Research Article

Mouna Tarik*, Ayoub Mniai, Khalid Jebari, and Aziz Ettouhami

A new model for maintenance prediction using altruistic dragonfly algorithm and support vector machine

https://doi.org/10.1515/jisys-2023-0078
received June 21, 2023; accepted July 29, 2024

Abstract: Predictive maintenance (PdM) is a proactive approach aimed at anticipating the future point of failure for a machine or a component, with the goal of reducing both the frequency and the expenses associated with unplanned downtime. Recent advances in machine learning (ML) techniques have enabled PdM to be more efficient with diverse and successful applications in various manufacturing industries. The support vector machine (SVM), a fundamental ML algorithm, is renowned for its effectiveness in addressing classification and regression tasks. Nevertheless, the successful application of SVM hinges on the careful tuning of its parameters, a process that significantly influences its predictive performance. This research seeks to optimize the selection of the regularization parameter C and the kernel parameter σ using metaheuristic methods. It suggests combining the altruistic dragonfly algorithm (ADA) with SVM to enhance the prediction of maintenance failures. The primary motive for integrating altruism into this research is the unprecedented utilization of altruistic principles within this specific area. In addition, ADA-SVM provides a balance between exploration and exploitation. This balance is achieved through the altruistic behavior of dragonflies, where they help each other find better solutions. Therefore, this model is not trapped in the local optimum. The effectiveness of the model ADA-SVM is assessed on aircraft engine sensor data in comparison with other metaheuristic optimization algorithms, namely, genetic algorithms (GA), particle swarm optimization (PSO), grey wolf optimization (GWO) and dragonfly algorithm (DA). The performance of the SVM has been improved significantly by using parameter optimization. Besides, while GA-SVM, PSO-SVM, and DA-SVM models were able to predict engine failures with 95% accuracy, and the GWO-SVM, which demonstrated a good performance in terms of accuracy compared to other metaheuristics algorithms, achieves an accuracy of 97%, the ADA-SVM reached the best accuracy value which is 98%. The findings, thus, reveal that the proposed model outperforms the other models in optimizing SVM parameters, and, therefore, improves the performance of the engines failures prediction.

Keywords: predictive maintenance, support vector machine, altruistic dragonfly algorithm, genetic algorithms, grey wolf optimization, particle swarm optimization

1 Introduction

Maintenance expenses form a substantial component within the total operating costs of every manufacturing plant [1]. Predictive maintenance (PdM) is, therefore, a prominent method aimed at addressing maintenance

*** Corresponding author: Mouna Tarik,** IABL, FSTT, University Abdelmalek Essaadi, Tetouan, BP 416, Morocco, e-mail: mouna.tarik@etu.uae.ac.ma
Ayoub Mniai: IABL, FSTT, University Abdelmalek Essaadi, Tetouan, BP 416, Morocco, e-mail: ayoub.mniai@etu.uae.ac.ma
Khalid Jebari: IABL, FSTT, University Abdelmalek Essaadi, Tetouan, BP 416, Morocco, e-mail: khalid.jebari@uae.ac.ma
Aziz Ettouhami: LCS Laboratory, Faculty of Sciences, Mohammed V University, Rabat, BP 1014, Morocco, e-mail: touhami@fsr.ac.ma

costs, minimizing downtime, and reducing the risks of failures. It also predicts the remaining useful life (RUL) of manufacturing systems or components. Approximately 200 billion dollars are allocated annually by the US industry for the maintenance of plants, equipment, and facilities. Ineffective maintenance, conversely, leads to a loss exceeding 60 billion dollars each year [2]. Amruthnath and Gupte [3] stated that the main purpose of PdM is to decrease unplanned downtime, leading to enhanced productivity and lowered production costs. Cheng et al. [2] stated that the objective of PdM is to achieve cost savings and enhance equipment reliability. "An introduction to PdM," a book by Keith Mobley [1], gives several other definitions, among which PdM techniques assist operators and technicians in monitoring the condition of equipment, automatically issuing warnings and alerts in case of potential problems.

This research addresses the identified demand and seeks to create and compare robust models in order to predict aircraft engine failures. According to the literature, machine learning (ML) stands out as the most appropriate method for dealing with prediction challenges [4]. Through real-time pattern monitoring and analysis of historical data, machines can acquire knowledge using algorithms, enabling them to make decisions or predictions. PdM, therefore, emerges as a highly pertinent domain for the application of ML within the industrial sector.

Support vector machine (SVM) becomes a well-known ML algorithm used across a wide range of applications such as PdM. One of the crucial research points in SVM is the selection of parameters. The first parameter is σ, the kernel of radial basis function, which plays an important role in the performance of SVM. The second parameter is C, which establishes a compromise between the minimization of the learning error and SVM complexity minimization. The SVM algorithm faces a significant challenge due to the random selection of its two parameters, C and σ. As a result, it becomes imperative to enhance this selection through the utilization of metaheuristics. The integration of metaheuristics techniques emerges as a compelling approach to optimize SVM's parameter selection. In this pursuit, and in order to address this performance limitation, the integration of the altruistic dragonfly algorithm (ADA) presents a promising avenue for optimizing the parameter selection process. The ADA-SVM model aims not only to improve the efficiency of the parameter selection process but also to elevate the overall predictive accuracy and generalization capabilities of the SVM algorithm.

The key contribution points of this article are outlined as follows:

- Applying the ADA in order to find out the optimal parameters for SVM. ADA-SVM is used to predict aircraft engine failures, with the aim of reducing maintenance costs.
- Comparing the proposed model with four well-known metaheuristics.
- By relying on the No free lunch theorem, the ADA metaheuristic gives the best optimization result of the two parameters C and (σ) of SVM.
- Finding out the improved model that reaches high performance in the PdM area.

It must be emphasized that, like other ML techniques, the performance of SVMs is greatly influenced by certain parameters, particularly the regularization parameter C and the kernel parameter s. Randomly choosing these parameters can result in poor generalization and very low performance. Metaheuristic algorithms, with their ability to explore and exploit search spaces effectively, can find a satisfactory optimal solution in a reasonable time. Given that the range of possible values for these parameters is vast, metaheuristics are valuable tools for enhancing the process of optimally selecting SVM parameters.

The rest of this article is organized as follows: Section 2 provides a theoretical background of PdM. Section 3 presents the proposed model. In Section 4, we provide a concise overview of genetic algorithms (GA), particle swarm optimization (PSO), and Grey wolf optimization (GWO) and how they can be used to improve the SVM parameters. Section 5 offers an overview of maintenance request and dataset used, along with a comparative analysis of experimental results, and provides research limitations. Section 6 gives a conclusion and outlines potential propositions for future research articles.

2 PdM

Maintenance strategies refer to the various approaches and techniques employed to guarantee the optimal performance and reliability of assets, systems, or equipment. These strategies are employed to prevent failures, reduce downtime, and extend the operational lifetime of assets. In the literature, various terms and categories of maintenance management strategies are prevalent [5]. This article explores the categorizations introduced by the research of Carvalho et al. [5]. Maintenance types can be classified as shown in Figure 1.

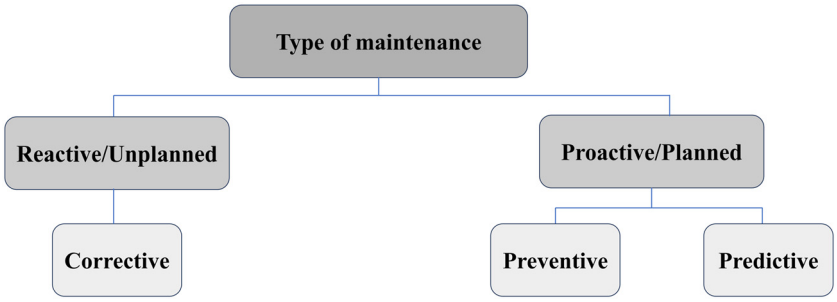

Figure 1: Maintenance strategies. Source: Created by the authors.

Run-to-Failure (R2F) or Corrective Maintenance: This the simplest maintenance strategy, and it involves the actual repair or replacement of malfunctioning, broken, or worn-down equipment. This strategy encompasses a set of actions undertaken when a machine is identified to be deviating from its intended functionality. These actions involve identifying the issue, rectifying it, and restoring the device to an operational state [6].

Preventive maintenance (PvM): When maintenance is systematically performed at scheduled intervals with the goal of anticipating potential process failures [7]. It involves the scheduling of repairs, lubrication, adjustments for all essential machinery within the plant [1].

PdM: This proactive approach helps prevent unexpected downtime and optimize maintenance efforts and costs. It allows the early detection of failures through historical data using various sensors. Different ML algorithms are used in the literature for failure prediction. In a work by Susto et al. [8], they presented a multiple classifier ML methodology for PdM applied to a benchmark semiconductor manufacturing maintenance problem. This classifier exhibits versatility, making it suitable for addressing a variety of maintenance issues characterized by integral-type faults, aiming at improving the maintenance management decisions, minimizing operating costs, and ensuring better performance compared with PvM approaches. Pech et al. [9] presented a review of the current literature concerning PdM and intelligent sensors in smart factories, and they illustrated that the significance of PdM is increasing over time, especially in correlation with Industry 4.0 technologies.

In Li et al. [10], ML techniques are employed for PdM in the railway sector. The models are applied to both historical and real-time data to predict conditions that could lead to failures, thereby preventing service interruptions and enhancing network velocity. Nowitz [11] identified, PdM as a key tool for the Smart Factory. Through the use of ML algorithms, the early detection of asset degradation and breakdowns is possible, enabling the efficient allocation of resources away from unnecessary maintenance activities. Cline et al. [12] have determined, through their research, that ML offers a complementary approach to maintenance planning. Through the analysis of substantial datasets involving individual machine performance and environmental variables, ML can discern potential failures and offer actionable predictions for specific parts. In another study, Susto [13] suggested an ML system for PdM. This system employs ML and regularized regression methods to enhance estimates of RUL along with associated costs and risks. Kim et al. [14] proposed a new prognosis model based on the SVM classifier in order to estimate the health state probability of the machine degradation process to provide long-term prediction. Yang et al. explored the potential of utilizing the random forests algorithm (RF) for machine fault diagnosis. They proposed a hybrid method that combines genetic

algorithms GAs to enhance the classification accuracy in this context [15]. The objective of Baraldi's work is the development of a diagnostic system to detect the onset of degradation and classify the type of defects using the K-nearest neighbor (KNN) classifier [16]. In the study by Jin, a novel method is introduced, leveraging GA and multi-parameter support vector machine (GSVM), for predicting the remaining life of fan bearings. Bukhsh et al. [17], applied tree-based classification techniques from ML to predict maintenance needs, activity type, and trigger status of railway switches. The primary goal of this study is to develop predictive models using available data from a railway agency, aiming to produce interpretable results. Wu et al. [18] presented a prognostic method for predicting tool wear based on random forests (RFs). Additionally, the study aims to compare the performance of RFs with feed-forward backpropagation (FFBP) artificial neural networks (ANNs) and support vector regression (SVR). Experimental results indicate that RFs can provide more accurate predictions compared to FFBP ANNs with a single hidden layer and SVR. Abed et al. [19] presented a new approach to predict the element rolling bearing defects using the dynamic neural network for fault classification. Stanulov and Yassine [20] aims to offer a survey of ML algorithms with potential utility in the aviation sector for forecasting upcoming patterns in air passenger traffic. They conducted a comparison involving long short-term memory, SVR machines, and RF, using mean squared error (MSE) and root mean squared error (RMSE) as evaluation metrics. The long short-term memory model demonstrated the highest generalization ability, achieving a performance evaluation on the testing dataset with an MSE/RMSE of 0.00445/0.06667.

3 The proposed method

PdM holds significant importance in today's industries due to several key reasons; PdM contributes to increased productivity, minimizes unexpected downtime, reduces repair costs, and prevents catastrophic failures. In the context of Industry 4.0, where interconnected systems and automation play a crucial role, PdM aligns perfectly with the concept. It integrates IoT sensors, AI-driven analytics, and real-time monitoring, facilitating a smarter and more efficient industrial landscape. It plays a pivotal role in the success and competitiveness of modern industries. In order to reach these goals and for more accurate and efficient PdM, we were looking for the importance of finding innovative approaches, hence the choice of the ADA-SVM model.

3.1 ADA

Metaheuristics are a family of approximate optimization techniques that have become popular in the last two decades [21].

They have been inspired by animals' behaviors, physical phenomena, or evolutionary concepts. The term "heuristic" originates from the ancient Greek word "heuriskein," denoting the art of discovering new strategies or rules for problem-solving. The Greek suffix "meta," meaning "upper-level methodology," was combined with "heuristic" to coin the term "metaheuristic," introduced by Glover [22]. Metaheuristics explore the search space and provide "acceptable" solutions in a reasonable time for solving hard and complex problems in science and engineering [21]. They are simple and flexible and can be applicable to different problems. They have better performance to avoid local optima. Metaheuristics find an acceptable solution but does not guarantee determining the optimal solution in each run [23]. The reason behind the robust searching mechanism of metaheuristic is the harmonization between two search schemas: exploration and exploitation [24].

While our work primarily explores the practical application and evaluation of metaheuristic algorithms in optimization scenarios, it is crucial to acknowledge and engage with alternative viewpoints that challenge the conventional understanding of their theoretical underpinnings. Camacho-Villalón et al. [25] offer a dissenting opinion regarding the efficacy of metaphorical inspirations driving the design of metaheuristic algorithms.

This discourse enables a holistic examination of the strengths and limitations inherent in these methodologies, fostering a nuanced understanding of their role in computational optimization [26].

The dragonfly algorithm (DA), created by Mirjalili [27], is a nature-inspired metaheuristic optimization algorithm. Dragonflies, being small carnivorous insects, exhibit hunting behavior, preying on various smaller insects such as butterflies, bees, ants, and mosquitoes, while avoiding potential predators like spiders and birds [28]. The DA is designed based on the natural dynamics, involving migratory phases, and static behaviors, such as feeding swarms, observed in dragonflies [28]. These swarming behaviors align with the two primary phases of optimization in metaheuristics: exploration and exploitation. Five parameters are used to describe the swarming behavior of dragonflies.

Separation: The swarm exhibits a tendency for preventing static collisions with neighboring individuals. The mathematical calculation for the separation between two contiguous dragonflies is expressed as follows:

$$S_i = -\sum_{j=1}^{N} X - X_j, \tag{1}$$

where X presents the position of the current individual and N is the total number of neighboring individuals.

Alignment: This value provides an indication of the current dragonfly's relative velocity to its nearby neighbors. This is mathematically formalized by the following equation:

$$A_i = \frac{\sum_{j=1}^{N} V_j}{N}, \tag{2}$$

where V_j stands for the velocity of the jth neighboring dragonfly.

Cohesion: reflects the tendency of the dragonflies to move toward the center of the swarms group. It is represented as follows:

$$C_i = \frac{\sum_{j=1}^{N} X_j}{N} - X. \tag{3}$$

Attraction: Based on the current location of the food, this parameter indicates how close the dragonflies should fly to it. It is calculated as follows:

$$F_i = F_p - X, \tag{4}$$

where X is the position of the current dragonfly and F_p is the position of the food source.

Distraction: This parameter regulates the distance at which dragonflies should navigate to evade potential enemy. This is modeled mathematically by

$$E_i = E_p + X, \tag{5}$$

where E_p is the enemy position and E_i denotes the position of the enemy of the ith individual. In order to calculate the next position of the dragonflies, the step vector parameter is introduced and given as

$$\Delta X_{t+1} = (sS_i + aA_i + cC_i + fF_i + eE_i) + w\Delta X_t, \tag{6}$$

where s, a, c, f, and e are random weight coefficients related, respectively, to separation, alignment, cohesion, attraction, and distraction, w presents the inertia weight, and t is the iteration counter [24].

As shown in the equation below, the new position of the dragonflies is determined with the help of the step vector:

$$X_{t+1} = X_t + \Delta X_t. \tag{7}$$

To facilitate exploitation, low alignment and high cohesion weights are employed, whereas for exploration, high alignment and low cohesion weights are utilized [27]. In the absence of neighboring solutions, a random walk, specifically a Levy fly, is employed to enhance exploration, introduce randomness, and contribute to the exploitation of dragonflies. The Levy is calculated as follows:

$$\text{levy}(x) = 0.01 \times \frac{r_1 \times \sigma}{r_2^{\frac{1}{\beta}}}, \tag{8}$$

where β is a constant, r_1 and r_2 are two random numbers in the range [0, 1] and σ is calculated as

$$\sigma = \left(\frac{\gamma(1+\beta) \times \sin\left(\frac{\pi\beta}{2}\right)}{\gamma\left(\frac{1+\beta}{2}\right) \times \beta \times 2^{\frac{\beta-1}{2}}} \right)^{\frac{1}{\beta}}. \tag{9}$$

Altruism refers to selfless concern and driven by a genuine desire to benefit others and promote their welfare. It is a simple technique that serves as a local search for the optimizer and reduces the likelihood of getting stuck in a local optimum [27]. It takes very less time to execute, and it is easy to conceptualize and implement [27]. Altruism is used to enhance the searchability of the DA.

The dragonfly that performs altruism is known as the altruist, and the dragonfly that receives the benefits is called the beneficiary [27]. A random number is used to vary the positions of the altruist and beneficiary, but they are always guaranteed to be between the lower and upper bounds of the problem according to Hamilton's rule [29] in the following equation:

$$r \times b > c, \tag{10}$$

where c represents the fitness cost to the altruist, b denotes the fitness benefit to the beneficiary, and r signifies their genetic relatedness [29]. Its advantages over other optimization techniques lie in its enhanced exploration, global optima search, and its capacity to navigate complex solution spaces.

3.2 SVM

The SVM is a supervised ML algorithm introduced by Vapnick [30] capable of both classification and regression tasks. SVMs are often referred to as maximum margin classifiers, as they aim to identify the optimal separating hyperplane between two classes [31]. Let us establish labeled training examples as $[x_i, y_i]$, an input vector $x_i \in R^n$, a class value $y_i \in \{-1, 1\}$, $i = 1, \ldots, l$. Linear separable means, for the training data a discriminant function $w^t x + b$ exists which defines a classification function $\text{sign}(w^t x + b)$, for which:

$$y = \text{sign}(w^T x + b), \quad 1 \le i \le l, \tag{11}$$

where Y is the outcome, $w \in R^n$ is a weight vector, and the bias b is a scalar.

The margin is defined as the minimal distance of the boundary function $w^t x + b = 0$ to all data points. The points located nearest to the dividing hyperplane are known as the support vectors. In general, in the case of linear separability, the decision rules are dictated by an optimal hyperplane, which separates the binary decision classes, can be expressed through the following equation in terms of the support vectors [32]:

$$Y = \text{sign}\left(\sum_{i=1}^{N} y_i \alpha_i (\mathbf{x} \cdot \mathbf{x}_i) + b \right), \tag{12}$$

where Y is the outcome and y_i is the class value of the training example x_i. The vector $x = (x_1, x_2, \ldots, x_n)$ corresponds to an input and the vectors $x_i, i = 1, \ldots, N$, are the support vectors. i are parameters that determine the hyper-plane. For the non-linearly separable case, a high-dimensional version of equation (9) is given as follows:

$$Y = \text{sign}\left(\sum_{i=1}^{N} y_i \alpha_i K(\mathbf{x}, \mathbf{x}_i) + b \right), \tag{13}$$

where $K(x, xi)$ is defined as the kernel function, with the constraint $0 \le \alpha_i \le C$ and C is a trades off margin parameter.

The well-known kernel functions for SVM are

– Linear kernels with the forms: $K(x_i, x_i') = \sum_{j=1}^{P} x_i x_{ij}$.

– Polynomial kernels: $K(x_i, x_i') = (1 + \sum_{j=1}^{P} x_i x_{ij}')^d$.
– Radial basis function kernels: $K(x_i, x_i') = \exp(-\gamma \sum_{j=1}^{P} (x_{ij} - x_{ij})^2)$.
– Gaussian kernels: $K(x_i, x_i') = \exp\left(-\frac{\sum_{j=1}^{P}(x_{ij} - x_{ij})^2}{2\sigma}\right)$.

In this work, we have used the RBF kernel and below are some advantages and disadvantages of utilizing the SVM algorithm:

Pros: Low generalization error, computationally inexpensive, handles non-linear data efficiently by using the kernel trick.

Cons: Prone to sensitivity regarding the selection of tuning parameters and choice of kernel. The SVM model is difficult to understand and interpret [33].

3.3 The ADA-SVM model

SVMs' performance is greatly influenced by its parameters, particularly the regularization parameter C and the kernel parameter σ. The random choice of these parameters can lead to a poor generalization and suboptimal performance [34].

Metaheuristic algorithms, with their capacity to navigate complex parameter spaces and optimize objective functions, stand poised as potentially valuable contenders for enhancing the parameter selection process in SVM. The ADA lies in its capacity to maintain a balance between exploration (searching diverse areas of the solution space) and exploitation (concentrating on promising areas with high fitness values). This balance is attained through the altruistic behavior of dragonflies, where they collaborate to find better solutions. Figure 2 describes the proposed model using SVM optimized by ADA.

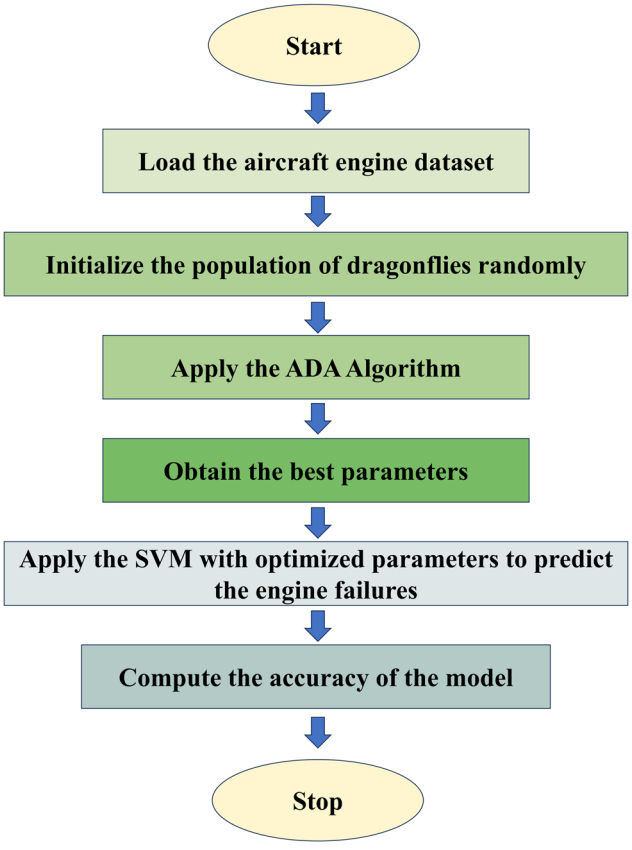

Figure 2: Flowchart of the suggested ADA-SVM model. Source: Created by the authors.

The process begins by initializing a population of dragonflies, these solutions correspond to different combinations of SVM's parameters (C and σ). The ADA-SVM method incorporates the altruistic behavior of dragonflies. Dragonflies share information about their fitness values with each other and make adjustments to their positions accordingly. This encourages exploration of promising regions in the parameter space. Dragonflies in the population continue to explore the solution space while exploiting regions with higher fitness. This balance between exploration and exploitation is a key feature of ADA. The algorithm performs multiple iterations, allowing dragonflies to iteratively adjust their positions based on their interactions and information sharing. This process gradually converges toward optimal or near-optimal parameter values for SVM. Once the iterations are complete, the dragonfly with the best fitness value (representing the optimal or near-optimal parameters) is selected as the final solution.

The SVM pseudocode after applying the ADA algorithm:

Algorithm 1 SVM improved by ADA algorithm pseudo-code

Require: A set of given input-output data pairs (or M training points)
Require: C and σ found by ADA algorithm
Ensure: The separating hyperplane designed by support vectors (only $\alpha_i > 0$)
  compute SVM solution α_i for training data: equation (3)
  compute outputs Y for all x_i: equation (3)
  while ((no changes in α) AND (iteration ≤ $\text{Max}_{\text{Iteration}}$))
    Compute Y
    Optimize α_i
    iteration = iteration + 1
  end while

Interpretable insights from the ADA-SVM model can aid in making maintenance decisions by ensuring an early detection of failures, and the allocation of maintenance resources to the assets or equipment with the highest predicted maintenance needs. Through the monitoring and continuous evaluation of model performance, organizations can refine and enhance their maintenance strategies over time.

3.4 Model complexity

The time complexity of an algorithm is the amount of time required for an algorithm to be executed. Since ADA-SVM combines both ADA and SVM, the overall complexity is influenced by both components. In the case of the proposed approach, solving the SVM optimization problem can have a time complexity of $O(n^2 * d)$, where n is the number of training samples and d is the number of features. However, the proposed approach presents an additive time complexity induced by the search for optimum values of the parameters (C and sigma) found by the metaheuristics. Indeed, the time complexity of the used metaheuristics is $O(p)$, where p represent the population size.

4 The models proposed for comparison

Throughout this section, we aimed to encompass a spectrum of optimization strategies, ranging from evolutionary algorithms (GA), swarm intelligence (PSO), and nature-inspired algorithms (GWO) to a more recent metaheuristic, DA and ADA. This diversity was intended to offer a broad comparative analysis across different optimization paradigms.

4.1 GAs with support vector machine model (GA-SVM)

GAs are stochastic search techniques inspired by natural genetics and evolutionary principles [35,36] applied in searching for the global optimum for many problems. The evolutionary process typically commences with a population of randomly generated individuals and progresses through generations. In each iteration, the fitness of each individual in the population is assessed; several individuals are chosen from the current population based on their fitness, and then modified to generate a new population. This new population is employed in the subsequent iteration of the algorithm. The algorithm concludes when either a maximum number of generations has been generated, or a satisfactory fitness level has been attained for the population.

GAs perform the search process according to the following phases: initialization, selection, crossover, and mutation. Initialization: In this stage, a population of genetic structures, called chromosomes, that are randomly distributed in the solution space. This configuration serves as the starting point for the search process [37].

Selection: The main purpose of the selection operator is to accentuate favorable solutions and discard unfavorable ones within a population based on their fitness values. The fitness value, determined by a fitness function, quantifies the optimality of a solution, providing a numerical representation of the chromosome's performance.

This value is employed to rank a specific solution relative to all other solutions. Various techniques can be employed to implement selection in GAs, including tournament selection, Roulette wheel selection, rank selection, and others.

Crossover: It generates a new offspring by combining two randomly selected "best parents." This operation involves swapping corresponding segments of a string representation of the parents, extending the search for new solutions in a broader direction [37].

Mutation: It is a mechanism used to explore a wider solution space in order to avoid converging in non-adequate local minima. Mutation is a rather random operation.

Termination: The algorithm terminates if the population has converged.

Pseudocode:

- Initialize population: Generate p solutions randomly.
- Evaluate fitness: For each p, compute its fitness.
  - **while** (Max_h fitness (h)< Threshold)
  - Selection: Probabilistically select a fraction of the best individuals in p.
  - Crossover: Probabilistically form pairs of the selected individuals and produce two offsprings by applying the crossover operator. Add all offsprings to P_{new}.
  - Mutation: Choose m% of P_{New} with uniform probability. For each selected solution, invert one randomly selected bit in its representation.
  - Update population: Set P to P_{new}.
  - Evaluate fitness: For each solution p in P, compute its fitness (p)
  - **end while**
  - Return the individual from P having the best fitness.

The SVM parameters targeted for optimization include the regularization parameter C, which determines the tradeoff cost between minimizing the training error and minimizing the model's complexity, and the parameter σ, associated with the Gaussian kernel. σ defines the non-linear mapping from the input space to a high-dimensional feature space [38]. The chromosome X is represented as $X = \{x, y\}$, where x and y indicates, respectively, the regularization parameter C and σ. In order to search optimal values of SVM parameters, we start by defining the fitness function. In our study, the mean squared error (MSE) was used:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(T_i - O_i)^2, \tag{14}$$

where n is the number of all data points, T is the value to be predicted (target), and O is the model output (prediction).

The tournament selection method is adopted to decide whether an individual can survive to the next generation so that they can be placed in a matting pool for crossover and mutation. By combining the information contained in the parent individuals, the crossover produces new individuals.

We consider $X^{\text{old}} = \{x_1, x_2, \ldots, x_n\}$, $Y^{\text{old}} = \{y_1, y_2, \ldots, y_n\}$. X^{old} and Y^{old} represent the pair of populations before crossover, and X^{new} and Y^{new} represent the pair of new populations after crossover operation.

Offspring solutions are generated using

$$X^{\text{new}} = x_i + \beta(y_i - x_i), \tag{15}$$

$$Y^{\text{new}} = x_i - \beta(y_i - x_i), \tag{16}$$

where β was a random number distributed in [0, 1]. Different mutation strategies are used; the uniform mutation method is applied in this study. The values are randomly mutated upward or downward.

4.2 Particle swarm optimization with support vector machine model (PSO-SVM)

Particle swarm optimization (PSO) is a nature-inspired algorithm that iteratively optimizes a problem to enhance a candidate solution based on a given measure of quality. Initially developed by Russell Eberhart and James Kennedy in 1995 [39], PSO was inspired by the social behavior of birds and fish. It is known for its ease of implementation and computational efficiency in terms of memory requirements and speed. PSO stands out as one of the most useful and well-known metaheuristics, having been successfully applied to a variety of optimization problems. The fundamental concept of the algorithm involves creating a swarm of particles that traverse the space, actively seeking their goal. Two optimization properties underlie the PSO algorithm:

- A particle determines its best position based on its individual experience, incorporating knowledge from the movements of other particles. The evaluation of positions is conducted through a fitness function, and the definition of which depends on the specific problem being optimized.
- To facilitate optimal exploration of the problem space, each particle's velocity incorporates a stochastic factor, enabling them to traverse unknown regions within the problem space. Each particle is characterized by a position and a velocity calculated as follows:

$$\text{Position } x_i(t+1) = x_i + v_i(t+1), \tag{17}$$

$$\text{Velocity } v_i(t+1) = v_i(t) + C1^{*}\text{Rand}_{(0,1)}[\text{PBest}_i(t) - x_i(t)] + C2^{*}\text{Rand}_{(0,1)}[\text{GBest}_i(t) - x_i(t)], \tag{18}$$

where x_i is the position of the particle i, v_i is the velocity of the particle i, PBest is the personal best position for the particle i, GBest is the global best position, Rand(0,1) is the random value between 0 and 1, $C1$ is the acceleration constant corresponding to the cognitive component, $C2$ is the acceleration constant corresponding to the social component. An inertia weight (w) is employed to regulate the velocity, influencing the convergence, exploration, and exploitation processes in the algorithm [40].

$$\text{Velocity } v_i(t+1) = w^{*}v_i(t) + C1^{*}\text{Rand}_{(0,1)}[\text{PBest}_i(t) - x_i(t)] + C2^{*}\text{Rand}_{(0,1)}[\text{GBest}_i(t) - x_i(t)]. \tag{19}$$

The way positions and velocities are initialized, which can have an important impact on the performance of the algorithm.

The PSO pseudocode is presented as follows:

Pseudocode:

– Initialize particle population
– **for** t = 1 To maximum number of generations
– **for** each particle i **do**
* **if** (x_i < Pbest$_i$) **then**
Pbest$_i$ = x_i
Gbest$_i$ = mini Pbest$_i$
* **end if**

* **for** each dimension d **do**
  $v_{i,d}(t+1) = v_{i,d}(t) + C1{*}R1(PBest_i(t) - x_{i,d}(t) + C2{*}R2(GBest_i(t) - x_{i,d}(t)$
  $x_{i,d}(t+1) = x_{i,d}(t) + V_{i,d}(t+1)$
* **end for**
* **end for**
* **end for**

– End.

4.3 Grey wolf optimizer with support vector machine model (GWO-SVM)

The GWO algorithm emulates the leadership hierarchy and hunting behavior of grey wolves in nature. To simulate the leadership hierarchy, four types of grey wolves are utilized: alpha, beta, delta, and omega. The hunting process involves three primary steps as documented in the literature: searching for prey, encircling prey, and attacking prey [41].

For the purpose of mathematically modeling the social hierarchy of wolves, apha (α) is considered as the fittest solution. The second and third best solutions are denoted as beta (β) and delta (δ), respectively. The remaining candidate solutions are considered omega (ω).

Pseudocode:

– Initialize the grey wolf population X_i ($i = 1, 2, \ldots, n$)
– Initialize a, A, and C
– Calculate the fitness of each search agent
  X_α = the best search agent
  X_β = the second best search agent
  X_δ = the third best search agent
– **while** ($t <$ Max number of iterations) **do**
  * for each search agent
  Update the position of the current search agent
  * End for
  Update a, A, and C
  Calculate the fitness of all search agent
  Update X_α, X_β and X_δ
  $t = t + 1$
– **end while**
  return X_α,

where t denotes the current iteration, $\vec{X}$ is the position of a wolf, and $\vec{A}$ and $\vec{C}$ are coefficient vectors. Throughout iterations, α, β, and δ wolves collectively estimate the likely position of the prey. Each candidate solution refines its distance from the prey. However, candidate solutions have a tendency to diverge from the prey when $||\vec{A}j|| > 1$ and converge towards the prey when $||\vec{A}j|| < 1$.

The components of vector $\vec{a}$ linearly decrease from 2 to 0 over iterations, aiming to accentuate both exploration and exploitation [42].

4.4 DA with support vector machine model (DA-SVM)

The DA is a swarm-based optimization algorithm inspired by the social behavior of dragonflies. The structure of the DA is introduced in Section 3.1.

In the DA-SVM algorithm, each dragonfly is a solution, representing a combination of parameters set (C, σ). During the optimization process, the DA iteratively updates the positions of the dragonflies based on their

fitness values and interactions with other dragonflies in the swarm. The main process of the DA algorithm is given as follows:

Pseudocode:

– Initialize the swarm X_i ($i = 1, 2, \ldots, n$)
– Initialize ΔX_i ($i = 1, 2, \ldots, n$)
– **while** (not stopping-criteria) **do**
– Evaluate all dragonflies
– Update the source food and enemy
– Update coefficients s, a, c, f, and e
– Calculate S_i, A_i, C_i, F_i, and E_i using equations (1) to (5)
– Update the neighboring radius.
– **if** A dragonfly has at least one neighboring dragonfly **then**
– Update velocity vector using equation (6)
– Update position vector using equation (7)
– **else**
– Update position vector using Levy flight equation (8)
– **end if**
– Return the best agents.
– **end while**

Table 1 presents a comparison study that highlights both the strengths and limitations of the optimization algorithms used.

Table 1: Comparison between different algorithms' strengths and limitations

Algorithm	Strength	Limitation
GA	Highly parallelizable, can be utilized for a diverse range of optimization problems, and adapt to changing problem landscapes	Requires careful tuning of various parameters and significant memory to store the population
PSO	Has a simple concept and implementation and is capable of finding global optima in complex, non-convex search spaces	Sensitive to Initialization and can get stuck in local optima if exploration is insufficient
GWO	Suitable for high-dimensional optimization Often converges quickly to a solution. Computationally efficient for many optimization tasks	Convergence to the global optimum can be influenced by factors like population size and exploration intensity
ADA	Provides a balance between exploration and exploitation. The altruistic behavior enables the algorithm to avoid getting trapped in local optima	Sensitive to the choice of parameters tuning (the number of dragonflies, iterations)
DA	Less complex compared to some other optimization algorithms	For high-dimensional and complex optimization problems, DA might face challenges in scalability and might struggle to efficiently explore vast search spaces

5 Experiments

5.1 The dataset

For Airlines companies, predicting engine failures ahead of time is a key to reducing the costly delays on their flights. The examination of engine status and functioning data historically pulled by the sensors is meant to ease the failure prediction of the material and equipment in usage, so the maintenance can be planned in a timely manner. The ML algorithm aims to predict engine failures within a given period of time by scrutinizing

engine sensor values over time, so it can learn the relation between these values and their changes to the historical failures.

The dataset comprises simulated R2F events of aircraft engines, including operational settings and measurements from 21 sensors. The assumption is that the degradation pattern of the engine is manifested in its sensor readings. The training data file encompasses 20,632 cycle records for a total of 100 engines. A detailed description is provided in Table 2.

Table 2: Description of dataset

Datatype	Multivariate
Attribute types	All numeric
Missing values	No
Number of instances	20,632
Number of attributes	26
Number of classes	2

The pie chart in Figure 3 represents the distribution of data types in the dataset. It reveals that 71.4% of the entries are float64 (shown in grey), and the other 28.6% are int64 (depicted in green). This suggests that the majority of the dataset comprises floating-point numbers, with a significant portion consisting of integer values.

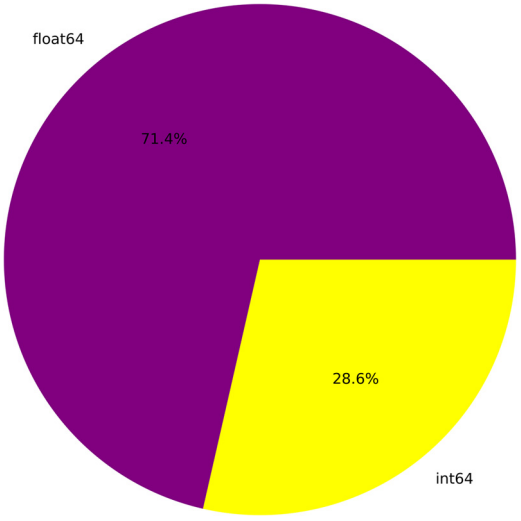

Figure 3: Distribution of data types in the dataframe. Source: Created by the authors.

The scatter plot in Figure 4 is primarily used for data visualization by reducing the data to three principal components, and it demonstrates the separability of the classes in the dataset using the first three principal components obtained from PCA. The data points are color-coded based on their class labels, with two classes: 0 (purple) and 1 (yellow), they are not linearly separable.

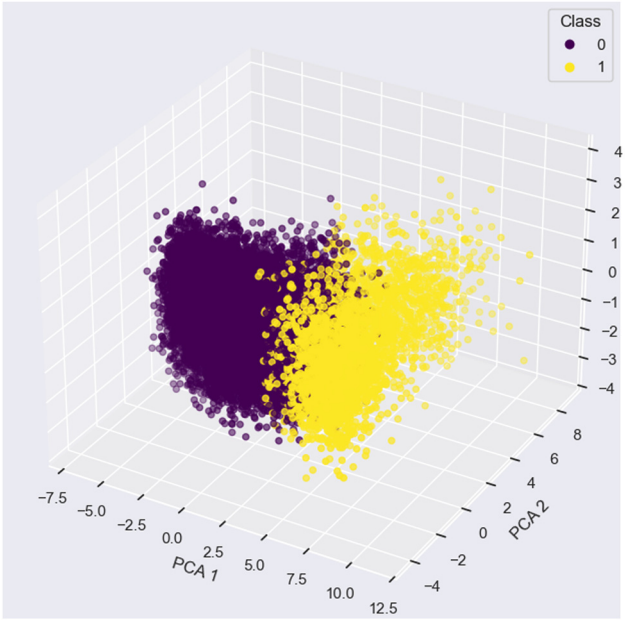

Figure 4: 3D PCA scatter plot of the dataset. Source: Created by the authors.

5.2 Results on the aircraft engine dataset

This section presents the experimental results concerning the employed ML models. The code was executed using Jupyter Notebook in Python and ran on a Core i5-7300U CPU processor. The evaluation metrics used for assessing the models are outlined in Table 3:

Table 3: Performance metrics

Metric	Formula
Recall	TP/(TP + FN)
Precision	TP/(TP + FP)
Accuracy	(TP + TN)/(TP + TN + FP + FN)
F-score	2* (Precision * Recall)/(Precision + Recall)

TP and TN are True Positive and True Negative values, and FP and FN are, respectively, False Positive and False Negative values. For the engine dataset, True Positive (TP) means that engines need maintenance and selected by the model.

True Negative (TN): engines that are fine and not selected by the model.

False Positive (FP): engines that are not fine but selected by the model.

False Negative (FN): engines that need maintenance but are not selected by the model.

The parameter ranges for experiments are presented in Table 4.

Table 4: Parameter ranges

Parameters	Min	Max	Type	Steps	Scale
C	0.1	100	Real	10	Logarithmic
σ	0.1	10	Real	10	Logarithmic

For the engine dataset and to better understand what the best parameters for tuning the GAs are, several tests were made. In our study, the parameters for GA were as follows: population size 100, crossover rate 0.5, mutation rate 0.1. We applied different tournament sizes and different percentages of new individuals in each generation. The GA-SVM model achieved 95% accuracy, and the best values of C and σ, respectively are 0.48785306578063 and 0.105606222983.

As mentioned above, to enhance performance, the SVM with an RBF kernel function necessitates the identification of optimal values for parameters C and σ. For that, we use the PSO algorithm and the optimal parameters that yield to the highest accuracy are the ones that will be used in the SVM model. After running the algorithm for 20 iterations, setting parameters of PSO on the selected benchmark dataset are as follow: the number of particles in the swarm $N = 64$, $c1 = 0.5$, $c2 = 0.3$, $w = 0.9$.

The parameter ranges for experiments are mentioned above in Table 4.

For the engine dataset, the PSO-SVM model achieved 95% accuracy, and the optimal values of C and σ, respectively, are 6.03996867 and 0.27701551.

Now, for the GWO-SVM model, the performance of the algorithm is evaluated by 100 iterations. The GWO-SVM model achieved 97% accuracy, and the best values of C and σ, respectively, are 2.64530006074 and 0.07709845. The parameters used of DA are the total number of dragonflies is 20, which gives the best results, and the attribute of Levy is 1.5 within 100 iterations. The ADA-SVM model reached 98% of accuracy with 17.62821022 and 0.06507381 as best values related to C and σ. Table 5 summarizes the values found according to the performance metrics used.

Table 5: Comparison of the methods

Model	C value	σ value	Accuracy	Precision	Recall	F-score
GA-SVM	0.49	0.11	0.95	0.93	0.98	0.95
PSO-SVM	6.04	0.28	0.95	0.93	0.98	0.95
GWO-SVM	2.65	0.08	0.97	0.94	0.99	0.96
ADA-SVM	17.63	0.06	0.98	0.97	0.98	0.97
DA-SVM	0.09	0.07	0.95	0.92	0.98	0.95

In SVM, C controls the flexibility of the model and model's ability to generalize well in unseen data. If a large value is assigned to constant c, then the margin tends to be smaller. In that case, as the tolerance of misclassification is very limited, the model tends to be less flexible. In contrast, if a very small value is assigned to constant c, then the margin tends to be larger. The tolerance level of misclassification is high, and this can result in a high misclassification rate and poor performance.

The hyperparameter σ controls the level of non-linearity introduced in the model. A small "σ" value leads to a highly non-linear decision boundary, making the model better suited for complex, non-linear data. Conversely, a large "σ" value tends to produce a more linear decision boundary, which is suitable for linearly separable data or cases where linearity is preferred. For the ADA-SVM model, the best accuracy is achieved with $C = 17.63$ and $\sigma = 0.06$. When compared to the other models, these values represent, respectively, the higher value for C and smaller for σ (Figure 5).

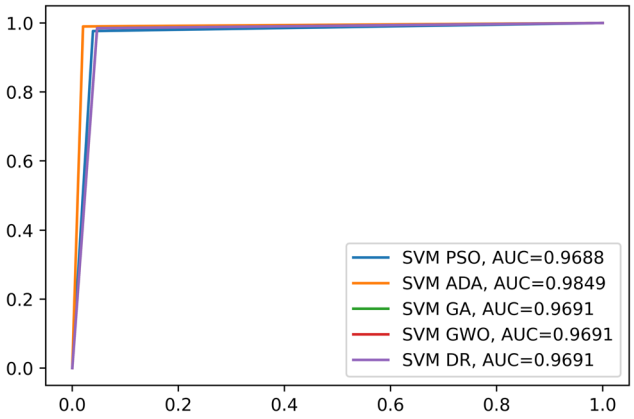

Figure 5: ROC curve. Source: Created by the authors.

Looking at the results of Table 5, we can deduce that all the models reached significant accuracy results. The best performance was attained by the ADA-SVM and GWO-SVM models. The ADA-SVM outperforms in terms of accuracy and other metrics.

To statistically evaluate the performance differences among the algorithmic approaches employed in this study, we utilized Student's *t*-test. The use of this test enables the rigorous comparison of mean performances between pairs of algorithms, allowing for an objective assessment of their relative efficacy in optimizing the SVM model for PdM tasks.

Through Student's *t*-test, we aim to ascertain whether observed differences in the accuracy. First, we calculate the *p*-value and test statistic value. After conducting independent sample *t*-tests between ADA-SVM, GA-SVM, GWO-SVM, and DA-SVM configurations, the obtained *p*-value was found to be less than the chosen significance level of 0.05. The results are presented in Figure 6 and Table 6.

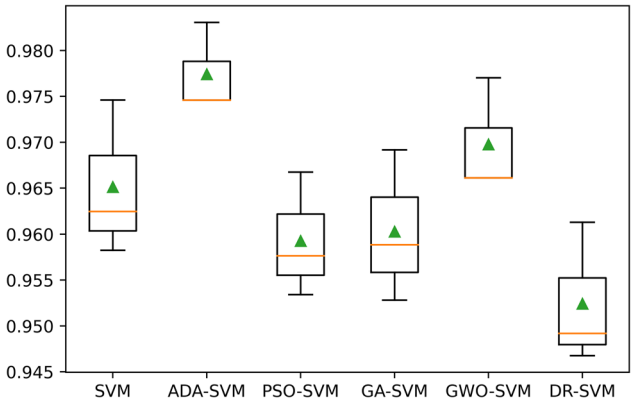

Figure 6: Plot of classification accuracy scores. Source: Created by the authors.

Table 6: Accuracy analysis

Model vs ADA-SVM	*P*-value	*T*-value
GA-SVM	0.002	–5.777
PSO-SVM	0.002	–5.741
GWO-SVM	0.08	0.97
DA-SVM	0.001	–6.843

In the context of GA-SVM, the p-value is 0.002, which is lower than 0.05, indicating strong evidence against the null hypothesis. ("The difference between mean performance is probably real".) The results obtained suggest a statistically significant difference in performance between the GA-SVM and ADA-SVM models. ADA-SVM is likely performing better based on the magnitude of the t-value.

Similar to GA-SVM vs ADA-SVM, PSO-SVM also shows a significant difference with a lower p-value (0.002) and a considerable t-value (−5.741).

There is a strong evidence of a significant difference between PSO-SVM and ADA-SVM, with ADA-SVM likely outperforming PSO-SVM based on the t-value.

For GWO-SVM, the p-value is greater than 0.05. The T-value (0.97) suggests a relatively smaller difference between GWO-SVM and ADA-SVM. This suggests that these models might perform somewhat similarly based on the measured performance metric.

5.3 Limitations

First, although SVM is a robust tool in diverse ML domains, it can have limitations such as the selection of appropriate hyperparameters, susceptibility to be prone to overfitting, and also its sensitivity to noisy data as outliers or misclassified data points. In addition, ADA itself can have limitations such as the choice of parameters tuning (the number of dragonflies, iterations, etc.). Now, once the integration of ADA with SVM is performed, this can lead to

- higher processing time,
- increased computational complexity,
- higher impact on the effectiveness of the model due to the measurement errors or variations that could occur in real-world maintenance datasets.

5.4 Discussion

Recent studies have highlighted the effectiveness of soft computing, ML techniques, and metaheuristics across various domains, showcasing their adaptability and robustness. Singh and Shrivastava [43] introduced an advanced feature selection methodology by integrating three soft computing-based optimization algorithms: teaching learning-based optimization, elephant herding optimization and a novel hybrid algorithm combining these two algorithms. Their method achieves a remarkable accuracy of 97.96%. MunishKhanna et al. [44] explored the use of ant-lion optimization to select a subset of features. These selected features are then utilized to train and evaluate four ML classifiers and their ensemble. The strategy proposed can reduce the initial feature set by up to 50% without compromising performance in terms of accuracy. In their work [45], one of the objectives is the application of the artificial bee colony algorithm for the regression test case prioritization problem and comparing its performance with other algorithms. Moreover, this study also presents two greedy-based tie-breaking algorithms designed to efficiently rank test cases based on their importance and likelihood of success. Previous study presented an improved version of SVM for PdM in industrial applications. They incorporate an entropy-based feature selection method to improve the accuracy of the model. The results show significant improvement over standard SVM methods.

Buabeng et al. [46] combined multiple ML techniques to enhance fault diagnosis accuracy and efficiency in complex industrial systems. By integrating methods such as improved complete ensemble empirical mode decomposition with adaptive noise, principal component analysis and least squares support vector machine optimized by the combination of coupled simulated annealing and Nelder-Mead simplex optimisation algorithms (ICEEMDAN-PCA-LSSVM). The analysis results indicated that the proposed ICEEMDAN-PCA-LSSVM technique is highly versatile and surpassed all the compared classifiers in terms of accuracy, error rate, and other evaluation metrics considered. Khana et al. [47] introduced a multi-objective approach for the

test suite reduction problem in which one objective is to be minimized and the remaining two maximized. In this study, experiments were performed on diverse versions of four web applications. This article introduces a multi-objective approach to address the test suite reduction problem, aiming to minimize one objective while maximizing the other two. The study conducts experiments on various versions of four web applications to validate the effectiveness of the proposed method. Through these experiments, the approach demonstrates its capability to optimize the test suite by significantly reducing the number of test cases while ensuring maximum test coverage and fault detection.

The ADA-SVM model stands out by incorporating the ADA for parameter optimization and SVM for classification. To thoroughly assess the proposed ADA-SVM model, it will be compared with other ML algorithms from the literature. Table 7 presents the results of six efficient classifiers from previous studies on aircraft engine data.

Table 7: Comparison with state-of-the-art studies

Literature	Model	Accuracy	Precision	Recall	*F*-score
[6]	LSTM	0.97	0.96	0.90	0.93
[48]	DT	0.88	0.933	0,56	0.7
[48]	ANN	0.87	0.875	0,56	0.682
[48]	SVM	0.944	0.944	0,79	0.91
[49]	CNN	0.893	UD	UD	UD
[50]	GNB	0.889	UD	UD	UD

Table 7 compares the performance metrics of various ML models used for PdM of aircraft engines. The models listed include LSTM, decision tree (DT), artificial neural network (ANN), SVM, convolutional neural network (CNN), and Gaussian Naive Bayes (GNB).

This indicates that ADA-SVM outperforms all the other algorithms listed in Table 7 in terms of classification metrics. The high accuracy of ADA-SVM implies better predictive capabilities and reliability, which are crucial for minimizing downtime and maintenance costs in industrial applications.

6 Conclusion

The effectiveness of the SVM algorithm is significantly impacted by its parameters, notably the regularization parameter C and the kernel parameter σ. Randomly selecting these parameters can result in inadequate generalization and suboptimal performance. This article has introduced a novel approach in the realm of PdM by hybridizing the ADA with the SVM to predict engine failures. The ADA algorithm refines the choice of SVM parameters allowing more precise identification of potential engine failures compared to traditional SVM. The ADA-SVM model performance is evaluated in comparison with other hybrid models that combine three well-known metaheuristics with SVM, namely, GWO, PSO and GA. To assess the performance of algorithms, metrics, like accuracy, precision, and *F*-score, are calculated. The ADA-SVM outperformed all other models, achieving the highest accuracy rate of 98%, while the GWO-SVM achieved 97%, and the remaining models, GA-SVM, PSO-SVM, and DA-SVM, reached 95%. The improved accuracy of the ADA-SVM model in predicting engine failures offers several practical advantages. It can lead to better maintenance decisions and improved operational efficiency. The accurate predictions of the ADA-SVM model enable maintenance teams to adopt a proactive approach. They can tackle potential engine failures at an early stage, preventing their escalation into significant problems. This leads to reduced unplanned downtime, avoids costly last-minute repairs, and enhances passengers safety. While the ADA-SVM model presents promising capabilities for enhancing SVM parameter selection in PdM, it presents certain limitations that can impact its applicability in real-world scenarios. The computational complexity introduced by the ADA optimization process can lead to longer

processing times, particularly with larger datasets. Furthermore, as with any ML model, the ADA-SVM's performance is constrained by the quality and representativeness of the training data. Future works will include the exploration of the ADA-SVM model in dynamic systems and benchmark its performance against various datasets. We can extend the application of the proposed model to various industries and investigate the robustness of the model under different scenarios, including noisy data, in order to perform features selection.

Acknowledgements: The authors thank the editor and referees for their valuable comments and suggestions.

Funding information: The authors state no funding involved.

Author contributions: Mouna Tarik is the principal author, performed the analysis, the algorithms selection, the data processing, Python coding and wrote the manuscript. Ayoub Mniai Contributed substantially to the development and refinement of the Python code. Khalid Jebari was involved in advising the article for important intellectual content.Aziz Ettouhami was in charge of supervision and review. The authors have accepted responsibility for the entire content of this manuscript and approved its submission.

Conflict of interest: The authors state no conflict of interest.

Data availability statement: Publicly available dataset was analyzed in this study. This data can be found at: https://www.kaggle.com/datasets/nafisur/dataset-for-predictive-maintenance.

References

[1] Keith Mobley R. An introduction to predictive maintenance. Butterworth-Heinemann. 2nd edn. 2002. p. 485–520.
[2] Chiu YC, Cheng FT, Huang HC. Developing a factory-wide intelligent predictive maintenance system based on Industry 4.0. J Chin Inst Engineers 2017;40(7):562–71. doi: 10.1080/02533839.2017.1362357.
[3] Amruthnath N, Gupta T. A research study on unsupervised machine learning algorithms for early fault detection in predictive maintenance. 2018 5th International Conference on Industrial Engineering and Aapplications (ICIEA). IEEE; 2018. doi: 10.1109/IEA.2018.8387124.
[4] Shalev-Shwartz S, Ben-David S. Understanding machine learning: From theory to algorithms. Cambridge: Cambridge University Press; 2014.
[5] Carvalho TP, Soares FA, Vita R, Francisco RD, Basto JP, Alcalá SG. A systematic literature review of machine learning methods applied to predictive maintenance. Comput Industr Eng. 2019;137:106024. doi: 10.1016/j.cie.2019.106024.
[6] Tarik M, Jebari K. Maintenance prediction based on long short-term memory algorithm. In: Kacprzyk J, Ezziyyani M, Balas VE, (eds) International Conference on Advanced Intelligent Systems for Sustainable Development. AI2SD 2022. Lecture Notes in Networks and Systems. vol 637. Cham: Springer; 2023. doi: 10.1007/978-3-031-26384-2_4.
[7] Susto GA, Beghi A, De Luca C. A predictive maintenance system for epitaxy processes based on filtering and prediction techniques. IEEE Trans Semiconductor Manufactur. Nov. 2012;25(4) : 638–49. doi: 10.1109/TSM.2012.2209131.
[8] Susto GA, Schirru A, Pampuri S, McLoone S, Beghi A. Machine learning for predictive maintenance: A multiple classifier approach. IEEE Trans Industr Inform. 2015;11:812–20. doi: 10.1109/TII.2014.2349359.
[9] Pech M, Vrchota J, Bednář J. Predictive maintenance and intelligent sensors in smart factory. Sensors 2021;21(4):1470. doi: 10.3390/s21041470.
[10] Li H, Parikh D, He Q, Qian B, Li Z, Fang D, et al. Improving rail network velocity: A machine learning approach to predictive maintenance. Transport Res Part C Emerg Tech. 2014;45:17–26. doi: 10.1016/j.trc.2014.04.013.
[11] Nowitz A. The economics of the smart factory how does machine learning lower the cost of asset maintenance part 1. 2017.
[12] Cline B, Stefan Niculescu R, Huffman D, Deckelet B. Predictive maintenance applications for machine learning. In: Annual reliability and maintainability symposium (RAMS). 2017. doi: 10.1109/RAM.2017.7889679.
[13] Gian Antonio S, Wan J, Pampuri S, Zanon M, Johnston AB, O'HARA PG, et al. An adaptive machine learning decision system for flexible predictive maintenance. 2014 IEEE International Conference on Automation Science and Engineering (CASE). IEEE; 2014. doi: 10.1109/CoASE.2014.6899418.
[14] Kim HE, Tan ACC, Mathew J, Choi BK. Bearing fault prognosis based on health State probability estimation. Engineering Asset Lifecycle Management: Proceedings of the 4th World Congress on Engineering Asset Management (WCEAM 2009). September 2009. p. 28–30. doi: 10.1016/j.eswa.2011.11.019.

[15] Yang BS, Di X, Han T. Random forests classifier for machine fault diagnosis. J Mech Sci Tech. 2008;22:1716–25. doi: 10.1007/s12206-008-0603-6.
[16] Baraldi P, Cannarile F, Di Maio F, Zio E. Hierarchical k-nearest neighbours classification and binary differential evolution for fault diagnostics of automotive bearings operating under variable conditions. Eng Appl Artif Intel. 2016;56:1–13. doi: 10.1016/j.engappai.2016.08.011.
[17] Bukhsh ZA, Saeed A, Stipanovic I, Doree AG. Predictive maintenance using tree-based classification techniques: A case of railway switches. Transport Res Part C Emerg Tech. 2019;101:35–54. doi: 10.1016/j.trc.2019.02.001.
[18] Wu D, Jennings C, Terpenny J, Gao RX, S Kumara. A comparative study on machine learning algorithms for smart manufacturing: tool wear prediction using random forests. J Manufactur Sci Eng. 2017;139:7. doi: 10.1115/1.4036350.
[19] Abed W, Sharma S, Sutton R. Diagnosis of bearing fault of brushless DC motor based on dynamic neural network and orthogonal fuzzy neighborhood discriminant analysis. 2014 UKACC International Conference on Control. IEEE; 2014. doi: 10.1109/CONTROL.2014.6915170.
[20] Stanulov A, Yassine S. A comparative analysis of machine learning algorithms for the purpose of predicting Norwegian air passenger traffic. Int J Math Stat Comput Sci. 2024;2:28–43. https://doi.org/10.59543/ijmscs.v2i.7851.
[21] Talbi EG. Metaheuristics: from design to implementation. John Wiley & Sons; 2009.
[22] Glover F. Future paths for integer programming and links to artificial intelligence. Comput Operat Res. 1986;13(5):533–49. doi: 10.1016/0305-0548(86)90048-1.
[23] Bello R, Gomez Y, Nowe A, Garcia MM. Two-step particle swarm optimization to solve the feature selection problem. Seventh International Conference on Intelligent Systems Design and Applications (ISDA 2007). IEEE; 2007. p. 691–6. doi: 10.1109/ISDA.2007.101.
[24] Blum C, Roli A. Metaheuristics in combinatorial optimization: Overview and conceptual comparison. ACM computing surveys (CSUR). 2003;35(3):268–308. doi: 10.1145/937503.937505.
[25] Camacho-Villalón CL, Dorigo M, Stützle T. Exposing the grey wolf, moth-flame, whale, firefly, bat, and antlion algorithms: six misleading optimization techniques inspired by bestial metaphors. Int Trans Operat Res. 2023;30(6):2945–71. doi: 10.1111/itor.13176.
[26] Aranha C, Camacho Villalón CL, Campelo F, Dorigo M, Ruiz R, Sevaux M, et al. Metaphor-based metaheuristics, a call for action: the elephant in the room. Swarm Intel. 2022;16(1):1–6. doi: 10.1007/s11721-021-00202-9.
[27] Mirjalili S. Dragonfly algorithm: a new metaheuristic optimization technique for solving single objective, discrete, and multi-objective problems. Neural Comput Appl. 2016;27(4):1053–73. doi: 10.1007/s00521-015-1920-1.
[28] Meraihi Y, Ramdane-Cherif A, Acheli D, Mahseur M. Dragonfly algorithm: a comprehensive review and applications. Neural Comput Appl. 2020;32:16625–46. doi: 10.1007/s00521-020-04866-y.
[29] Waibel M, Floreano D, Keller L. A quantitative test of Hamilton's rule for the evolution of Altruism. PLoS Biology. 2011;9(5):e1000615. doi: 10.1371/journal.pbio.1000615.
[30] Vapnik V. Statistical learning theory Wiley. Vol. 1, No. 624, New York; 1998. p. 2.
[31] Gray D, Bowes D, Davey N, Sun Y, Christianson B. Using the support vector machine as a classification method for software defect prediction with static code metrics. Engineering Applications of Neural Networks: 11th International Conference, EANN 2009, London, UK, August 27–29, 2009. Proceedings 11. Berlin Heidelberg: Springer; 2009. p. 223–34. doi: 10.1007/978-3-642-03969-0_21.
[32] Shin KS, Lee TS, Kim HJ. An application of support vector machines in bankruptcy prediction model. Expert Syst Appl. 2005;28(1):127–35. doi: 10.1016/j.eswa.2004.08.009.
[33] Harrington P. Machine learning in action. Manning Publications. 2012. p. 101.
[34] Harrington P. Machine learning in action. Manning Publications. 2012. p. 39.
[35] Sampson JR. Adaptation in natural and artificial systems (John H. Holland). 1976. doi: 10.1137/1018105.
[36] Goldberg DE. Genetic algorithms in search, optimization, and machine learning. Reading, Massachusetts: Addison-wesley; 1989.
[37] Shin KS, LEE YJ. A genetic algorithm application in bankruptcy prediction modeling. Expert Syst Appl. 2002;23(3):321–8. doi: 10.1016/S0957-4174(02)00051-9.
[38] Wu CH, Tzeng GH, Goo YJ, Fang WC. A real-valued genetic algorithm to optimize the parameters of support vector machine for predicting bankruptcy. Expert Syst Appl. 2007;32(2):397–408. doi: 10.1016/j.eswa.2005.12.008.
[39] Eberhart E, Kennedy J. A new optimizer using particle swarm theory. MHS'95. in: Proceedings of the Sixth International Symposium on Micro Machine and Human Science. IEEE; 1995. p. 39–43. doi: 10.1109/MHS.1995.494215.
[40] Bansal JC, Singh Mukesh Saraswat PK, Verma A, Jadon SS, Abraham A. Inertia weight strategies in particle swarm optimization. 2011 Third world congress on nature and biologically inspired computing. IEEE; Oct. 2011. p. 633–40.
[41] Mirjalili S, Mirjalili SM, Lewis A. Grey wolf optimizer. Adv Eng Softw. 2014;69:46–61. doi: 10.1016/j.advengsoft.2013.12.007.
[42] Zhou Z, Zhang R, Wang Y, Zhu Z, Zhang J. Color difference classification based on optimization support vector machine of improved grey wolf algorithm. Optik. 2018;170:17–29. doi: 10.1016/j.ijleo.2018.05.096.
[43] Singh LK, Shrivastava K. An enhanced and efficient approach for feature selection for chronic human disease prediction: A breast cancer study. Heliyon 2024;10(5):e26799. doi: 10.1016/j.heliyon.2024.e26799.
[44] MunishKhanna, Singh LK, Garg H. A novel approach for human diseases prediction using nature inspired computing & machine learning approach. Multimedia Tools Appl. 2024;83(6):17773–809. doi: 10.1007/s11042-023-16236-6.
[45] Khanna M, Chauhan N, Sharma DK. Search for prioritized test cases during web application testing. Int J Appl Metaheuristic Comput. 2019;10(2):1–26. doi: 10.4018/IJAMC.2019040101.

[46] Buabeng A, Simons A, Frempong NK, Ziggah YY. Hybrid intelligent predictive maintenance model for multiclass fault classification. Soft Comput. 2023;28:1–22. doi: 10.1007/s00500-023-08993-1.
[47] Khanna M, Chauhan N, Sharma DK. Search for prioritized test cases during web application testing. Int J Appl Metaheuristic Comput (IJAMC). 2019;10(2):1–26. doi: 10.4018/IJAMC.2019040101.
[48] Tarik M, Jebari K. Maintenance prediction by machine learning: study review of some supervised learning algorithms. In: Proceedings of the 2nd African International Conference on Industrial Engineering and Operations Management. Harare, Zimbabwe: IEOM Society International. 2020.
[49] Hermawan AP, Kim DS, Lee JM. Predictive maintenance of aircraft engine using deep learning technique. in: 2020 International Conference on Information and Communication Technology Convergence (ICTC). IEEE; 2020.
[50] Azar K, Naderkhani F. Semi-supervised learning approach for optimizing condition-based-maintenance (CBM) decisions. 2020 IEEE International Conference on Prognostics and Health Management (ICPHM). IEEE; 2020.