbine engines based on a public dataset provided by NASA [6] in [7]. The author shows promising results in comparison with other models like Multi-Layer Perceptron (MLP) and Support Vector Regression (SVR). Another interesting work is presented in [8] where the authors proposed a dual-task deep Long Short-Term Memory (LSTM) model for joint learning of degradation stage assessment and RUL prediction for achieving more robust and accurate result. Authors in [9] proposed a new DL method based on Convolution Neural Networks (CNN) to estimate the RUL. After comparison with other methods, they show the effectiveness and excellence of their model. Authors in [10], developed a data-driven approach based on Similarity-based Interpolation (SBI) to estimate the RUL. In their method, the high-dimensional sensors' measurement is first transformed into a one-dimensional Health Index (HI) that shows the degradation and the health of the system. After that, RUL is estimated using the SBI method. A bidirectional RNN autoencoder-based model is constructed to compute the HI values more robustly in comparison to the RNN autoencoder model that was traditionally used. Furthermore, a domain-specific rule is proposed to more accurately predict the RUL. In addition to proposing a novel one-dimensional HI, Shi and Chehade [11], developed a novel Dual-LSTM network that could detect the change point, which is a point in time where the underlying system starts to deteriorate. The authors discussed that detecting the change point is beneficial since it is not reasonable to predict RUL before that, and the data before this change point does not provide potentially useful information. A review of the current literature and status of this line of research, along with the future opportunities can be found in [12], and more references can be found in [13–18]. Although the results are promising and highly accurate, it is believed that RUL estimation alone without design of a decision support system is not a sufficient measure for maintenance decision-making. Furthermore, while RUL can provide useful information, it may not be an adequate measure to make decisions solely based on its value. That is because, in the literature, a wide range of papers deal with estimating RUL without considering that in some systems, mostly where a failure can be catastrophic, it is preferable to predict the probability that a machine or a system operates without failure until the next inspection interval. In such systems, relying only on how much time is left before observing a failure (i.e., the RUL) is not enough for maintenance operators to decide whether the inspection interval is suitable or not. On the other hand, "Hazard Rate" is a sufficient measure and it can address the mentioned problem. Moreover, the RUL can be calculated from the hazard rate, but not vice versa.

In the literature, there is plenty of existing research on CBM including both statistical and ML-based approaches. Conventional statistical CBM models such as Markov Decision Process (MDP) have been extensively used within the maintenance optimization context to develop optimal maintenance policies. A Proportional Hazard Model (PHM), referred to as the Cox's PHM, is a frequently used model in survival analysis and can be used to formulate and estimate the hazard rate. Such a model considers both the age of the system, or baseline hazard, and the system's degradation, or hazard risk. In the conventional Cox's PHM, the covariates, that associate the hazard rate with the degradation of the system, are assumed to be consistent over time. Despite that, a time-varying Cox's PHM consists of covariates that can change over time. The parameters of such a model can be estimated with a statistical approach and an optimal maintenance policy can then be achieved. One of the early works in this context is presented by [19,20], where the optimal replacement policy is derived in a Semi-Markov Decision Process (SMDP) framework. In this work, the Cox's PHM is considered with a right continuous stochastic covariate process under periodic maintenance. Finally, to minimize the system's long-run expected average cost, a recursive algorithm was developed. In [21], the previous discrete-time approximation assumption was relaxed, and optimal policy for a deteriorating system using PHM was proposed. An autoregressive model with the time effect is considered for the first time in [22] to formulate the system's degradation. Next, the Mean Residual Life (MRL) is derived based on this autoregressive model. Finally, they determined an optimal maintenance policy under different inspection strategies to minimize the long-run cost of the system by formulating the decision problem in the SMDP framework. The authors in [23] designed a two-level cost-optimal Bayesian control chart based on the Hidden Markov Model (HMM) for early failure detection of mechanical equipment. In their work, dependent failure modes are considered, which are referred to as degradation and catastrophic failures. The difference between these two failure modes is that catastrophic failure can happen any time when the system is under operation, but the degradation failure can happen only after the sojourn time in the unhealthy state of the system. In this regard, to model this failure dependency, it is assumed that in the healthy state, the joint distribution of the time to the catastrophic failure and sojourn time follows Marshall–Olking bivariate exponential distribution. The optimization problem for the developed control chart is solved in the SMDP framework, and MRL is calculated using the Bayesian approach. To evaluate their model, they used a real multivariate dataset from a milling machine and compared their model with a single sampling interval and a single control limit. In [24], a CBM policy with a dynamic threshold (instead of a constant one) is proposed for a system subject to minor or catastrophic failures. They assumed multiple actions, namely no maintenance, imperfect maintenance, and preventive replacement, in each inspection epoch and developed a modified policy-iteration algorithm within the SMDP framework to solve their optimization problem. Finally, a complete literature review of CBM models with emphasis on mathematical modeling and optimization approaches is provided in [25]. An optimal maintenance policy for defense application is presented in [26], which is another application of CBM methodology within the PHM framework. More references can be found in [27–31].

Conventionally, policies and decision support systems developed based on Cox's PHM are statistical in nature. However, given availability of extensive CM and HDMS sensory data, statistical approaches are proficient for most of today's problems. That is because statistical approaches can be used on problems with a few input data and features. However, when the size of input data and the number of features increases, it is hard for such approaches to handle the relationships and correlations inherent in the data. Specifically, statistical solutions for Cox's PHM models pose serious analytical challenges to deal with high dimensionality and variety of the data. Moreover, most of the statistical approaches are time-consuming as the number of inputs increase. Finally, most statistical approaches, typically, rely on some unrealistic pre-assumptions rendering their application for real-world problems impractical. As stated previously, the complexity of modern manufacturing systems and exponential growth of CM data in different modalities, make such conventional models impractical. This has resulted in the emergence of promising research ideas in the design and application of state-of-the-art AI/ML algorithms that contribute to the advancement of maintenance decision making. One of the most widely used ML-based solutions for CBM optimization is known as Reinforcement Learning (RL) in which a learner, or an agent, learns what actions to take in each state of the system from its interactions with the environment. As a recent example, we can refer to Ref. [32] where the authors proposed an RL-based solution to find the optimal policy for long-term pavement maintenance planning. The authors discussed that since the state–action space is large, using conventional methods such as dynamic programming is time-consuming and inefficient. Therefore, first, they have applied Principle Component Analysis (PCA) to reduce the state–space dimension, and then, an Artificial Neural Network (ANN) is developed to approximate the Q-value function. After tuning hyper-parameter values, the optimal maintenance policy is obtained including different types of treatments for a 15-year maintenance planning horizon. Similarly, the authors in Ref. [33] applied a Deep RL (DRL) model to obtain the optimal maintenance policy with the application to bridge maintenance, where the number of components is large. Due to the infinite nature of the state–action space, the problem cannot be solved by dynamic programming. Therefore, the authors used a Convolutional Neural Network (CNN), as a non-linear approximator, to learn the state–action Q-value. They discussed that the network could learn using either a large historical data source or simulation data. Moreover, the transition probabilities are obtained via both simulations and based on historical data. In Ref. [34], authors proposed an RL-based and non-tabular solution for the problem of optimal Operation and Maintenance (O&M) scheduling of power grids. They combined Q-learning with an ensemble of ANN to be applicable for large systems with high dimensional state–action spaces. For more information in this line of research, please refer to [35–39]. In this context, since the state–action space is exponentially growing, either dimension reduction techniques (such as PCA) are utilized or DRL for CBM will be used, which do not need dimension reduction. In particular, DRL incorporates deep learning to efficiently solve high-dimensional MDPs by representing the policy or other learned functions as a neural network. In this paper, an alternative approach is proposed to utilize the hazard rate for maintenance decision-making, which only depends on two factors, namely baseline hazard, and hazard risk. All available CM data are incorporated by the proposed model to build an optimal MDSS. Consequently, the proposed model does not require dimension reduction techniques and would not suffer from the curse of dimensionality.

2.2. Novelty and contribution

In this paper, we propose a semi-supervised clustering-based framework for maintenance decision making based on Cox's PHM model. More specifically, we propose an efficient and novel hybrid semi-supervised model for fault diagnostic and prognostics considering CM data, which combines both statistical approaches and ML algorithms. In the proposed model, we first compute two different components of a time-dependent Cox's PHM model, namely, Baseline Hazard (BH) function, which considers the age of the system, and Hazard Risk (HR) function, which takes the CM data into account. Then, due to unavailability of labeled data, a semi-supervised method is applied to estimate the state of the system based on its age and sensors' measurements. Consequently, all CM data are used to build the clusters, after which the state–space with its associated transition probability are defined based on the aggregated clustering. The results obtained from the introduced aggregated clustering method are then fed to the RL/DRL module to close the loop and minimize the total cost. The proposed framework provides superior performance from application point of view without relying on pre-knowledge and pre-assumption about the data. Furthermore, the proposed framework has reduced computational complexity and reduced sensitivity to dimensionality, variety and modality of the data. The proposed model can be further considered as a basis to develop a smart MDSS that can monitor the system via sensors, analyze the state of the system, and make optimal maintenance decisions without any human intervention. In brief, contributions of the paper can be summarized as follows

- Providing a semi-supervised clustering and RL/DRL-based framework for efficient interpretation and proper use of CM data along with the age of the system for fault diagnostic and prognostic.
- Using hazard function to design an alternative maintenance model, which is a more generalizable and sufficient measure compared to the RUL.
- Developing an innovative methodology to estimate the state of the system without domain-specific knowledge or time-consuming and cost-inefficient assumptions.

2.3. Overview

The rest of the paper is organized as follows: Section 3 deals with the problem statement. Section 4 describes the dataset used in development of the proposed decision support system and provides the required background. Section 5 presents the proposed hybrid and semi-supervised ML-based MDSS framework. Finally, Section 6 concludes the paper.

3. Problem statement

As we discussed earlier, CBM is a state-of-the-art maintenance policy that uses CM data to track the health condition of the system, predict the equipment failure, and prevent the occurrence of costly failures. We consider a machine or a system, which is subject to stochastic degradation and random failures that can happen at any time epoch. The system's health condition is monitored via sensory devices deployed in the system, which provide a large amount of sensor measurements and signals over time. The collected CM data should be processed and analyzed for proper maintenance decision making. CM date can have different formats such as value type, waveform type (such as vibration signals) and high-dimensional types (e.g., image sequences). In this paper, we propose a model based on ML-based solutions for diagnostic and prognostic of a system subject to failure, which takes raw CM data to make maintenance decisions based on the state of the system. We consider that the system can be in three distinguishable states, namely, healthy, warning, and failure state. Upon system reaching each state, an appropriate maintenance action should be taken. If the system is found to be in the healthy state, no action is required. On the other hand, if the system found to be in the warning state, preventive maintenance or minor repair will be performed. Finally, if the system enters the failure state, corrective maintenance will be carried out.

This completes the problem statement. Next, we present the dataset used to develop and evaluate the proposed MDSS framework. It is worth reiterating, as discussed in Section 1, that recently the main focus of DNN/ML modeling based on the utilized dataset was on RUL prediction. It is, however, strongly believed that RUL alone without design of a proper decision support system is not sufficient for maintenance decision-making. The paper aims to address this gap.

4. Dataset and background

In this section, first, we briefly introduce the dataset used to develop the proposed semi-supervised MDSS framework. Next, Cox's PHM model and ML models utilized to design the proposed MDSS framework are briefly presented.

Table 1
Summary of C-MAPSS dataset description.

Sensor ID	Mean	STD	Min	Max
SM1	518.67	0.00	518.67	518.67
SM2	642.68	0.50	641.21	644.53
SM3	1590.52	6.13	1571.04	1616.91
SM4	1408.93	9.00	1382.25	1441.49
SM5	14.62	0.00	14.62	14.62
SM6	21.61	0.00	21.60	21.61
SM7	553.37	0.89	549.85	556.06
SM8	2388.10	0.07	2387.90	2388.56
SM9	9065.24	22.08	9021.73	9244.59
SM10	1.30	0.00	1.30	1.30
SM11	47.54	0.27	46.85	48.53
SM12	521.41	0.74	518.69	523.38
SM13	2388.10	0.07	2387.88	2388.56
SM14	8143.75	19.08	8099.94	8293.72
SM15	8.44	0.04	8.32	8.58
SM16	0.03	0.00	0.03	0.03
SM17	393.21	1.55	388.00	400.00
SM18	2388.00	0.00	2388.00	2388.00
SM19	100.00	0.00	100.00	100.00
SM20	38.82	0.18	38.14	39.43
SM21	23.29	0.11	22.89	23.62

4.1. Dataset description and pre-processing

The Commercial Modular Aero-Propulsion System Simulation (C-MAPSS) dataset is one of the most widely used datasets in the prognostic and health monitoring context. The C-MAPSS dataset is provided by NASA and consists of run-to-failure data. This dataset includes four subsets of data known as FD001, FD002, FD003, and FD004 and each subset. In this paper, we have used the FD001, FD002, FD003, and FD004 datasets such that each dataset is split into train and test datasets. In total, 21 number of sensors are installed to each component of the system in order to collect CM data. In total, there are 100 engines under operations, which are subject to random failures. The data for each subset of the C-MAPSS dataset simulates several scenarios of the degradation behavior under three operational conditions. Table 1 shows descriptive analysis of the measurements obtained from the 21 sensors related to Engine 9. Pre-processing of data consists of two main steps: (i) Sensor selection, and; (ii) Normalization which are described below:

Sensor Selection: As it can be observed from Table 1, some of the sensors' measurements have constant values over time or low standard deviation. Therefore, these sensors are excluded from the model and only 14 sensors were selected, namely, SM2, SM3, SM4, SM7, SM8, SM9, SM11, SM12, SM13, SM14, SM15, SM17, SM20, and SM21.

Normalization: To achieve consistent results, typically, DL/ML models rely on a normalization mechanism before training when features have different ranges. Since the value range of different sensors' measurements is different, greater values might influence the model substantially. This may lead to a biased final result. To prevent this problem, it is recommended [40] to scale each feature into the same range of values, which in turn boosts the training speed. However, interpretation of results by using such normalization techniques might become harder, because the scale of the values is changed. Given the aforementioned remarkable advantages that come with normalization, normalized data is used for training purposes. Normalization is performed based on the following model:

$$X'_i := \frac{X_i - \min\{X_i\}}{\max\{X_i\} - \min\{X_i\}}, \tag{1}$$

where X and X' represent the ith sensor measurement before and after normalization. This completes our discussion on dataset description and pre-processing steps. Next, Cox's PHM model is briefly overviewed.

4.2. The Cox's PHM model

As stated previously, the hazard rate can be considered as a sufficient measure for maintenance decision making. Cox's PHM model is a popular model to estimate the hazard rate as follows:

$$h\big(t, Z(t)\big) = h_0(t) \times g\big(Z(t)\big), \tag{2}$$

where $h_0(t)$ represents the Baseline Hazard (BH) that shows the effect of the age on the system's failure, and function $g(\cdot)$ shows the Hazard Risk (HR) that represents the effect of covariates. The BH and HR can be modeled by a Weibull and an exponential distribution as follow:

$$h\big(t, Z(t)\big) = \frac{\beta}{\eta}\Big(\frac{t}{\eta}\Big)^{\beta-1} \times \exp\Big(\sum_i \lambda_i \times z_i(t)\Big), \tag{3}$$

where β and η are shape and scale parameters of the fitted Weibull distribution, $z_i(t)$ represents the value of the ith covariate at time t, and λ is the corresponding coefficient.

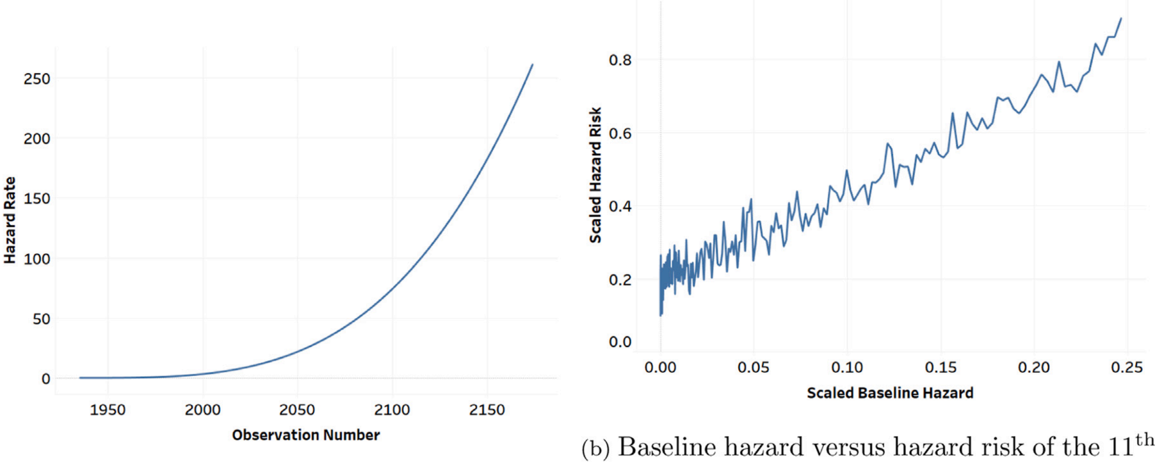
The first two parameters, namely scale and shape parameters, can be estimated by fitting a Weibull distribution on the failure data. The estimated parameters are as follows:

$$\hat{\beta} = 4.409, \quad \text{and} \quad \hat{\eta} = 225.026. \tag{4}$$

As β is greater than one, the hazard function is increasing, which indicates degradation of the system over time. For systems subject to wear out, Preventive Maintenance (PM) should be taken in order to bring the system to the healthy condition and minimize the risk of failure. In addition, if the cost of failure replacement is greater than the cost of PM action, which is almost the case in practical scenarios, using a predictive maintenance policy is justifiable. By using a predictive maintenance policy, the expected number of failures will reduce. Consequently, the expected cost of the failure replacement decreases, which results in the increase of the reliability and availability of the system over time.

We would like to clarify that although the value of β is high, performing preventive maintenance could still be an effective maintenance strategy. According to [41], it is important to note that preventive replacement actions, i.e., actions taken before equipment reaches a failed state, require two necessary conditions: (i) The total cost of the failure replacement must be greater than that of the preventive replacement, and; (ii) The hazard rate of the equipment must be increasing. In the CBM literature, there are several research works considering PM actions while the value of β is high. For example, in [42], and [43], the estimated β is 4.9 and [4.03–8.97], respectively. Authors in [44] used a method based on Weibull distribution and deep belief networks to assess the bearing's performance degradation. The estimated shape parameter in this work is considered as 5.067. Moreover, authors in [45] developed an opportunistic CBM strategy for offshore wind farms with four different components. Two out of four components have the estimated shape parameters equal to 3. Similarly, in [46] another method to obtain the CBM for offshore wind turbines is proposed and three out of four components in the system have the estimated values equal to 3. Finally, authors in [47] proposed a CBM policy for a system subject to soft and hard failures. The hard failure hazard rate is described by a PHM model with Weibull-distributed baseline hazard function. The estimated shape parameter is equal 4.63. The results obtained based on the proposed RL approach also confirms the above discussion.

The coefficients of the model are estimated using statistical approaches within the context of time-varying Cox's PHM framework using Maximum Likelihood Estimation (MLE). We would like to clarify that the parameter estimation is implemented based on the "lifelines", which is a complete survival analysis library written in Python [48]. In particular, for parameter estimation, we used the "CoxTimeVaryingFitter" function that fits Cox's time-varying proportional hazard model. The utilized function uses Breslow estimation [49] considering both effects of age and covariates. It is also worth mentioning that



(a) Hazard rate of the 11th engine

(b) Baseline hazard versus hazard risk of the 11th engine

Fig. 1. (a) Hazard rate corresponding to the 11th engine. (b) Baseline hazard (BH) versus Hazard Risk (HR) associated with the 11th engine.

Table 2
Estimated coefficients for C-MAPSS dataset.

Sensor ID	Estimated value λ_i for
SM2	1.424
SM3	0.021
SM4	0.145
SM7	−0.898
SM8	−1.698
SM9	−0.010
SM11	4.497
SM12	−1.647
SM13	6.201
SM14	0.026
SM15	3.394
SM17	0.245
SM20	−4.873
SM21	−3.720

supervised learning models are developed by extracting features to predict a certain label. In other words, the process of extracting and choosing the right set of features (the so called "Feature Engineering") is the most critical component in this context. In our work, choosing Baseline Hazard and Hazard Risk are part of our Feature Engineering scheme. More specifically, the Baseline Hazard and Hazard Risk are utilized as our features. The results are shown in Table 2. Finally, for illustration purposes, the hazard rate as well as the BH versus the HR associated with the 11th engine are shown in Fig. 1. As it can be seen, the hazard rate is increasing over time. The next step is to develop a maintenance model based on the calculated hazard rate using ML-based algorithms.

4.3. Semi-supervised machine learning models

As stated previously, ML is a sub-field of AI algorithms where computational statistics are used to allow machines and computers learn automatically from data without explicit programming. With the recent growth of the computation power as well as the abundance of CM data, it is now possible to utilize such a powerful tool for different problems. Generally speaking, ML models can be classified into supervised and unsupervised learning techniques. Because the C-MAPSS dataset does not have the state of the system, i.e., labels, supervised models cannot be readily used. We, therefore, utilize an unsupervised learning algorithm, which is a modified K-means algorithm to build the labels. After that, since the labels are qualitative, we train classification models to predict the state of the system based on the hazard rate. There are several popular algorithms for classification in the literature. In this paper, we have trained some of the most popular classification models. These models are briefly reviewed below:

1. **Traditional ML Models**: Conventional ML models consist of K-Nearest Neighbors (KNN) and Support Vector Machines (SVM), and Naïve Bayes, to name but a few. These models are easy to implement and are more interpretable and less complex than other types of classification models.
2. **Ensemble Learning Models**: Ensemble learning models are meta-algorithms that incorporate the results of heterogeneous or homogeneous base ML models to make more precise predictions and reduce the variance of the predictions. Such models consist of Max Voting, Stacking, and Random Forest. With regards to the first type of models, these models are more complex and harder to implement, but they can ensemble the result of different models and present more accurate models. The detailed descriptions of three ensemble models proposed in this paper are as follows:

   (a) ***Max Voting***: In this method, multiple models are applied to make predictions for each data point. The predictions by each model are considered as a vote. The final prediction is based on the predictions from the majority of the models.

   (b) ***Stacking***: Stacking is another ensemble model that combines different predictions from homogeneous or heterogeneous ML models. In this algorithm, there are two levels involved. In the first level, multiple models, known as base-models, are used to make predictions for each row of the data. Then, in the second layer, the previous predictions are used as inputs for a meta-learner model. In other words, a meta-learner uses predictions from base-models as a new feature to combine all of the base-models. In this paper, we have used logistic regression as the meta-learner. Logistic Regression predicts the probability of a certain data belonging to a class and uses the concept of a regression model to classify the data. Without any modification to the regression model, we cannot use it for classification problems. That is because the output of a regression model may fall outside of the [0, 1] interval. A way that we can assure the output of a regression model falls in the mentioned interval is by using a sigmoid function as follows:

$$p(X) = Pr(Y = 1|X) = \frac{\exp(B^T * X)}{1 + \exp(B^T * X)}; \tag{5}$$

where $Pr(Y = 1|X)$ means the probability of Y (response variable) belongs to class one if we observe X, and B is the model coefficients or parameters. This monotone transformation is also called the log-odds or logit transformation of $p(X)$. Cross-Entropy cost function is used instead of Mean Squared Error (MSE) to estimate the underlying parameters. This cost function has the following from:

$$C(\theta) = -\frac{1}{N}\sum_{i=1}^{N}(\log(h_\theta(x)) + \log(1 - h_\theta(x))). \tag{6}$$

By applying a decision threshold on the calculated probability, we can classify each data. For example, we can consider a 0.5 threshold, which means if the noted probability is lower than 0.5, the data belongs to the first class, and if it is more than 0.5, the data belongs to the other class.

(c) ***Random Forest***: Random forest is one of the ensemble learning methods that use classification or regression trees as basic learners. This algorithm makes up of many decision trees, and trains each one based on a random sampling of observations during building the trees over a random/limited subset of attributes when splitting the nodes. The final decision of a random forest is made by averaging the output of each tree for a regression problem, and by majority vote for a classification problem.

3. **Deep Learning Models**: More recent ML type of models are referred to as DL-based models, which consist of CNN, LSTM, and Feed Forward Neural Networks (FFNN), to name but a few. The rise of DL models is one of the latest trends in AI because of their powerful potential to extract useful information and high-level features from raw dataset. Hence, these models can be a good choice to assess the health of a system. Similar to the ensemble learning models, DL models are more accurate but they are less interpretable and need more time to train. Generally speaking, a DL model consists of blocks or units, called artificial neurons, and the network is trained by minimizing a loss function. In each epoch, a batch of data is fed forward and backward through the network, and at the end of each epoch, the parameters of the network are updated so that the loss function is minimized. This process is called forward and backward propagation, respectively. In this paper, LSTM network is also trained for MDSS. LSTM is a refined Recurrent Neural Network (RNN) model to deal with the gradient vanishing problem. LSTM includes a long-term cell state C_t to determine what information needs to be held through the time and what to be forgotten. The input of each LSTM cell consists of the data point in time t, X_t, and information from the previous cell, h_{t-1}. The information flow is controlled based on three gates or valves, i.e., forget gate, input gate, and output gate. The forget gate determines what information needs to be completely forgotten and what to be kept. The second gate decides what new information needs to be stored in the cell state. The last one decides what information needs to be transferred to the next cell as the current history.

5. Proposed hybrid and semi-supervised framework

In this section, we present the proposed hybrid and semi-supervised ML-based MDSS framework. First, in Section 5.1, we explain how to construct the label from extracted features, i.e., BH and HR, based on a semi-supervised learning method with an embedded K-means component. Second, after constructing the labels, we train supervised models to classify the points into the defined categories in the previous step. We trained different types of classification models in this paper, but they fall into three main categories: (i) Traditional Models, (ii) Ensemble Learning Models, and (iii) Deep Learning Models. This will thoroughly be discussed in Section 5.4.

5.1. Label construction stage

Classification models are supervised learning models, therefore, need outputs or labels. However, sensors' measurements in the C-MAPSS dataset do not have labels and we need to construct the labels (i.e., healthy, warning or failure) autonomously. In this section, we follow our intuitively pleasing procedure to extract and build such data from existing sensors' measurements. To do so, we can take advantage of clustering models. Clustering is an unsupervised learning model that clusters data into different groups. Points in each group are as much similar as possible, and as much different as possible with other groups. This can be achieved by minimizing the inertia or Within Cluster Sum of Square (WCSS) metric. There are several clustering algorithms in the literature. One of the popular and common algorithms in this context is K-mean clustering, which partitions data into the desired number of groups. In this paper, we utilized a three-step semi-supervised learning procedure and exploit K-means clustering in different steps to cluster data into three groups, namely healthy, warning, and most degraded or failure states. The aforementioned steps are discussed below:

1. *Step 1. Independent clustering of BH and HR*: In this step, we have used K-means clustering to independently group BH and HR data into two different clusters, namely healthy or unhealthy:

$$C_{\text{BH}} = \begin{cases} 1 & \text{BH is healthy} \\ 0 & \text{BH is unhealthy} \end{cases}$$

$$C_{\text{HR}} = \begin{cases} 1 & \text{HR is healthy} \\ 0 & \text{HR is unhealthy} \end{cases}$$

where C_{BH} and C_{HR} show the clustering label of BH and HR, respectively. The final independent clustering of BH and HR are shown in Figs. 2(a) and 2(b).

2. *Step 2. Aggregate Clustering*: In the Step 1, we partitioned BH and HR data into two different clusters independent of each other. In this step, we aggregate the clusters into one single cluster based on the following methodology:

   (1) If both BH and HR were clustered into the same group, i.e., both were clustered into the healthy or the unhealthy state, the final cluster will be healthy or most degraded/failure state, respectively.

   (2) If both BH and HR were clustered into different groups, i.e., BH was clustered into the healthy state and HR was clustered into the unhealthy state and vice versa, the final cluster will be the warning state. In summary, the combination of all possible scenarios can be written as:

$$C_{\text{Total}} = \begin{cases} 2 & C_{\text{BH}} = C_{\text{HR}} = 1 \\ 1 & C_{\text{BH}} \neq C_{\text{HR}} \\ 0 & C_{\text{BH}} = C_{\text{HR}} = 0 \end{cases}$$

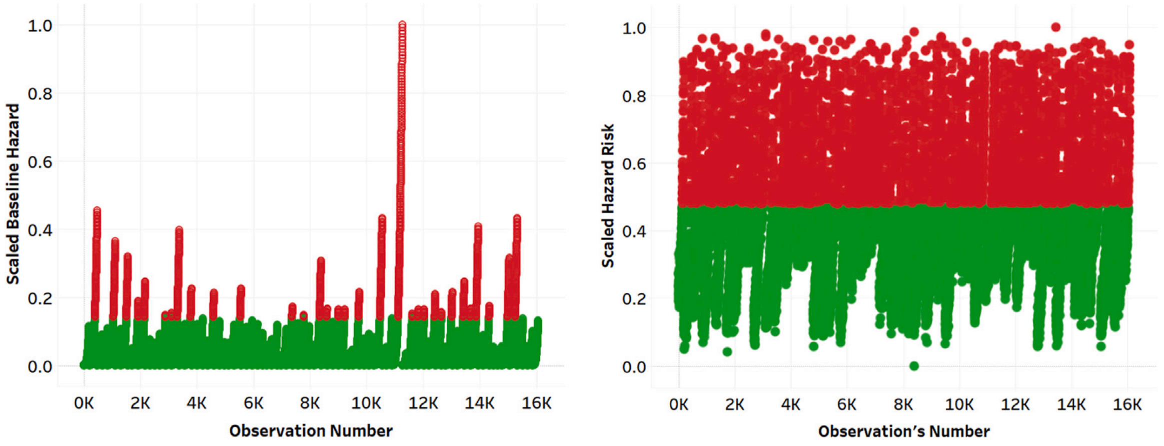
   where C_{Total} shows the final or aggregated cluster label.

3. *Step 3. Failure Clustering*: Since the failure time of engines is in a wide range (between 128 to 362), it is possible that for some engines that failed earlier, there would not be any most degrade or failure state. If we train the classification model without considering this problem, some potential earlier failures would not be detected. This problem can be handled by manually changing the last d labels of each engine into the most degraded or failure state, i.e.,

$$C_{\text{Total}} := \begin{cases} 0 & \text{The } d \text{ last run-to-failure data} \\ C_{\text{Total}} & \text{otherwise} \end{cases}$$

Finally, we can consider each cluster number as a label for the next step. The final result is shown in Fig. 3. This completes our discussion on the labeling stage.

As a final note to our discussion, we would like to clarify the rationale behind independent consideration of HR and BH. Generally



(a) BH clustering.

(b) HR clustering

Fig. 2. Individual Clustering.

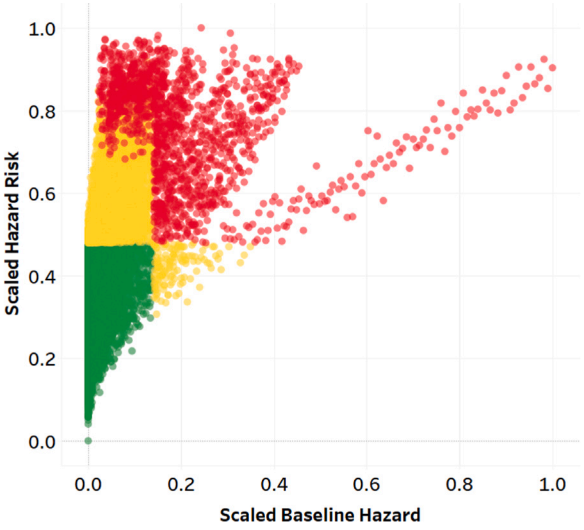


Fig. 3. Aggregated clustering.

speaking, the Cox proportional hazard regression model can be written as $h\big(t, Z(t)\big) = h_0(t) \times g(Z(t))$, where $g(Z(t)) = e^{\beta Z}$. The failure rate is considered to be the product of a baseline failure rate dependent only on the age of the unit and a positive function $g(Z(t))$ dependent on the values of the stochastic process Z. Thus, the predictors have a multiplicative or proportional effect on the predicted hazard [50]. It was shown that the optimal decision is to replace the item whenever the estimated hazard, calculated upon completion of the CM inspection, exceeds an optimal threshold. Following the above-mentioned mathematical points, Ref. [50] constructed the following decision strategy. Whenever an inspection is made, the values of the key risk factors are obtained (covariates). These measurements are multiplied by their weighting factors, and then combined to form the composite covariate value, which is marked on the Y-axis. The X-axis shows the age of the item at the time of inspection.

We have followed a similar intuition and constructed our decision graph based on BH and HR such that the BH is showing the working age of an engine on the X-axis and the effect of covariates are plotted across the Y-axis. The idea behind separating BH and HR is coming from the concept of "Proportional" hazards model, where a partial likelihood for β is proposed without involving the baseline function. Suppose we observe $(X_i; \delta_i; Z_i)$ for Engine i, where X_i is a possibly censored failure time random variable; δ_i is the failure/censoring indicator (1 = fail, and 0 = censor), and; Z_i represents a set of covariates. Let $R(t) = \{X_i \geq t\}$ denote the set of engines, which are at risk of failure at time t, called the risk set. Under the proportional hazards assumption, $h(t, Z_t) = h_0(t)e^{\beta Z}$, therefore, the partial likelihood is computed as follows

$$L(\beta) = \Pi_{j=1}^{k} \frac{e^{\beta} Z_j}{\sum_{l \in R(X_j)} e^{\beta Z_l}}. \tag{7}$$

As it can be observed from Eq. (7), the partial likelihood is independent of baseline and it uses the ranks of the failure times. As a final note, it is worth mentioning that in most maintenance decision-making problems, the true state of the system is hidden, however, the observations, i.e., condition monitoring data carry partial information about the system's state. In other words, by incorporating condition monitoring (sensory) data, the true state of the system becomes partially observable at each sampling epoch. Incorporation of the hazard risk, which itself depends on the instantaneous measurement values (hence conveying partial information about the true state of the system at the execution time), results in a non-static model. We came up with the idea of using an unsupervised learning method to discover the hidden pattern in the unlabeled data (which otherwise would have been left unused) and categorize the data into different groups. Therefore, by assuming three states in the data (healthy, warning, and failure states), it is logical to assume that their baseline hazard and hazard risk may have similar or close values for observations in a same state or cluster. The parameter estimation is a part of the feature engineering stage of our proposed framework. Every classical ML model needs some features, and feature engineering is a critical component of ML models. Even at times, input to Deep Learning models are engineered features.

5.2. Reinforcement learning-based cost minimization

We have implemented a novel RL approach based on the results obtained from the clustering method to close the loop and design a maintenance decision support system with focus on cost minimization. More specifically, the Q-learning model is developed as the optimization tool for maintenance decision making. Q-learning is a model-free and value-based RL algorithm to learn the value of $Q(s_t, a_t)$ for an action a_t in a given state s_t. The value of $Q(s_t, a_t)$ in each time slot is updated as follows:

$$Q(s_t, a_t) \leftarrow (1 - \lambda)Q(s_t, a_t) + \lambda\big(r_t + \gamma \max Q(s_{t+1}, a_{t+1})\big), \tag{8}$$

where γ and λ represent the discount rate and the learning rate, respectively. Term $r_t = \mathcal{R}(s_t, a_t)$ denotes the reward value associated with action a_t at state s_t. In this context, the developed RL model consists of following main components:

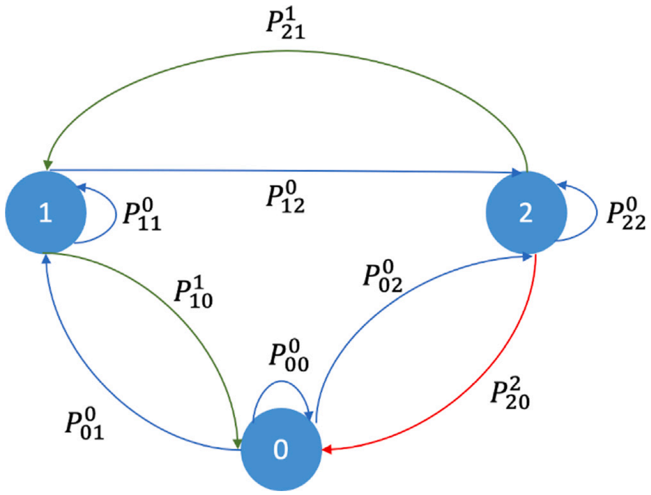


Fig. 4. Space-action space.

(i) **Agent:** *The management system, as the intelligent component of the engine*, acts as the agent and interacts with the environment based on a set of maintenance decisions.

(ii) **State–Space:** The state–space is associated with the *aggregated cluster label*, which consists of healthy ($s_t = 0$), warning ($s_t = 1$), and failure ($s_t = 2$) states.

(iii) **Action-Space:** Given the current state, the optimal action is taken by the agent. The action-space refers to the maintenance operation, discretized into three levels, i.e., no repair ($a_t = 0$), minor repair ($a_t = 1$), and major repair ($a_t = 2$).

(iv) **Reward:** Reward is defined as the maintenance cost. The main goal of the optimal policy within the RL model is to select the optimal maintenance actions minimizing the total cost or equivalently maximizing the total rewards for a long period of time. In this context, the reward function associated with each action is defined as follows:

$$\mathcal{R}(s_t, a_t) = \begin{cases} 0, & a_t = 0 \\ -20, & a_t = 1 \\ -100, & a_t = 2. \end{cases} \quad . \tag{9}$$

Due to the finite number of state and action spaces, we use Q-learning approach to update the state–action value function in each state, where the size of the Q-table is (3×3). Capitalizing on the importance of the exploration–exploitation trade-off, the ϵ-greedy action selection policy is considered in this context, where ϵ_t in episode t is calculated as follows:

$$\epsilon_t = \epsilon_{min} + (\epsilon_{max} - \epsilon_{min}) \exp(\epsilon_d t), \tag{10}$$

where ϵ_{min}, ϵ_{max}, and ϵ_d represent the minimum exploration rate, the maximum exploration rate, and the exploration decay rate, respectively. In this work, it is assumed that $\epsilon_{min} = 1/N_{total}$, $\epsilon_{max} = 1$, and $\epsilon_d = 0.1$, respectively. Term N_{total} is the total number of epochs. In the simulation study, $N_{total} = 200$ number of epochs are considered, where each epoch includes 400 iterations. Finally, the learning rate λ and discount rate γ are considered as 0.01 and 0.99, respectively. The proposed RL model determines the state–action space without incorporation of a data reduction technique such as PCA. More specifically, we have used the results obtained from the introduced aggregated clustering method and fed them to the RL model. In particular, all CM data is used to build the clusters, after which state–space with its associated transition probability are defined based on the aggregated clustering, which is finally fed to the RL model to minimize the total cost.

Table 3
Summary results of the RL approach.

Total cost	Total actions	Average of minor actions	Average of major actions
40.75	398.7125	0.5375	0.75

Fig. 4 shows the state–action space for the system under study, including three states 0, 1, and 2, and three actions 0, 1, and 2. In each time, the engine can transit from state i, for ($i \in \{0, 1, 2\}$) to state j, for ($j \in \{0, 1, 2\}$) with action a, ($a \in \{0, 1, 2\}$) with the probability $P^a_{i,j}$. As it can be seen from Fig. 4, blue arrows show the transition from state i to state j, when “no repair” action is selected. Similarly, green and red arrows represent “minor repair” and “major repair” actions, respectively. For instance, P^0_{12} represents the probability that an engine transits from state 1 to state 2 when no repair action ($a = 0$) is selected. The green arrows show the transition between two states when minor repair action is performed. We consider the realistic scenario that the effect of minor action is imperfect such that it returns the system to its previous state not the healthy state. Finally, when the system enters the absorbing failure state (state 2), major maintenance action will be performed, which brings the system to healthy state, denoted by red arrow in Fig. 4. The transition probability from state i to state j in a single unit time is calculated based on the clustering results as follows:

$$P_{i,j} = \frac{n_{i,j}}{\sum_{k=1}^{3} n_{i,k}}, \tag{11}$$

where $n_{i,j}$ is equal to the total number of observations in which the current state is equal to i, and the next state is j. Based on Eq. (11), the transition probability matrix for the system under study is obtained as follow:

$$\begin{bmatrix} 0.94533 & 0.05452 & 0.00015 \\ 0.13224 & 0.82936 & 0.03840 \\ 0.04433 & 0.04159 & 0.91408 \end{bmatrix} \tag{12}$$

By considering the above-mentioned transition probability formulation, and reward of -10 and 40 associated with the cost of minor and major repairs, respectively, the results of the RL implementation are shown in Table 3. The second column of Table 3 represents the total actions taken by the agent in each epoch. The last two columns are the average number of minor and major actions taken by the agent when the system reaches the steady state. According to Fig. 5, the steady state happens after the 75th epoch where the cost converges. Therefore, we calculated the average of the number of minor and major actions between 75 and 200 epochs.

5.3. DRL-based cost minimization

Here, we consider that the state–space consists of the hazard risk (i.e., a continuous state–space) and its corresponding age of the engine, where the total number of engines is equal to 100. In such a scenario where the state values are continuous, it is not possible to directly use a Q-table for updating the Q values. Consequently, deep learning models are applied to approximate the state–action value function. As shown in Fig. 6, we apply a Multi-Layer Perceptron (MLP) on the Q-Network to estimate Q-values, where the action-space and the reward function are exactly the same as the previous case study. After selecting a maintenance action with the highest value of $Q(s_t, a_t)$, the information associated with the corresponding action is stored in the experience replay, denoted by $D = \{\mathbf{e}_1, \ldots, \mathbf{e}_t\}$, where $\mathbf{e}_t = (\boldsymbol{\phi}_t, a_t, r_t, \boldsymbol{\phi}_{t+1})$. Term $\boldsymbol{\phi}_t = [s_{t-\beta}, a_{t-\beta}, \ldots, a_{t-1}, s_t]$ is the observed state sequence, including state–action pairs between time slots $t - \beta$ and t. The state sequence in replay buffer $\mathbf{e}_m$ is randomly selected to update the weight parameter ξ_t of the learning model based on the Stochastic Gradient Descent (SGD) method. Given ξ_t, the loss function $\mathbb{L}(\xi_t)$ which is the mean-squared error of the optimal and the selected Q-values, should be minimized, where $\mathbb{L}(\xi_t)$ is given by

$$\mathbb{L}(\xi_t) = \mathbb{E}_{\boldsymbol{\phi}_t, a_t, r_t, \boldsymbol{\phi}_{t+1}} \left[\left(Q_T - Q(\boldsymbol{\phi}_t, a_t | \xi_{t-1}) \right)^2 \right], \tag{13}$$

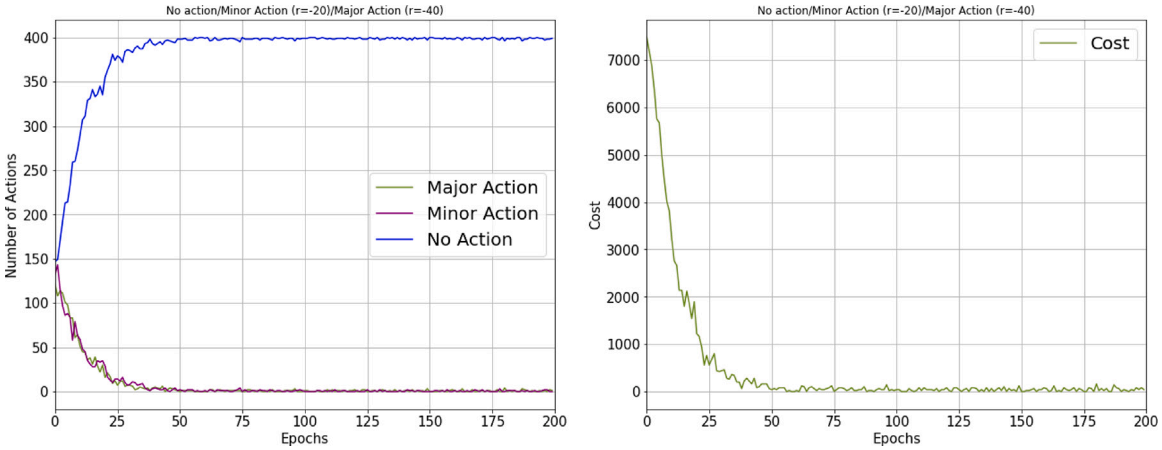


Fig. 5. (a) Steady-state action plot. (b) Steady-state cost plot.

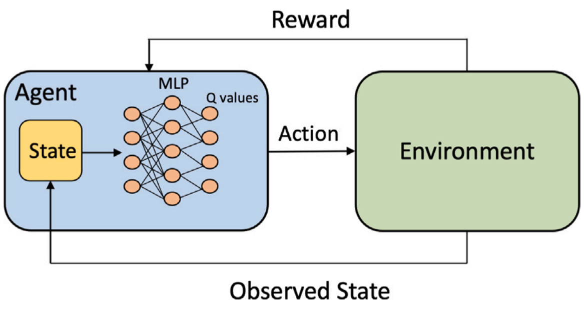


Fig. 6. The block diagram of the DRL model.

where Q_T as the target optimal Q-value is expressed as

$$Q_T = r_t + \gamma \max_{a_t'} Q(\boldsymbol{\phi}_{t+1}, a_t' | \xi_{t-1}). \tag{14}$$

Given the state s_t, the best action a_t^* is selected based on the ϵ-greedy algorithm with the probability of $(1 - \epsilon)$ as follows

$$a_t^* = \arg\max_{a_t'} Q(\boldsymbol{\phi}_t, a_t'). \tag{15}$$

Finally, the reward r_t associated with the corresponding action is calculated and the new experience $\{\boldsymbol{\phi}_t, a_t, r_t, \boldsymbol{\phi}_{t+1}\}$ is stored in the replay memory.

The DRL-based maintenance architecture utilizes a MLP model with one hidden layer with 512 nodes and sigmoid as the activation function. The size of the input and the output layers depend on the state and the action spaces, which are 2 and 3, respectively, where the activation function of the output layer is softmax. The DRL-based maintenance model is performed in 200 epochs. Considering that major maintenance brings the system to the healthy state, we assume that each epoch is terminated if a major repair $a = 2$ is selected by the agent. Fig. 7 illustrates the cumulative reward and the maintenance cost of the DRL-based maintenance architecture, respectively. As it can be seen from Fig. 7(a), increasing the number of epochs increases the cumulative reward, which shows that the model is well trained. As shown in Fig. 7(b), although the maintenance cost converges using the DRL model, it has the following drawbacks: (i) The DRL model has higher time complexity in comparison to the RL-based maintenance model. The main reason is that the DRL architecture uses a neural network to approximate the Q-value from the given action and state instead of building a Q-value table. Considering the fact that the continuous hazard risk is used as the state in this model, it takes more time to learn the best action for a given state; (ii) Based on our simulation results, the convergence of the DRL model depends on the definition of the minor repair. For instance, if the minor repair is defined so that the next state is $s_{t+1} = s_{t-5}$, the cost of DRL architecture could not converge to the steady-state, and; (iii) Although considering more layers in the DRL design potentially improves performance of the model, the maintenance cost cannot converge in an acceptable time.

5.4. Prediction model training

The main objective is to design and develop a hybrid MDSS framework for fault diagnostic/prognostic. The proposed MDSS in place should be able to construct optimal or near-optimal control and maintenance policies for decision-making based on CM data. Hence, we need to monitor the hazard rate of the system continuously to detect any abnormality in data that may indicate a change in the process or the system. As mentioned earlier, PM action is justifiable for a system with an increasing hazard rate. Performing proper PM actions at the right time will increase the system's availability, reliability, and efficiency.

In order to design such a smart system, we need to train a classification model since the outputs are categorical. In our case, a multi-class classification model should be used because we have three different classes, namely healthy, warning, and most degraded or failure class. Furthermore, after constructing the prediction model, each model has a decision graph in which we are interested. To prevent data leakage, first, data should be split into training and test sets. Therefore, in each run, we choose 80% of engines and considered their data as the training set, and the remaining ones as the test set. Then clustering should be fitted on the training set based on the procedure explained in Section 5.1. After that, we can transform training and test sets based on the cluster centers found in the previous step. After clustering and constructing the labels, we can train different classification models on the labels. As we discussed in Section 4.3, we divided the classification algorithm into three groups. The models and their results will be discussed more in depth in Sections 5.4.1, 5.4.2 and 5.4.3.

Besides, the same model can produce different results in each run due to the uncertainty involved in the process. Consequently, we exerted 10-fold Cross-Validation to get relevant results, the measure of bias and variance of the predictions, and to be able to compare different models. The results are shown in Table 4. In the following, we will go more into detail and discuss the results.

5.4.1. Results of traditional models

As mentioned previously, these models consist of some well-known and frequently used algorithms in the ML area, including KNN and SVM. Both models have different hyperparameters. To get the best results and compare their decision graphs, we trained each model with different hyperparameters. Figs. 8, 9 and 10 show the decision graph of KNN, SVM, and Naïve Bayes, respectively, with different

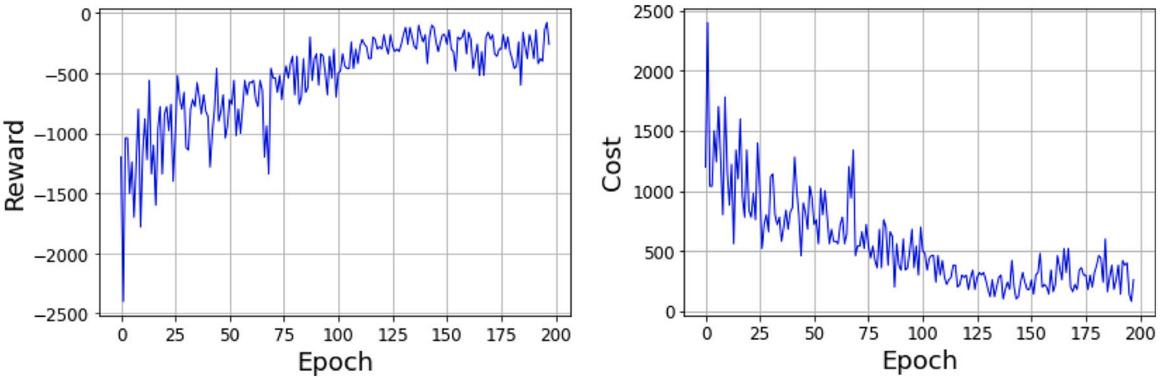


Fig. 7. (a) Variation of cumulative rewards versus different epochs. (b) Steady-state cost plot.

Table 4
Train accuracy average, standard deviation (SD), and test accuracy.

Model	Average of train accuracy	SD of train accuracy	Test accuracy
Random forest with 15 trees in the forest	0.9856	0.0044	0.9826
Stacking classifier	0.9843	0.0049	0.9805
KNN with 3 neighbors	0.9819	0.0049	0.9782
KNN with 1 neighbors	0.9793	0.0052	0.9771
KNN with 5 neighbors	0.9825	0.0036	0.9768
KNN with 4 neighbors	0.9812	0.0034	0.9752
KNN with 2 neighbors	0.9764	0.0043	0.97
SVM with 7-degree polynomial kernel	0.9695	0.018	0.9606
Two-hidden-layer ANN with 25 batch size and 100 epochs	0.9727	0.0102	0.9583
SVM with Radial Bases Function (RBF) kernel and scale coefficient	0.9728	0.0077	0.9569
SVM with 10-degree polynomial kernel	0.9703	0.0161	0.956
Max voting, Version 14	0.9724	0.0073	0.9551
Max voting, Version 19	0.9716	0.0069	0.9541
SVM with 6-degree polynomial kernel	0.9693	0.0049	0.9537
Max voting, Version 10	0.9732	0.0066	0.9528
SVM with 5-degree polynomial kernel	0.9682	0.0048	0.9505
SVM with 4-degree polynomial kernel	0.9689	0.0053	0.9479
SVM with 3-degree polynomial kernel	0.9684	0.0086	0.9434
SVM with 2-degree polynomial kernel	0.9607	0.0098	0.9349
Logistic Regression	0.9507	0.0089	0.9305
SVM with RBF kernel and auto coefficient	0.9525	0.0101	0.9291
SVM with sigmoid kernel and auto coefficient	0.9492	0.0091	0.9278
Gaussian Naive Bayes	0.9014	0.016	0.8847
Complement Naive Bayes	0.6483	0.0342	0.6654
SVM with sigmoid kernel and scale coefficient	0.4013	0.0728	0.3974

hyperparameters’ configuration. As it is clear, KNN with 3 neighbors, SVM with the 7-degree polynomial kernel, and Gaussian Naïve Bayes have the best results in each category.

5.4.2. Results of ensemble learning models

Ensemble learning is a hybrid learning system. As mentioned in the previous section, the basic idea behind ensemble learning is to train different models, known as base models or weak learners, and aggregate their predictions into one single output. By applying ensemble models, it is expected to enforce the prediction precision and obtain more accurate results. In this paper, we have developed three ensemble models including Random Forest, Max Voting, and Stacking which the results are presented as follows:

Random forest results. We trained Random Forest with a number of trees in the range of [1, 50]. Figs. 11(a) and 11(b) show 10-fold cross-validated accuracy score and standard deviation for the different number of trees in the forest, respectively. The best results are obtained by using 15 trees. The final Decision graph is shown in Fig. 11(c).

Max voting results. Here, different classification models are trained on the same training data, and then, these models are used to make predictions for each data point. Each prediction will be considered as a vote, and finally, the data point will be classified into a class that has the majority of the votes. In this paper, we considered three different scenarios:

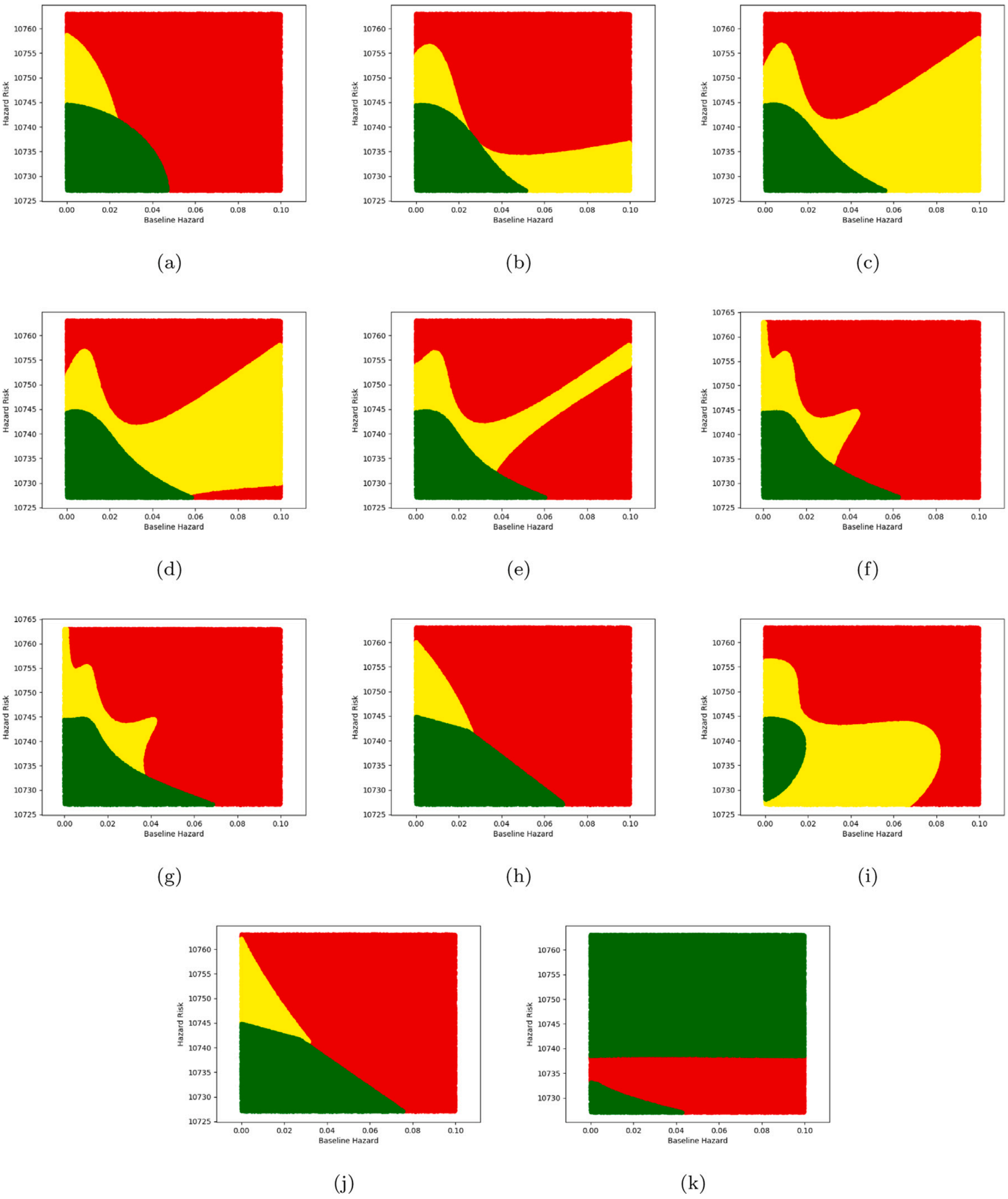
1. **Scenario 1**: In this scenario, we have considered all the traditional models in Table 4 and Random Forest.
2. **Scenario 2**: In this scenario, we have considered Random Forest, KNN with 1, 3, and 5 neighbors, Logistic Regression, SVM with 2, 3, 4, 6, 7 degrees polynomial kernel, and SVM with RBF kernel with scale and auto coefficients, SVM with sigmoid kernel and auto coefficient, and Gaussian Naive Bayes.
3. **Scenario 3**: In this scenario, we have considered Random Forest, KNN with 5 neighbors, Logistic Regression, SVM with 2, 3, 4, 5, 6, 7, 10 degrees polynomial kernel.

The decision graph of each scenario is shown in Fig. 12.

Stacking results. As mentioned in Section 4.3, stacking for classification problems, like Max Voting, considers heterogeneous base models. Each model is used to predict each data point, and these predictions are used as inputs for a meta-model or meta learner. In this paper, we considered KNN with 3 neighbors, SVM with the 7-degree polynomial kernel, and Random Forest with 15 trees as base models (best models based on obtained results) and Logistic Regression as the meta-model. The decision graph is shown in Fig. 13.

5.4.3. Results of deep learning models

Two deep learning models, namely, ANN and LSTM, are developed. ANN model with 25 batch size and 100 epochs is shown in Fig. 14. The



(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

(j) (k)

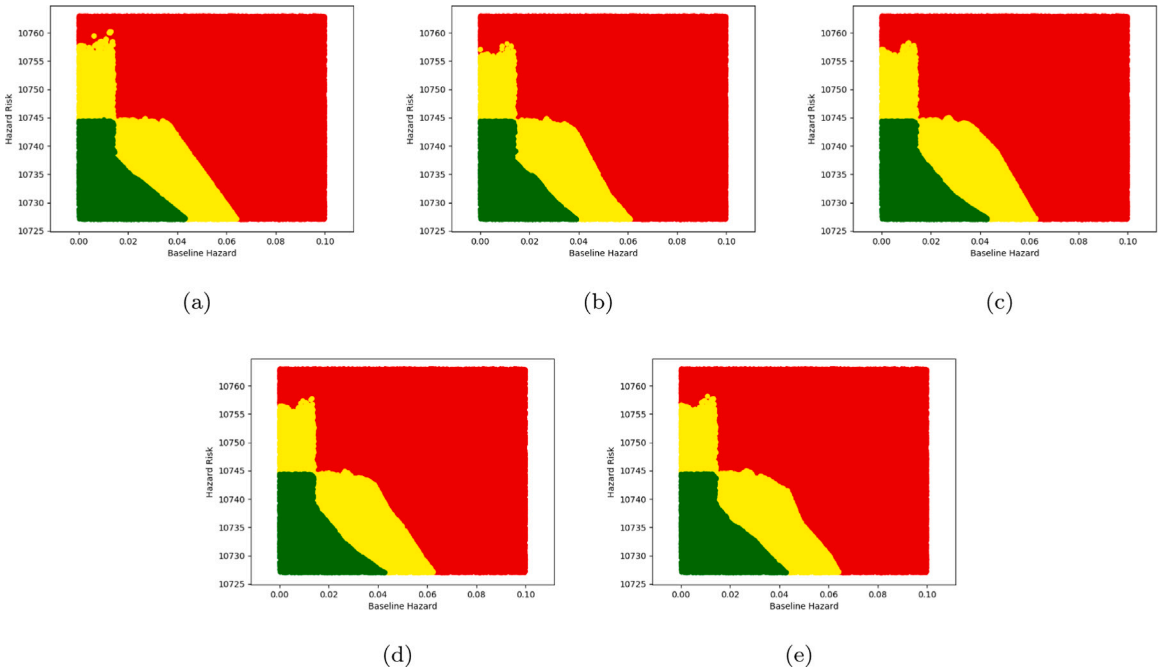
Fig. 8. SVM decision boundaries with following hyperparameter configurations: 8(a) 2-degree polynomial kernel, 8(b) 3-degree polynomial kernel, 8(c) 4-degree polynomial kernel, 8(d) 5-degree polynomial kernel, 8(e) 6-degree polynomial kernel, 8(f) 7-degree polynomial kernel, 8(g) 10-degree polynomial kernel, 8(h) RBF kernel and auto gamma parameter, 8(i) RBF kernel and scale gamma parameter, 8(j) Sigmoid kernel and auto gamma parameter, 8(k) Sigmoid kernel and scale gamma parameter.

configuration of this network is composed of two hidden layers with 6 nodes in each layer, and hyperbolic tangent as the activation function of such layers. LSTM is a refined model of ANN, which can be used to analyze sequence data, like time series. Since run-to-failure data can be considered as a time series, it is beneficial to use LSTM in our case (see Table 5). We trained three LSTM models with different structure as follows:

We cannot plot LSTM's decision graphs like other plots since such models are dependent on time and previous sensor measurements value. Therefore, we content ourselves with plotting the performance of LSTM models in the existing dataset. The results are shown in Fig. 15. It is worth mentioning that all these models are developed based on FD001 dataset. Similar analysis have been performed considering the other three datasets, i.e., FD002, FD003, and FD004, which are omitted here to save the space. The results are available upon request.

5.4.4. *Performance evaluations under different maintenance costs and policies*

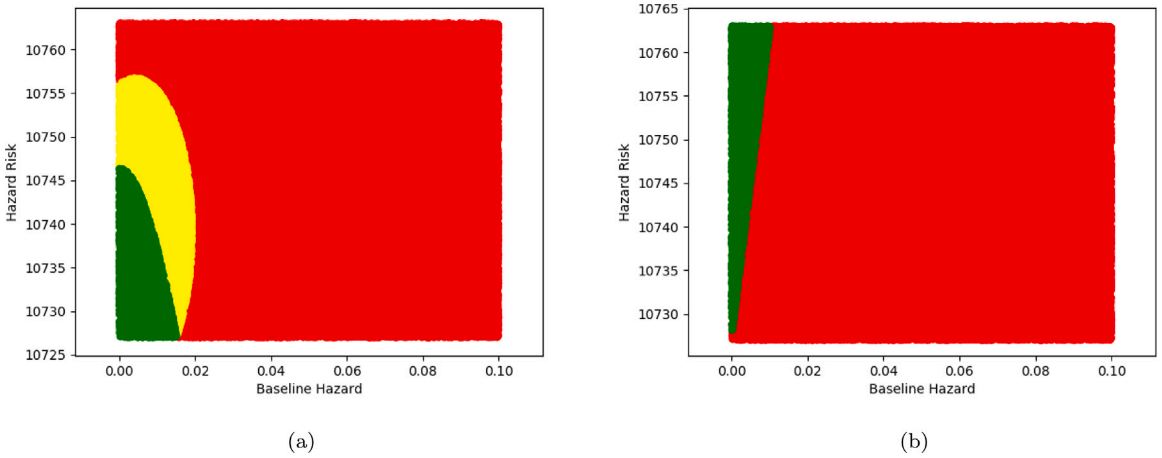
We would like to point out that unavailability of true labels is the case in several real applications. In such application domains, it is common, to construct labels based on operators and expert knowledge. As an example, we can refer to the pioneering research work of Ref. [51], which develops a replacement maintenance policy based on PHM using



(a) (b) (c)

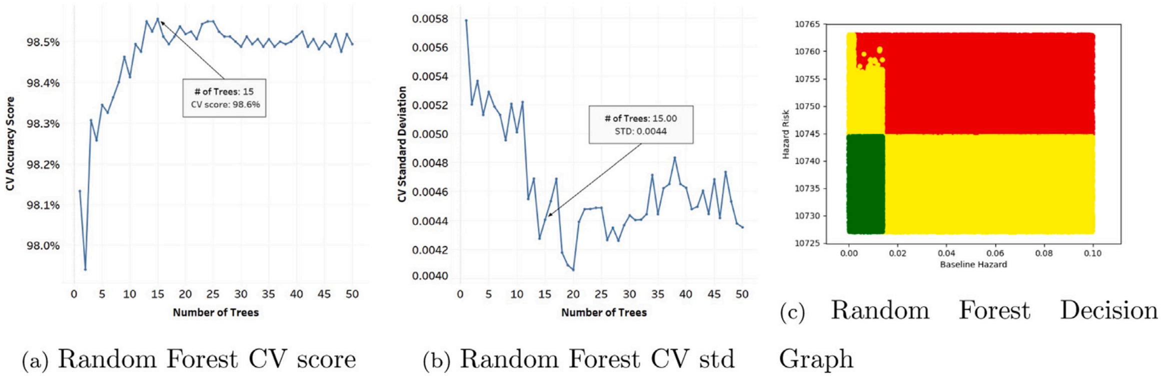
(d) (e)

Fig. 9. KNN decision boundaries with the following hyperparameter configurations: 9(a) one neighbor, 9(b) two neighbor, 9(c) three neighbor, 9(d) four neighbor, 9(e) five neighbor.



(a) (b)

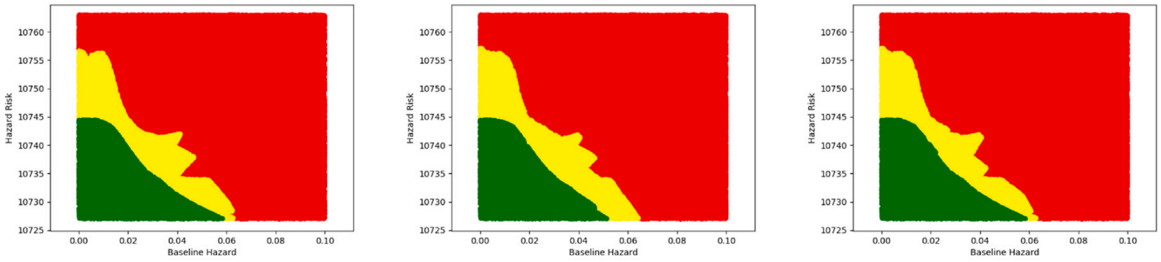
Fig. 10. Naïve Bayes with different algorithms: 10(a) gaussian Naïve Bayes, 10(b) complement Naïve Bayes.



(a) Random Forest CV score (b) Random Forest CV std (c) Random Forest Decision Graph

Fig. 11. Random Forest training history and decision graph.

the real dataset from a fleet of heavy hauler truck transmission's spectrometric oil analysis. In order to calculate the required cost function, the stochastic behavior of metal particles (covariates), which are represented by ppm of wear metal particles in engine oil, should be defined.



(a) Scenario 3 (b) Scenario 2 (c) Scenario 1

Fig. 12. Max voting decision graphs considering three scenarios.

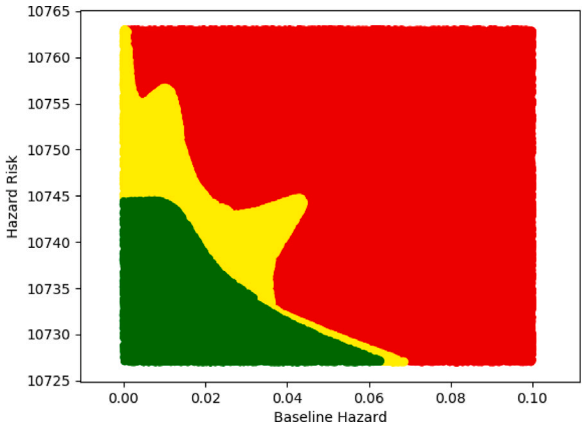


Fig. 13. Optimizing the CBM decision based on Stacking.

Table 5
LSTM models' structure.

	Number of hidden layers	Number of nodes in each hidden Layer	Hidden layer's activation function
First structure	5	64	relu
Second structure	5	64	tanh
Third structure	5	128	relu
Fourth structure	5	128	tanh
Fifth structure	3	128	relu
Sixth structure	3	128	tanh
Seventh structure	3	64	relu
Eighth structure	3	64	tanh

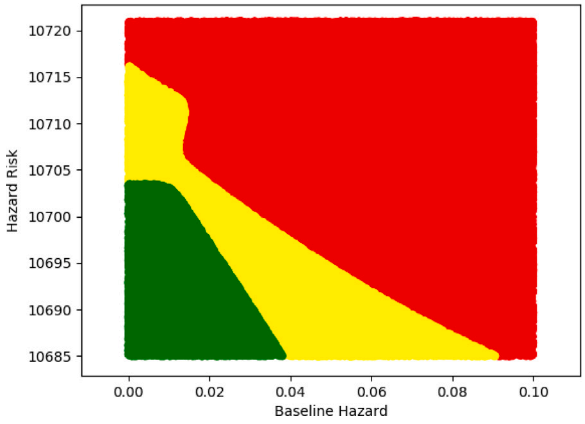


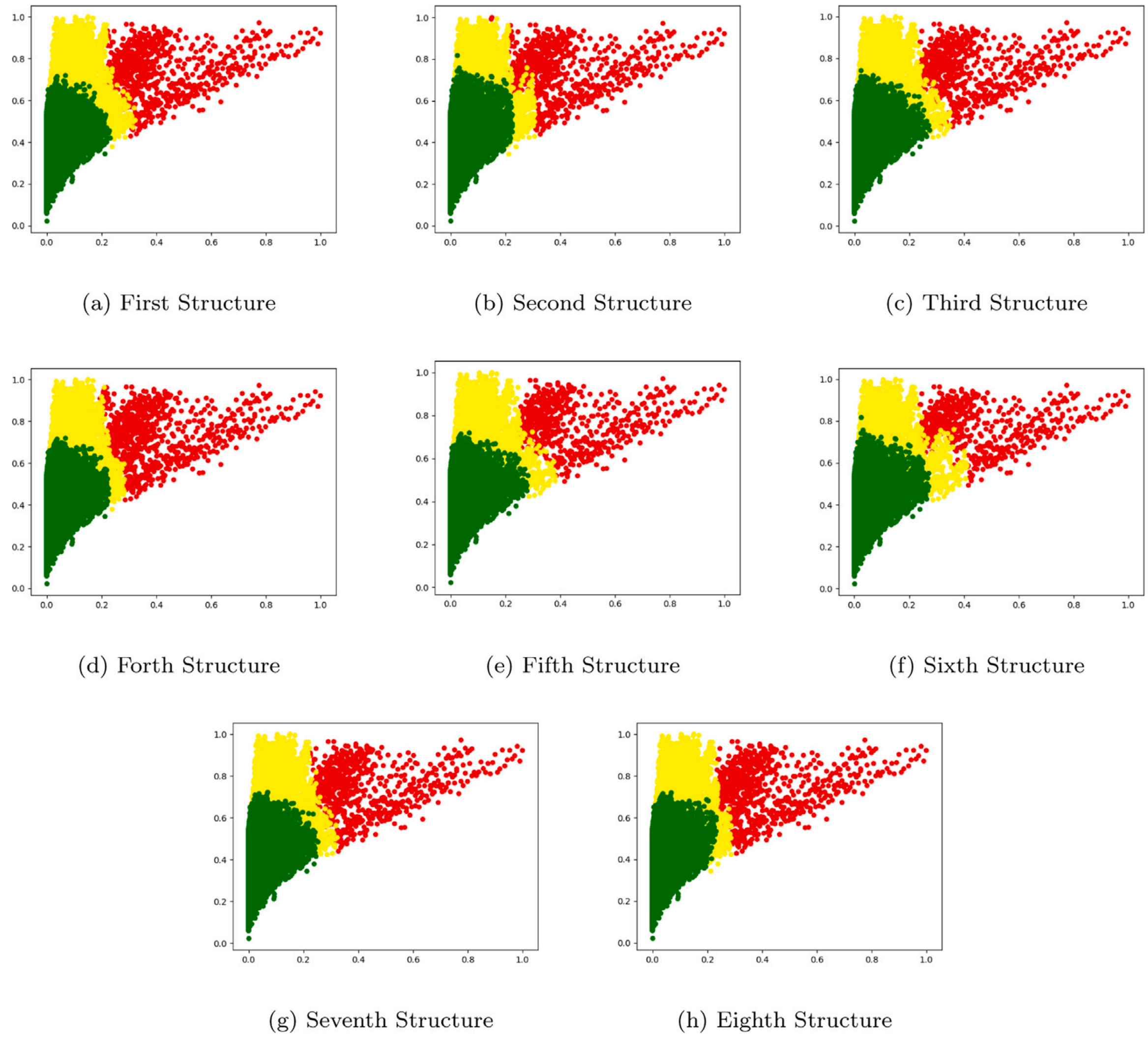
Fig. 14. Optimizing the CBM decision based on Feed Forward Neural Network.

As true labels are not available, authors assumed that covariates behave as a homogeneous Markov process and determined the corresponding transition probability matrix. To determine the transition probability matrix for the underlying two covariates, i.e., iron and copper, the value of metal iron and copper are divided into 5 intervals, representing the states of the Markov process. In particular, [0,15) interval for iron (ppm accumulated) is considered as State 0, i.e., healthy state; [15, 30) is considered as State 2 and so on. The classification of states and labeling of the CM data, was selected based on expert knowledge.

In the case of the C-MAPSS data, we are dealing with a situation similar to the above discussion. Our main objective for introducing the aggregated clustering method is to address the issue of dependence on expert knowledge. In this regard, we have developed an ML-based solution for construction of labels without human intervention. In order to validate our proposed model, we evaluate the performance of the model by considering different maintenance policies. It is worth mentioning that, to the best of our knowledge, this is the first work on the C-MAPPS data developed within the PHM context as such we cannot directly compare our results with other methods. Therefore, to validate the proposed model, we evaluate its performance under different maintenance costs and policies by considering cost minimization as the objective function based on the newly developed RL model. In particular, performance evaluations are performed under the following six different scenarios:

1. **Scenario 1**: Replace only on failure (R-O-O-F) policy with failure cost of 100, i.e., $C_F = 100$.
2. **Scenario 2**: R-O-O-F policy with failure cost of 40, i.e., $C_F = 40$.
3. **Scenario 3**: R-O-O-F policy with failure cost of 20, i.e., $C_F = 25$.
4. **Scenario 4**: Perform minor and major repairs with associated costs of $C_{PM} = 20$ and $C_F = 40$, respectively.
5. **Scenario 5**: Perform minor and major repairs with associated costs of $C_{PM} = 20$ and $C_F = 25$.
6. **Scenario 6**: Perform minor and major repairs with associated costs of $C_{PM} = 20$ and $C_F = 100$.

Results are shown in Table 6. The following conclusions can be obtained from results: (i) As the results show, preventive maintenance strategy is the optimal maintenance policy in comparison to the R-O-O-F policy. It makes sense as the hazard rate of the system is increasing over time; (ii) By increasing the failure cost, the total cost will be increased accordingly; (iii) The average number of minor actions in Scenario 4 is less than that of Scenario 6. We were expecting this behavior from the proposed model. The logic behind this is that when the failure cost is increasing while minor maintenance cost is fixed, the system avoids to go to the failure state (due to high failure cost) and it prefers to perform more minor actions, and; (iv) When the cost of minor and major actions are close, the system will either do major or minor action which is logical, i.e., Scenario 5.



(a) First Structure (b) Second Structure (c) Third Structure

(d) Forth Structure (e) Fifth Structure (f) Sixth Structure

(g) Seventh Structure (h) Eighth Structure

Fig. 15. LSTM decision graphs.

Table 6
Evaluation results obtained under different maintenance cost and policy scenarios.

Scenarios	Total cost	Total actions	Minor action	Major action
1	98.75	399.0125	0	0.98
2	45.5	398.8625	0	1.13
3	23.125	399.075	0	0.925
4	40.75	398.7125	0.5375	0.75
5	28.6875	398.725	0.6375	0.6375
6	85.5	398.25	0.775	0.7

5.4.5. Extension to multi-unit systems

The proposed model can be modified for application to a system consisting of heterogeneous functional elements (referred to as a multi-unit system). Consider a multi-unit system (such as the power supply system) with n components. Each component can have different numbers of sensors, i.e., the ith unit can have n_i number of sensors that measure specific covariates that can indicate the health of that component. Afterwards, we can fit n different clustering models for each component and extract the cluster labels. In the RL section, the state definition should change. We can have two different scenarios and two different approaches to define the states: (i) The cluster labels can be aggregated like what we have done in the third step of our clustering procedure. In this case, the state–space has one dimension, and; (ii) We can avoid doing the aggregation and use each label independently. In this case, our state–space has n dimensions. The action-space would also change, where the number of actions for each component depends on how many different actions are applicable. The final action space is a collection of all possible combinations of localized actions.

For example, consider a system with two units ($n = 2$). The first unit with 10 sensors, and the second one with 20 sensors. First, we should fit two different k-means clustering models on the sensory data from the first unit and the second unit, and the cluster labels from each clustering model are extracted. We call these labels S_1 and S_2, with $(S_1, S_2 \in 0, 1, 2)$, respectively. In the first scenario for defining the RL state–space, the aggregation can be done as follows:

$$S_{\text{Total}} = \begin{cases} 0 & S_1 = 0 \ \& \ C_2 = 0 \\ 1 & S_1 = 0 \ \& \ C_2 = 1 \\ 2 & S_1 = 0 \ \& \ C_2 = 2 \\ 3 & S_1 = 1 \ \& \ C_2 = 0 \\ 4 & S_1 = 1 \ \& \ C_2 = 1 \\ 5 & S_1 = 1 \ \& \ C_2 = 2 \\ 6 & S_1 = 2 \ \& \ C_2 = 0 \\ 7 & S_1 = 2 \ \& \ C_2 = 1 \\ 8 & S_1 = 2 \ \& \ C_2 = 2 \end{cases}$$

It is noteworthy to mention that this type of aggregation is the most strict one since we have considered a different space for each combination, but this could be simplified as well. For example, we can define three states as follows:

$$S_{\text{Total}} = \begin{cases} 0 & S_1 = 0 \ \& \ C_2 = 0 \\ 1 & S_1 \neq C_2 \\ 2 & S_1 = 2 \ \& \ C_2 = 2 \end{cases}$$

If we want to use the second scenario to define the state, the state space is a two-dimensional array, i.e., (S_1, S_2), and there is no need for a global aggregation. Finally, if we consider three actions (i.e., do nothing, minor repair, and major repair), for each unit, we will have 9 different combinations that constitute our final action-space. As a final note, we should elaborate further application of the proposed framework to a heterogeneous multi-state system. In such scenarios, either the variability between the sensors' measurements could be high, or the type of some of the sensors' measurements could differ from each other. Since the scale of the data that each sensor can measure differs from each other, high variability could be the case most of the time in the maintenance. Feature scaling is a crucial part of the pre-processing step that takes care of the noted problem. Data normalization and standardization are two effective but simple feature scaling techniques that map the original data space into a new space in which all the features have the same scale. Solving the other problem, i.e., different data types, consists of a set of pre-processing techniques prior to using the method proposed in the paper. These pre-processing techniques should bring all the data into the same domain (format). In summary, the proposed method can be applied to a heterogeneous system by adoption of the aforementioned pre-processing techniques.

6. Conclusion

While RUL prediction is the main focus of ML modeling based on the NASA's C-MAPSS dataset, we believe that RUL alone without design of a proper decision support system is not sufficient for maintenance decision-making. The paper aimed to address this gap and proposed an efficient and novel hybrid Maintenance Decision Support System (MDSS) framework for fault diagnostic and prognostic considering CM data. The proposed MDSS model is a hybrid ML-based solution coupled with statistical techniques. We consider the MDSS to have three distinguishable states, namely, healthy, warning, and failure state. To find an optimal maintenance policy, we concentrate the attention on a time-dependent PHM augmented with a semi-supervised ML approach. The MDSS monitors the hazard rate of the system based decision graphs to continuously detect any abnormality in data that may indicate a change in the process or the system. Given that sensors' measurements in the C-MAPSS dataset do not have decision labels, an intuitively pleasing unsupervised procedure is developed to extract and build the required labels. In order to design such a smart system, different prediction models are constructed, each with a unified decision graph. Several different ML models are utilized and evaluated ranging from traditional ML models to Ensemble Learning Models and DL-based architectures. The effectiveness of the proposed MDSS model is demonstrated through comprehensive set of comparisons with different ML algorithms.

CRediT authorship contribution statement

Kamyar Azar: Data curation, Investigation, Methodology, Software, Writing – original draft. **Zohreh Hajiakhondi-Meybodi:** Data curation, Investigation, Methodology, Software, Writing – review & editing. **Farnoosh Naderkhani:** Conceptualization, Funding acquisition, Investigation, Supervision, Writing – original draft, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This work was partially supported by the Natural Sciences and Engineering Research Council (NSERC) of Canada through the NSERC Discovery Grant RGPIN 2019 06966.

References

[1] Zhao R, Yan R, Chen Z, Mao K, Wang P, Gao RX. Deep learning and its applications to machine health monitoring. Mech Syst Signal Process 2019;115:213–37.

[2] Bai Y, Sun Z, Deng J, Li L, Long J, Li C. Manufacturing quality prediction using intelligent learning approaches: A comparative study. Sustainability 2018;10(1):85.

[3] Wu B, Tian Z, Chen M. Condition-based maintenance optimization using neural network-based health condition prediction. Qual Reliab Eng Int 2013;29(8):1151–63.

[4] Tian Z, Jin T, Wu B, Ding F. Condition based maintenance optimization for wind power generation systems under continuous monitoring. Renew Energy 2011;36(5):1502–9.

[5] Jardine AK, Lin D, Banjevic D. A review on machinery diagnostics and prognostics implementing condition-based maintenance. Mech Syst Signal Process 2006;20(7):1483–510.

[6] Saxena A, Goebel K, Simon D, Eklund N. Damage propagation modeling for aircraft engine run-to-failure simulation. 2008, p. 1–9.

[7] Wang J, Wen G, Yang S, Liu Y. Remaining useful life estimation in prognostics using deep bidirectional lstm neural network. In: 2018 prognostics and system health management conference. 2018, p. 1037–42.

[8] Miao H, Li B, Sun C, Liu J. Joint learning of degradation assessment and RUL prediction for aeroengines via dual-task deep LSTM networks. IEEE Trans Ind Inf 2019;15(9):5023–32.

[9] Li X, Ding Q, Sun J-Q. Remaining useful life estimation in prognostics using deep convolution neural networks. Reliab Eng Syst Saf 2018;172:1–11.

[10] Yu W, Kim IY, Mechefske C. An improved similarity-based prognostic algorithm for RUL estimation using an RNN autoencoder scheme. Reliab Eng Syst Saf 2020;199:106926.

[11] Shi Z, Chehade A. A dual-LSTM framework combining change point detection and remaining useful life prediction. Reliab Eng Syst Saf 2021;205:107257.

[12] Xu Z, Saleh JH. Machine learning for reliability engineering and safety applications: Review of current status and future opportunities. Reliab Eng Syst Saf 2021;107530.

[13] Rengasamy D, Jafari M, Rothwell B, Chen X, Figueredo GP. Deep learning with dynamically weighted loss function for sensor-based prognostics and health management. Sensors 2020;20(3):723.

[14] Li X, Zhang W, Ding Q. Deep learning-based remaining useful life estimation of bearings using multi-scale feature extraction. Reliab Eng Syst Saf 2019;182:208–18.

[15] Zheng S, Ristovski K, Farahat A, Gupta C. Long short-term memory network for remaining useful life estimation. 2017, p. 88–95.

[16] Babu GS, Zhao P, Li X-L. Deep convolutional neural network based regression approach for estimation of remaining useful life. 2016, p. 214–28.

[17] Ellefsen AL, Bjørlykhaug E, Æsøy V, Ushakov S, Zhang H. Remaining useful life predictions for turbofan engine degradation using semi-supervised deep architecture. Reliab Eng Syst Saf 2019;183:240–51.

[18] Wang Q, Zheng S, Farahat A, Serita S, Gupta C. Remaining useful life estimation using functional data analysis. In: 2019 IEEE international conference on prognostics and health management. 2019, p. 1–8.

[19] Makis V, Jardine AK. Optimal replacement in the proportional hazards model. INFOR: Inf Syst Oper Res 1992;30(1):172–83.

[20] Makis V, Jardine A. Computation of optimal policies in replacement models. IMA J Manag Math 1991;3(3):169–75.

[21] Wu X, Ryan SM. Value of condition monitoring for optimal replacement in the proportional hazards model with continuous degradation. IIE Trans 2010;42(8):553–63.

[22] Tang D, Makis V, Jafari L, Yu J. Optimal maintenance policy and residual life estimation for a slowly degrading system subject to condition monitoring. Reliab Eng Syst Saf 2015;134:198–207.

[23] Duan C, Makis V, Deng C. A two-level Bayesian early fault detection for mechanical equipment subject to dependent failure modes. Reliab Eng Syst Saf 2020;193:106676.

[24] Zheng R, Chen B, Gu L. Condition-based maintenance with dynamic thresholds for a system using the proportional hazards model. Reliab Eng Syst Saf 2020;204:107123.

[25] Alaswad S, Xiang Y. A review on condition-based maintenance optimization models for stochastically deteriorating system. Reliab Eng Syst Saf 2017;157:54–63.

[26] Wong EL, Jefferis T, Montgomery N. Proportional hazards modeling of engine failures in military vehicles. J Qual Maintenance Eng 2010.

[27] Jardine A, Makis V, Banjevic D, Braticevic D, Ennis M. A decision optimization model for condition-based maintenance. J Qual Maintenance Eng 1998.

[28] Chen N, Chen Y, Li Z, Zhou S, Sievenpiper C. Optimal variability sensitive condition-based maintenance with a cox ph model. Int J Prod Res 2011;49(7):2083–100.

[29] Li Z, Zhou S, Choubey S, Sievenpiper C. Failure event prediction using the cox proportional hazard model driven by frequent failure signatures. IIE Trans 2007;39(3):303–15.

[30] Naderkhani F, Jafari L, Makis V. Optimal cbm policy with two sampling intervals. J Qual Maintenance Eng 2017.

[31] Jafari L, Naderkhani F, Makis V. Joint optimization of maintenance policy and inspection interval for a multi-unit series system using proportional hazards model. J Oper Res Soc 2018;69(1):36–48.

[32] Yao L, Dong Q, Jiang J, Ni F. Deep reinforcement learning for long-term pavement maintenance planning. Comput-Aided Civ Infrastruct Eng 2020;35(11):1230–45.

[33] Wei S, Bao Y, Li H. Optimal policy for structure maintenance: A deep reinforcement learning framework. Struct Saf 2020;83:101906.

[34] Rocchetta R, Bellani L, Compare M, Zio E, Patelli E. A reinforcement learning framework for optimal operation and maintenance of power grids. Appl Energy 2019;241:291–301.

[35] Huang J, Chang Q, Arinez J. Deep reinforcement learning based preventive maintenance policy for serial production lines. Expert Syst Appl 2020;160:113701.

[36] Liu Y, Chen Y, Jiang T. Dynamic selective maintenance optimization for multi-state systems over a finite horizon: A deep reinforcement learning approach. European J Oper Res 2020;283(1):166–81.

[37] Andriotis CP, Papakonstantinou KG. Deep reinforcement learning driven inspection and maintenance planning under incomplete information and constraints. Reliab Eng Syst Saf 2021;212:107551.

[38] Zhan N, Si W. Deep reinforcement learning for condition-based maintenance planning of multi-component systems under dependent competing risks. Reliab Eng Syst Saf 2020;203:107094.

[39] Yang H, Li W, Wang B. Joint optimization of preventive maintenance and production scheduling for multi-state production systems based on reinforcement learning. Reliab Eng Syst Saf 2021;107713.

[40] Zhang C, Lim P, Qin AK, Tan KC. Multiobjective deep belief networks ensemble for remaining useful life estimation in prognostics. IEEE Trans Neural Netw Learn Syst 2017;28(10):2306–18.

[41] Jardine AK, Tsang AH. Maintenance, replacement, and reliability: theory and applications. CRC Press; 2005.

[42] Banjevic D, Jardine AKS, Makis V, Ennis M. A control-limit policy and software for condition-based maintenance optimization. INFOR: Inf Syst Oper Res 2001;39(1):32–50.

[43] Jardine AKS, Anderson PM, Mann DS. Application of the weibull proportional hazards model to aircraft and marine engine failure data. Qual Reliab Eng 1987;3:77–82.

[44] Ma M, Chen X, Wang S, Liu Y, Li W. Bearing degradation assessment based on weibull distribution and deep belief network. In: 2016 international symposium on flexible automation. IEEE; 2016, p. 382–5.

[45] Zhou P, Yin PT. An opportunistic condition-based maintenance strategy for offshore wind farm based on predictive analytics. Renew Sustain Energy Rev 2019;109.

[46] Lu Y, Sun l, Zhang X, Feng F, Kang J, Fu G, et al. Condition based maintenance optimization for offshore wind turbine considering opportunities based on neural network approach. Appl Ocean Res 2018;74:69–79.

[47] Zheng R, Makis V. Optimal condition-based maintenance with general repair and two dependent failure modes. Comput Ind Eng 2020;141.

[48] Davidson-Pilon C, Kalderstam J, Jacobson N, Reed S, Kuhn B, Zivich P, et al. Camdavidsonpilon/lifelines: v0.24.16 (version v0.24.16). Zenodo. 2020, http://dx.doi.org/10.5281/zenodo.3937749, 2020 July 9.

[49] Xia F, Ning J, Huang X. Empirical comparison of the Breslow estimator and the kalbfleisch prentice estimator for survival functions. J Biom Biostat 2018;9:2.

[50] Makis V, Jardine AKS. Optimal replacement in proportional hazard model. Inf Syst Oper Res 1992;30(1):172–83.

[51] Jardine AKS, Makis V, Banjevic D, Braticevic D, Ennis M. A decision optimization model for condition-based maintenance. J Qual Maintenance Eng 1998.