computers

MDPI

Article

B-PSA: A Binary Pendulum Search Algorithm for the Feature Selection Problem

Broderick Crawford [1,*], Felipe Cisternas-Caneo [1,*], Katherine Sepúlveda [1], Ricardo Soto [1], Álex Paz [2], Alvaro Peña [2], Claudio León de la Barra [3], Eduardo Rodriguez-Tello [4], Gino Astorga [5], Carlos Castro [6], Franklin Johnson [7] and Giovanni Giachetti [8]

1 Escuela de Ingeniería Informática, Pontificia Universidad Católica de Valparaíso, Avenida Brasil 2241, Valparaíso 2362807, Chile; katherine.sepulveda.b@mail.pucv.cl (K.S.); ricardo.soto@pucv.cl (R.S.)
2 Escuela de Ingeniería de Construcción y Transporte, Pontificia Universidad Católica de Valparaíso, Avenida Brasil 2147, Valparaíso 2362804, Chile; alex.paz@pucv.cl (Á.P.); alvaro.pena@pucv.cl (A.P.)
3 Escuela de Negocios y Economía, Pontificia Universidad Católica de Valparaíso, Valparaíso 2340025, Chile; claudio.leondelabarra@pucv.cl
4 Unidad Tamaulipas, Cinvestav, Km. 5.5 Carretera Victoria—Soto La Marina, Victoria 87130, Mexico; ertello@cinvestav.mx
5 Escuela de Negocios Internacionales, Universidad de Valparaíso, Viña del Mar 2572048, Chile; gino.astorga@uv.cl
6 Departamento de Informática, Universidad Técnica Federico Santa María, Avenida España 1680, Valparaíso 2390123, Chile; carlos.castro@inf.utfsm.cl
7 Departamento de Ciencias de Datos e Informática, Universidad de Playa Ancha, Valparaíso 2360004, Chile; franklin.johnson@upla.cl
8 Facultad de Ingeniería, Universidad Andres Bello, Santiago 7591538, Chile; giovanni.giachetti@unab.cl
* Correspondence: broderick.crawford@pucv.cl (B.C.); felipe.cisternas.c@mail.pucv.cl (F.C.-C.)

Abstract: The digitization of information and technological advancements have enabled us to gather vast amounts of data from various domains, including but not limited to medicine, commerce, and mining. Machine learning techniques use this information to improve decision-making, but they have a big problem: they are very sensitive to data variation, so it is necessary to clean them to remove irrelevant and redundant information. This removal of information is known as the Feature Selection Problem. This work presents the Pendulum Search Algorithm applied to solve the Feature Selection Problem. As the Pendulum Search Algorithm is a metaheuristic designed for continuous optimization problems, a binarization process is performed using the Two-Step Technique. Preliminary results indicate that our proposal obtains competitive results when compared to other metaheuristics extracted from the literature, solving well-known benchmarks.

Keywords: Pendulum Search Algorithm; Feature Selection Problem; Binarization Schemes; Combinatorial Optimization

Citation: Crawford, B.; Cisternas-Caneo, F.; Sepúlveda, K.; Soto, R.; Paz, Á.; Peña, A.; León de la Barra, C.; Rodriguez-Tello, E.; Astorga, G.; Castro, C.; et al. B-PSA: A Binary Pendulum Search Algorithm for the Feature Selection Problem. *Computers* **2023**, *12*, 249. https://doi.org/10.3390/computers12120249

Academic Editors: Osvaldo Gervasi and Damiano Perri

Received: 30 October 2023
Revised: 22 November 2023
Accepted: 27 November 2023
Published: 29 November 2023

1. Introduction

In recent years, the use of different machine learning algorithms has become of great interest in the scientific community. This is due to two main reasons: (1) access to large computational capacities and (2) large volumes of existing data. These techniques have been successfully applied in image analysis, natural language processing, and sentiment analysis, among others.

However, the large volumes of data available for processing may contain redundant or unnecessary variables for classification that decrease the accuracy of the results and the performance of the predictive algorithms. Due to this sensitivity to the input data, it is very difficult to arrive at reliable results without proper data preprocessing.

Feature selection is one of the most widely used preprocessing techniques and aims to remove irrelevant and/or redundant information in order to keep only the most representative information of the dataset [1].

In [2], the authors tell us that there are three methods to solve the Feature Selection Problem:

- Filter Methods: They are independent of learning or classification algorithms since the filtering is based on expert judgment or statistical techniques that allow us to determine which are the most important features.
- Wrapper Methods: The cleaning process is performed online, i.e., it always interacts directly with the classification or learning algorithm. This is because, iteratively, the algorithms need constant performance feedback.
- Embedded Methods: This method is a hybridization between a filter method and a wrapper method. Feature selection is part of the training process of the learning or classification algorithm.

Although wrapper methods are computationally very expensive, their performance is better than that of filter methods. Metaheuristics fall into the category of wrapper methods.

Metaheuristics are general-purpose algorithms that, with a few modifications, can solve different optimization problems. Although metaheuristics do not guarantee global optimization, these provide us with high-quality solutions in a reasonable time. This characteristic makes different authors use metaheuristics to solve the Feature Selection Problem [2,3]. Figure 1 shows how to solve the Feature Selection Problem using metaheuristics.

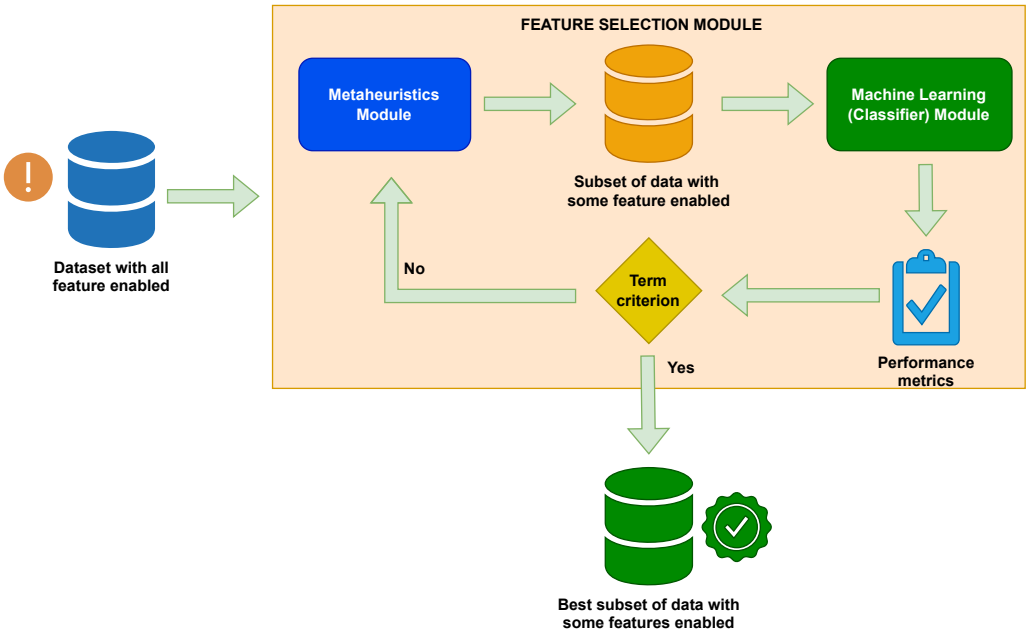


Figure 1. Feature Selection process flow chart.

As input, we have the original dataset where all features are being used. This set enters the metaheuristic module, which will be responsible for eliminating irrelevant and redundant features, resulting in a subset of data with some activated features. This subset of data enters the machine learning module, where the performance of a classifier or predictor with this subset of data will be tested. As the output of this module, we will have different performance metrics. Then, we evaluate whether we have reached our termination criterion. If we have not reached it, we return to the metaheuristic module, but with the feedback of performance obtained previously; if we have reached the termination criterion, we obtain, as a result, the best subset of data with some activated features.

In the literature, we can find different metaheuristics developed by researchers [4]. These researchers rely on the No Free Lunch (NFL) theorem to further deepen their understanding of metaheuristics, as this theorem indicates that there is no optimization algorithm capable of solving all existing optimization problems [5]. This theorem has not only led to the creation of new metaheuristics but also to the modification of existing ones, where

we can highlight binarization. Binarization involves adapting metaheuristics designed to solve continuous optimization problems so that they can solve binary combinatorial optimization problems.

According to the work conducted in [6], we can observe that the Feature Selection Problem is the most addressed issue by researchers who binarize metaheuristics. Given this significant interest, there is a motivation to binarize recent metaheuristics that have not yet solved this problem.

In this work, we binarize the Pendulum Search Algorithm (PSA), a recently developed metaheuristic designed to solve continuous optimization problems. Such binarization is performed so that PSA can solve binary combinatorial optimization problems such as the Feature Selection Problem. The motivation for using the Pendulum Search Algorithm is that it has no configurable parameters, and this saves us the process of parameter tuning.

To validate the performance of the proposal, we compared our proposal with the Sine Cosine Algorithm and Moth-Flame Optimization, two widely used metaheuristics in the literature to solve the Feature Selection Problem with good results. Preliminary results indicate that PSA has good competitive results when compared to the other two metaheuristics.

The paper is organized as follows: Section 2 defines the Feature Selection Problem, Section 3 reviews how metaheuristics solve the Feature Selection Problem, Section 4 defines the Pendulum Search Algorithm, Section 5 defines our proposal Binary Pendulum Search Algorithm, the results obtained are shown in Section 6 to end with the conclusion in Section 7.

2. Feature Selection Problem

The Feature Selection Problem (FS) is a multi-objective combinatorial optimization problem that arises from the need to eliminate irrelevant and redundant information since this information is detrimental to the learning tasks of prediction or classification algorithms. On the one hand, we need to improve the prediction of classification algorithms, and on the other hand, we need to decrease the number of features that make up the original dataset.

In its original definition, FS works by assuming a dataset S, which contains a d number of f features, such that $S = \{f_1, f_2, f_3, \ldots, f_d\}$. The objective of the problem is to select the best subset of features $D = \{f_1, f_2, f_3, \ldots, f_n\}$ with $n < d$ so that the features belonging to the selected subset are the most representative of the set of original data.

Internally, Feature Selection has a binary domain, so the representation of vectors S and D has the same character, as exemplified in Figure 2, where a 1 would mean the activation or use of a feature, and a 0 the deactivation or the discarding of it. The vector S represents the utilization of the complete original dataset; that is, all the cells of the vector have a value of 1, whereas the vector D represents the resulting subset of features; that is, the vector is composed of 1 s and 0 s.

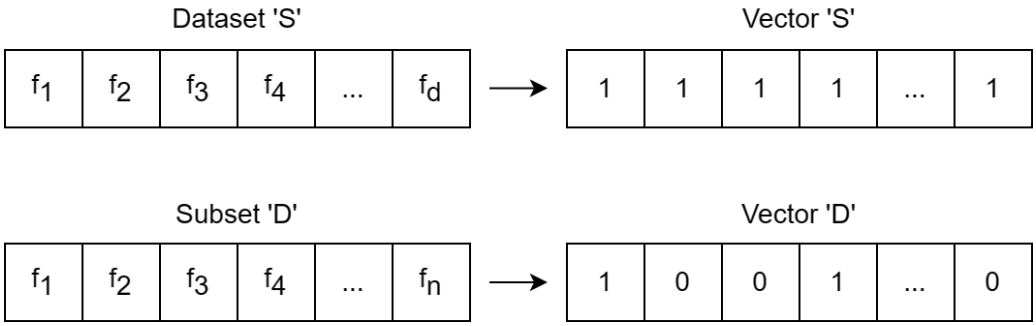


Figure 2. Binary representation of vectors S and D.

In this way, the objective function Z is calculated as the following equation:

$$min\ Z = \alpha \cdot Y_R(D) + \beta \cdot \frac{|R|}{|N|} \quad (1)$$

where $Y_R(D)$ represents the classification or prediction error of the ML algorithm, $|R|$ is the number of selected features, and $|N|$ the total number of features, with $\alpha \in [0,1]$ and $\beta = (1-\alpha)$ parameters that regulate the importance of the quality of the results and the size of the subset, respectively.

3. Continuous Metaheuristics Solving Feature Selection Problem

Using metaheuristics to solve the Feature Selection Problem is of high interest to the scientific community. In the literature [2,3,7–9], different literature reviews can be observed, focusing on documenting which metaheuristics have been used to solve the FSP. For instance, we can highlight Ant Lion Optimization (ALO) [10,11], Artificial Bee Colony Algorithm (ABC) [12,13], Bat Algorithm (BA) [14,15], Cuckoo Search (CS) [16–18], Grey Wolf Optimizer (GWO) [19–21], Dragonfly Algorithm [22], Harris' Hawk Optimization (HHO) [23,24], Moth Flame Optimization (MFO) [25–27], Sine Cosine Algorithm (SCA) [28–30] and, Whale Optimization Algorithm (WOA) [31,32].

All these metaheuristics have one characteristic in common: they are metaheuristics designed to solve continuous optimization problems that were modified to solve binary combinatorial optimization problems, such as the Feature Selection Problem.

3.1. Two Step Techniques

In the literature, there are different ways of binarizing continuous metaheuristics [6], but the most widely used is the Two-Step Technique [33]. As the name suggests, the binarization process is performed in two steps:

- Application of a transfer function, which transfers the continuous value to a value within the range [0, 1].
- Application of a binarization rule which determines the assignment of a 1 or a 0.

In the literature [6], we can find different transfer functions, among which the S-Shaped and V-Shaped transfer functions stand out. Table 1 and Figure 3 present the families of transfer functions utilized in this study. The notation d_i^j observed in Table 1 corresponds to the continuous value of the j-th dimension of the i-th individual resulting after the perturbation performed by the continuous metaheuristic.

Table 1. S-Shaped and V-Shaped transfer functions.

S-Shaped		V-Shaped	
Name	**Equation**	**Name**	**Equation**
S1	$T(d_i^j) = \frac{1}{1+e^{-2d_i^j}}$	V1	$T(d_i^j) = \left\lvert erf\left(\frac{\sqrt{\pi}}{2} d_i^j\right)\right\rvert$
S2	$T(d_i^j) = \frac{1}{1+e^{-d_i^j}}$	V2	$T(d_i^j) = \left\lvert tanh(d_i^j)\right\rvert$
S3	$T(d_i^j) = \frac{1}{1+e^{\frac{-d_i^j}{2}}}$	V3	$T(d_i^j) = \left\lvert \frac{d_i^j}{\sqrt{1+(d_i^j)^2}}\right\rvert$
S4	$T(d_i^j) = \frac{1}{1+e^{\frac{-d_i^j}{3}}}$	V4	$T(d_i^j) = \left\lvert \frac{2}{\pi} arctan\left(\frac{\pi}{2} d_i^j\right)\right\rvert$

Additionally, in the literature [33], we can find five different binarization rules, of which we can highlight the following:

- Standard (STD): If the condition is satisfied, the standard binarization rule returns 1; otherwise it returns 0. Mathematically, it is defined as follows:

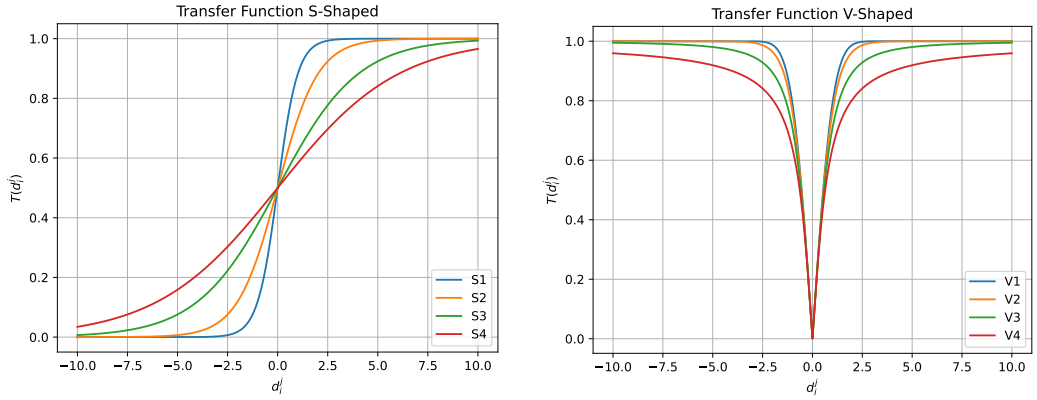
$$X_{new}^{j} = \begin{cases} 1 & if\ rand \leq T(d_i^j) \\ 0 & else. \end{cases} \tag{2}$$

- Elitist (ELIT): The best value is assigned if a random value is within the probability; otherwise, a zero value is assigned. Mathematically, it is defined as follows:

$$X_{new}^{j} = \begin{cases} X_{Best}^{j} & if\ rand < T(d_i^j) \\ 0 & else. \end{cases} \tag{3}$$

- Elitist Roulette (ELIT_ROU): This binarization rule selects randomly among the best individuals of the population, with a probability that is directly proportional to their fitness. Mathematically, it is defined as follows:

$$X_{new}^{j} = \begin{cases} P[X_{new}^{j} = \zeta_j] = \frac{f(\zeta)}{\sum_{\delta \in Q_g} f(\delta)} & if\ rand \leq T(d_i^j) \\ P[X_{new}^{j} = 0] = 1 & else. \end{cases} \tag{4}$$



(**a**) S-Shaped (**b**) V-Shaped

Figure 3. S-Shaped and V-Shaped transfer functions.

3.2. Evaluation Metrics

As seen in Equation (1), we need the error rate, and this is a classifier performance metric. However, the classifier can not only measure their performance with the error rate but also with the f-score [2]. Both metrics are constructed from the confusion matrix obtained after classifier validation. Figure 4 shows how the confusion matrix is constructed.

	Actually Positive (1)	Actually Negative (0)
Predicted Positive (1)	True Positive (tP)	False Positive (fP)
Predicted Negative (0)	False Negative (fN)	True Negative (tN)

Figure 4. Confusion Matrix.

With this, the error rate and f-score are constructed as follows:

$$Error\ rate = \frac{\mathsf{fP} + \mathsf{fN}}{\mathsf{tP} + \mathsf{fN} + \mathsf{fP} + \mathsf{tN}} \tag{5}$$

$$f.score = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall},\ where \tag{6}$$

$$Recall = \frac{\mathsf{tP}}{\mathsf{tP} + \mathsf{fN}}\ ,\ Precision = \frac{\mathsf{tP}}{\mathsf{tP} + \mathsf{fP}} \tag{7}$$

These metrics are used by the authors to demonstrate that the metaheuristics applied to solve the Feature Selection Problem did so correctly and reliably.

4. Pendulum Search Algorithm

The Pendulum Search Algorithm is a new population-based metaheuristic created in 2022 by Nor Azlina Ab. Aziz and Kamarulzaman Ab. Aziz [34]. This metaheuristic was designed to solve continuous optimization problems, and it is inspired by the harmonic motion of the simple pendulum. The equation of movement is very similar to the one proposed in Sine Cosine Algorithm. The authors incorporate an exponential function, which would improve the balance between exploration and exploitation [35].

The search agents are initialized randomly, and their position is updated using Equation (8).

$$X_{i,j}^{t} = X_{i,j}^{t} + pend_{i,j}^{t} \cdot (Best_j - X_{i,j}^{t}) \tag{8}$$

where $X_{i,j}^{t}$ is the position of the i-th solution in the j-th dimension in t-th iteration, $Best_j$ it is the position of the best solution in the j-th dimension in the t-th iteration and $pend_{i,j}^{t}$ is a parameter which is calculated as follows:

$$pend_{i,j}^{t} = 2 \cdot e^{(-t/tmax)} \cdot cos(2 \cdot \pi \cdot rand) \tag{9}$$

where t is the current iteration, $tmax$ is the maximum number of iterations and $rand$ is a uniform random number between [0, 1]. The pseudo-code of PSA is shown in Algorithm 1.

Algorithm 1 Pendulum Search Algorithm

Input: The population $X = \{X_1, X_2, \ldots, X_i\}$
Output: The updated population $X' = \{X'_1, X'_2, \ldots, X'_i\}$ and *Best*

Initialize random population X
Evaluate the objective function of each individual in the population X
Identify the best individual in the population (*Best*)
for *iteration* (t) **do**
    for *solution* (i) **do**
        for *dimension* (j) **do**
            Update $pend_{i.j}^{t}$ by Equation (9)
            Update the position of $X_{i,j}^{t}$ using Equation (8)
        end for
    end for
    Evaluate the objective function of each individual in the population X
    Update *Best*
end for
Return the updated population X' where *Best* is the best result

5. Binary Pendulum Search Algorithm

As mentioned in Section 4, PSA is a metaheuristic designed to solve continuous optimization problems, so it is necessary to convert the solutions to the binary domain to

solve the Feature Selection. Additionally, in Section 3, it was mentioned that the Two-Step Technique is one of the most widely used methods for binarizing continuous metaheuristics.

In [6,33], the authors present eight different transfer functions and five binarization rules. In this work, after preliminary experimenting with different combinations, we decided to use the following transfer function and binarization rule. Equation (10) represents the transfer function, and Equation (11) represents the binarization rule used in this work.

$$T(d_w^j) = \left| \tfrac{2}{\pi} arctan\left(\tfrac{\pi}{2} d_w^j \right) \right| \tag{10}$$

$$X_{new}^j = \begin{cases} 1 & if \;\; rand \leq T\left(d_w^j\right) \\ 0 & otherwise \end{cases} \tag{11}$$

with this, we can construct the Binary Pendulum Search Algorithm (B-PSA). First, we initialize the solutions in the binary domain, and in each iteration, the following will be performed: (1) perturbing the binary solutions with Equations (8) and (9), the classical equation of motion of the PSA. (2) after perturbing all the solutions, they will leave the binary domain, and binarization will be applied using Equations (10) and (11). This process is repeated until the end of the iterations. After the solution generation and binarization step, a feasibility test and solution repair are performed (Algorithm 2). In the feasibility test, we verify that each solution has at least one activated feature; that is, there is at least one "1" in the solution vector. If this condition is not met, a new random binary solution is generated, and the feasibility test is applied again. This process is repeated until all feasible solutions are obtained. The pseudo-code of B-PSA is shown in Algorithm 3.

Algorithm 2 Feasibility test and repair solution

Input: The population $X = \{X_1, X_2, \ldots, X_i\}$
Output: The feasible population $X = \{X_1, X_2, \ldots, X_i\}$

repeat
  for *solution* (i) **do**
    if $solution_i$ is composed of only 0 **then**
      Generate a new binary random solution
    else
      feasible solution
    end if
  end for
until all solutions are feasible
return the feasible population X

Algorithm 3 Binary-Pendulum Search Algorithm

Input: The population $X = \{X_1, X_2, \ldots, X_i\}$
Output: The updated population $X' = \{X_1', X_2', \ldots, X_i'\}$ and *Best*

Initialize binary random population X
Apply feasibility test according to Algorithm 2
Evaluate the objective function of each individual in the population X
Identify the best individual in the population (*Best*)
for *iteration* (t) **do**
  for *solution* (i) **do**
    for *dimension* (j) **do**
      Update $pend_{i.j}^t$ by Equation (9)
      Update the position of $X_{i,j}^t$ using Equation (8)

Algorithm 3 *Cont.*

```
        end for
    end for
    Binarization of population X
    Apply feasibility test according to Algorithm 2
    Evaluate the objective function of each individual in the population X
    Update Best
end for
return the updated population X′ where Best is the best result
```

6. Experimental Results

To validate the performance of our proposal, test datasets provided by https://archive.ics.uci.edu/ (accessed on 29 October 2023) were used and compared to the Binary Sine Cosine Algorithm and Binary Moth-Flame Optimization. Experiments using the Sine Cosine Algorithm and Moth-Flame Optimization were run from zero without using results from previous work. In this repository, there are several datasets where different authors validate their algorithms that solve the Feature Selection Problem. Table 2 shows the characteristics of the resolved instances. The first column represents the solved instance, the second column represents the data type of the datasets, the third column refers to the field belonging to the instance, the fourth column refers to the number of features that compose the datasets, the fifth column refers to the number of samples in the datasets, the sixth column represents the number of classes to be classified and the seventh column refers to whether the dataset has missing data.

Table 2. Instances used.

Instance	Data Type	Area	N° of Features	Sample Size	N° of Class	Missed Value?
Breast cancer wisconsin (Original)	Integer	Life	10	699	2	Yes
Divorce	Integer	Life	54	170	2	No
Immunotherapy	Integer Real	Life	8	90	2	No
Ionosphere	Integer Real	Physical	34	351	2	No
Connectionist Bench (Sonar, Mines vs Rocks)	Real	Physical	60	208	2	No
Breast Cancer Wisconsin—Diagnostic (wdbc)	Real	Life	32	569	2	No

The K-Nearest Neighbors (KNN), one of the most widely used classifiers in the literature [2], was used as a classifier in the Machine Learning Module mentioned in Figure 1. The datasets were separated into two, with 75% of the data being used as training data and 25% of the data being used as validation data. The population size used for the three metaheuristics is 10 individuals, and 100 iterations were performed. The rest of the parameters are shown in Table 3.

Fitness and f-score are the metrics used to show the good performance of our proposal. Table 4 shows the fitness obtained for each instance using the three metaheuristics under study. The first column of this table indicates the instance under study, and the second, third, and fourth columns are repeated for each metaheuristic. The first column indicates the best fitness obtained; the second indicates the average obtained with the 31 independent runs performed; the third indicates the standard deviation of the 31 runs performed. The best result per instance is highlighted in bold type.

Table 3. Configuration of parameters.

Parameter	Value
Population size	10
Iterations	100
Independent runs	31
Parameter k of KNN	5
Parameter b of MFO	1
Parameter a of SCA	2
PSA has no parameteris	-

Table 4. Fitness obtained.

Instance	B-PSA			B-SCA			B-MFO		
	best	avg	std-dev	best	avg	std-dev	best	avg	std-dev
ionosphere	0.034	0.045	0.011	0.034	0.045	0.012	**0.024**	**0.043**	**0.01**
sonar	**0.02**	**0.089**	**0.021**	0.039	0.095	0.023	0.059	0.098	0.016
Immunotherapy	**0.09**	**0.09**	**0.0**	**0.09**	**0.091**	**0.007**	**0.09**	**0.09**	**0.001**
Divorce	**0.0**	**0.025**	**0.01**	**0.0**	**0.021**	**0.009**	**0.0**	**0.014**	**0.011**
wdbc	0.015	0.027	0.008	**0.009**	**0.03**	**0.007**	0.016	0.031	0.007
breast-cancer-wisconsin	**0.032**	**0.033**	**0.003**	**0.032**	**0.033**	**0.001**	**0.032**	**0.032**	**0.0**

In this table, we can see that B-PSA obtains good results compared to the other two metaheuristics and is even better in the sonar instance. This indicates that our proposal is competitive.

Additionally, Table 5 shows the f-score obtained for each instance using the three metaheuristics under study. The first column of this table indicates the instance under study and the second. The third and fourth columns are repeated for each metaheuristic. The first indicates the best f-score obtained; the second indicates the average obtained with the 31 independent runs performed; the third indicates the standard deviation of the 31 runs performed.

Table 5. F-score obtained.

Instance	B-PSA			B-SCA			B-MFO		
	best	avg	std-dev	best	avg	std-dev	best	avg	std-dev
ionosphere	0.971	0.962	0.01	0.971	0.962	0.01	**0.98**	**0.962**	**0.009**
sonar	**0.981**	**0.918**	**0.019**	0.962	0.911	0.022	0.943	0.907	0.015
Immunotherapy	**0.95**	**0.95**	**0.0**	**0.95**	**0.949**	**0.004**	**0.95**	**0.95**	**0.0**
Divorce	**1.0**	**0.974**	**0.011**	**1.0**	**0.978**	**0.01**	**1.0**	**0.985**	**0.012**
wdbc	0.989	0.979	0.006	**0.994**	**0.978**	**0.006**	0.989	0.976	0.006
breast-cancer-wisconsin	**0.961**	**0.96**	**0.003**	**0.961**	**0.961**	**0.0**	**0.961**	**0.961**	**0.0**

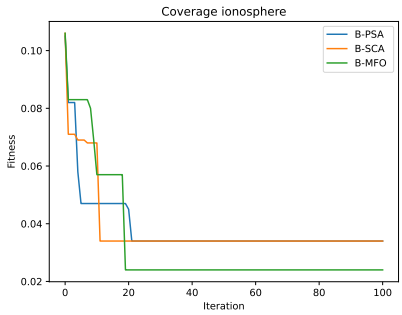
In this table, we can see that B-PSA obtains f-score above 95% in all instances. This indicates that the B-PSA manages to eliminate unnecessary information correctly so that the KNN can perform a very good classification.

Figure 5 shows the convergence graphs of the best executions performed with the three metaheuristics used for each of the six solved instances.

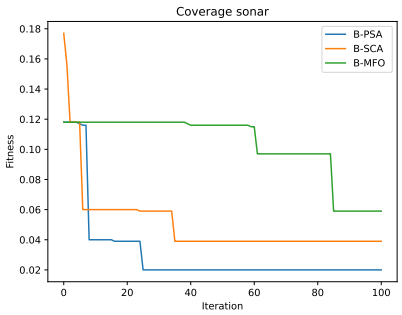
From these Figures, we can observe that B-PSA has a fast convergence in four of the six solved instances, falling in the best value before 50% of the total iterations to be executed are reached. In particular, it is interesting the behavior obtained in the sonar instance (see Figure 5b) where B-PSA has a fast convergence to very good results while B-MFO has a very slow convergence and even very far from the value obtained by B-PSA.

In addition, Figure 6 shows the evolution of the f-score obtained from the best runs performed with the three metaheuristics used for each of the six solved instances. In the

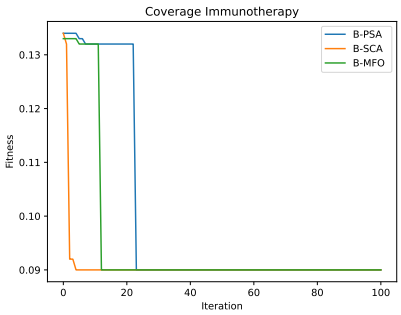
figures, we can observe a behavior similar to those obtained in the convergence graphs, except that for the f-score, we look for a value close to 1 as a good result.



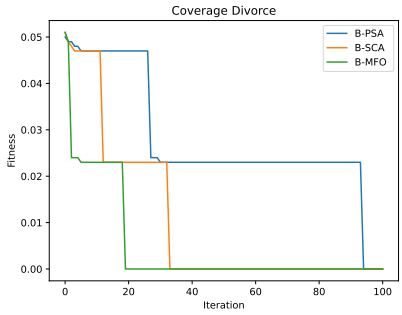
(**a**) ionosphere



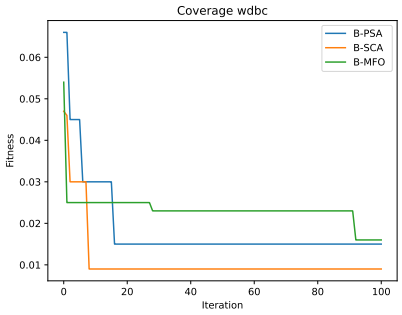
(**b**) sonar



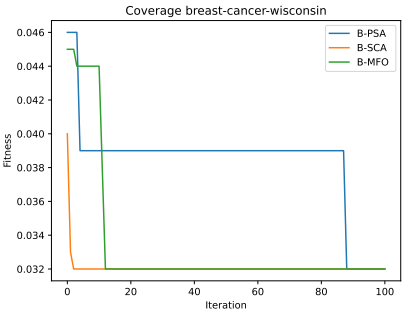
(**c**) Immunotherapy



(**d**) Divorce

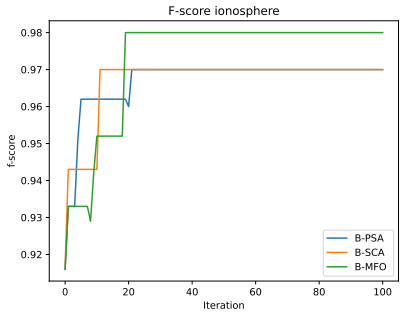


(**e**) wdbc

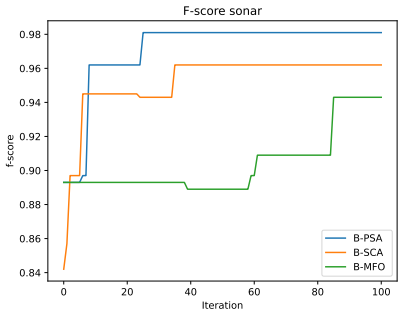


(**f**) breast-cancer-wisconsin

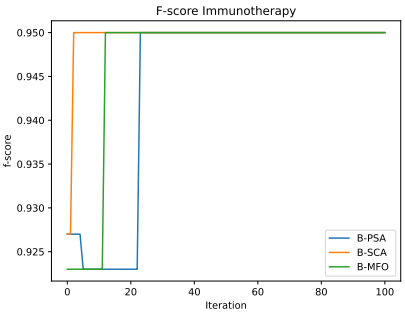
Figure 5. Convergence graph obtained in each instance for each metaheuristic.



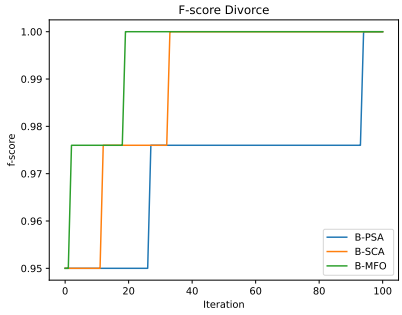
(**a**) ionosphere



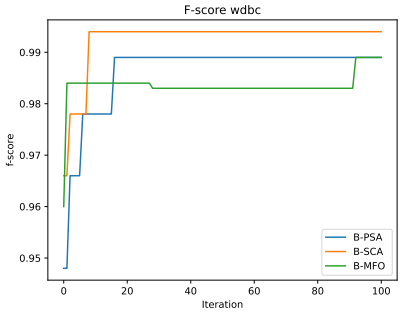
(**b**) sonar



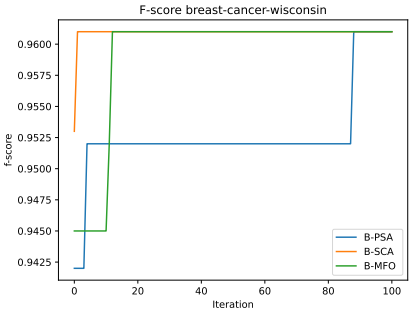
(**c**) Immunotherapy



(**d**) Divorce



(**e**) wdbc

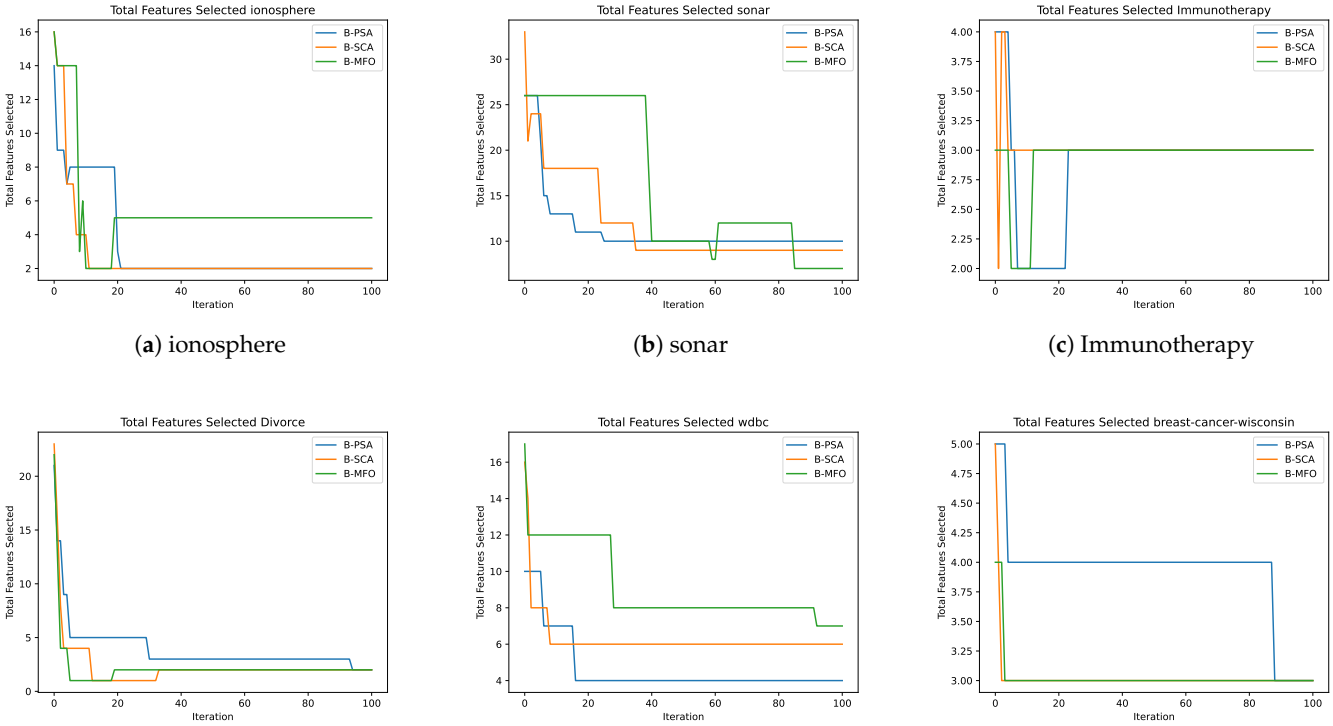


(**f**) breast-cancer-wisconsin

Figure 6. F-score graph obtained in each instance for each metaheuristic.

Finally, Figure 7 shows the evolution of the number of features selected by the best execution of each of the three metaheuristics used for each of the six solved instances. Here, we can see that in four of the six instances, the behavior is similar except for two, which are wdbc and sonar. Regarding the wdbc instance (see Figure 7e), we can observe that B-PSA selects fewer features than the other two metaheuristics, but upon reviewing Table 4, we can see that B-PSA's results are very close to those obtained by B-SCA, which is the metaheuristic that achieves the best results in that instance.

On the other hand, in the sonar instance (see Figure 7b), we can observe that B-PSA is the metaheuristic that selects more features compared to the others. This is interesting as, despite B-PSA selecting more features, it achieves better results, indicating good performance.



(**a**) ionosphere (**b**) sonar (**c**) Immunotherapy

(**d**) Divorce (**e**) wdbc (**f**) breast-cancer-wisconsin

Figure 7. Total Features Selected graph obtained in each instance for each metaheuristic.

7. Conclusions

The Feature Selection Problem has practical applications in the real world in various fields such as health, engineering, telecommunications, or marketing. Given this, it is critical to have highly reliable solutions that deliver high-quality results in reduced time.

In the present work, we demonstrate that the Binary Pendulum Search Algorithm is competitive in solving the Feature Selection Problem, considering both the fitness obtained and the f-score metric. With this, we were able to demonstrate how simple it can be to adapt a continuous metaheuristic to solve binary combinatorial problems. A major advantage of the Binary Pendulum Search Algorithm over the Binary Sine Cosine Algorithm and Binary Moth-Flame Optimization is that it has no configurable parameters. This allows a quick adaptation of the metaheuristic so that it can solve different optimization problems.

However, as indicated by authors in [6], there are different ways to binarize continuous metaheuristics. Such as different transfer functions (U-shaped [36], O-Shaped [37], X-Shaped [38], Z-Shaped [39], time-varying S-Shaped [40] and time-vatying V-Shaped [41]), other binarization rules [33], clustering-based binarization techniques [42] or even binarization supported by machine learning techniques such as Q-Learning [43] or SARSA [44]. The No Free Lunch Theorem [45] tells us that there is no algorithm capable of reaching the

global optimum of all existing optimization problems. Therefore, in future work, another binarization technique could be implemented in order to validate whether the good performance obtained in the present work is maintained. In addition, our proposal could be validated with other datasets and even applied to another combinatorial problem, such as the Set Covering Problem or the Knapsack Problem.

Author Contributions: Conceptualization, B.C. and F.C.-C.; methodology, B.C., F.C.-C. and R.S.; software, F.C.-C. and K.S.; validation, Á.P., A.P., C.L.d.l.B., E.R.-T., G.A., C.C., F.J. and G.G.; formal analysis, B.C., F.C.-C., K.S., R.S.; investigation, B.C., F.C.-C. and R.S.; resources, F.C.-C., K.S., Á.P., A.P., C.L.d.l.B., E.R.-T., G.A. and C.C.; writing—original draft B.C., F.C.-C., K.S. and R.S.; writing—review and editing, Á.P., A.P., C.L.d.l.B., E.R.-T., G.A., C.C., F.J. and G.G.; supervision, B.C., F.C.-C. and R.S.; funding acquisition, B.C. and R.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by: Broderick Crawford, Felipe Cisternas-Caneo, Ricardo Soto, Álex Paz, Alvaro Peña, Claudio León de la Barra, Eduardo Rodriguez-Tello, Gino Astorga, Carlos Castro and Franklin Johnson are supported by grant DI Investigación Asociativa Interdisciplinaria/VINCI/PUCV/039.347/2023. Broderick Crawford and Ricardo Soto are supported by Grant ANID/FONDECYT/REGULAR/1210810. Felipe Cisternas-Caneo is supported by National Agency for Research and Development (ANID)/Scholarship Program/DOCTORADO NACIONAL/2023-21230203.

Data Availability Statement: Data are contained within the article.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Liu, H.; Yu, L. Toward integrating feature selection algorithms for classification and clustering. *IEEE Trans. Knowl. Data Eng.* **2005**, *17*, 491–502. [CrossRef]
2. Agrawal, P.; Abutarboush, H.F.; Ganesh, T.; Mohamed, A.W. Metaheuristic algorithms on feature selection: A survey of one decade of research (2009–2019). *IEEE Access* **2021**, *9*, 26766–26791. [CrossRef]
3. Sadeghian, Z.; Akbari, E.; Nematzadeh, H.; Motameni, H. A review of feature selection methods based on meta-heuristic algorithms. *J. Exp. Theor. Artif. Intell.* **2023**, 1–51. [CrossRef]
4. Rajwar, K.; Deep, K.; Das, S. An exhaustive review of the metaheuristic algorithms for search and optimization: Taxonomy, applications, and open challenges. *Artif. Intell. Rev.* **2023**, *56*, 13187–13257. [CrossRef] [PubMed]
5. Ho, Y.C.; Pepyne, D.L. Simple explanation of the no-free-lunch theorem and its implications. *J. Optim. Theory Appl.* **2002**, *115*, 549–570. [CrossRef]
6. Becerra-Rozas, M.; Lemus-Romani, J.; Cisternas-Caneo, F.; Crawford, B.; Soto, R.; Astorga, G.; Castro, C.; García, J. Continuous Metaheuristics for Binary Optimization Problems: An Updated Systematic Literature Review. *Mathematics* **2022**, *11*, 129. [CrossRef]
7. Pham, T.H.; Raahemi, B. Bio-Inspired Feature Selection Algorithms with Their Applications: A Systematic Literature Review. *IEEE Access* **2023**, *11*, 43733–43758. [CrossRef]
8. Dokeroglu, T.; Deniz, A.; Kiziloz, H.E. A comprehensive survey on recent metaheuristics for feature selection. *Neurocomputing* **2022**, *494*, 269–296. [CrossRef]
9. Abu Khurma, R.; Aljarah, I.; Sharieh, A.; Abd Elaziz, M.; Damaševičius, R.; Krilavičius, T. A Review of the Modification Strategies of the Nature Inspired Algorithms for Feature Selection Problem. *Mathematics* **2022**, *10*, 464. [CrossRef]
10. Emary, E.; Zawbaa, H.M.; Hassanien, A.E. Binary ant lion approaches for feature selection. *Neurocomputing* **2016**, *213*, 54–65. [CrossRef]
11. Wang, M.; Wu, C.; Wang, L.; Xiang, D.; Huang, X. A feature selection approach for hyperspectral image based on modified ant lion optimizer. *Knowl.-Based Syst.* **2019**, *168*, 39–48. [CrossRef]
12. Schiezaro, M.; Pedrini, H. Data feature selection based on artificial bee colony algorithm. *EURASIP J. Image Video Process.* **2013**, *2013*, 47. [CrossRef]
13. Zhang, Y.; Cheng, S.; Shi, Y.; Gong, D.w.; Zhao, X. Cost-sensitive feature selection using two-archive multi-objective artificial bee colony algorithm. *Expert Syst. Appl.* **2019**, *137*, 46–58. [CrossRef]
14. Jeyasingh, S.; Veluchamy, M. Modified bat algorithm for feature selection with the wisconsin diagnosis breast cancer (WDBC) dataset. *Asian Pac. J. Cancer Prev. APJCP* **2017**, *18*, 1257. [PubMed]
15. Taha, A.; Mustapha, A.; Chen, S. Naive bayes-guided bat algorithm for feature selection. *Sci. World J.* **2013**, *2013*, 325973. [CrossRef] [PubMed]
16. Pandey, A.C.; Rajpoot, D.S.; Saraswat, M. Feature selection method based on hybrid data transformation and binary binomial cuckoo search. *J. Ambient. Intell. Humaniz. Comput.* **2020**, *11*, 719–738. [CrossRef]

17. Aziz, M.A.E.; Hassanien, A.E. Modified cuckoo search algorithm with rough sets for feature selection. *Neural Comput. Appl.* **2018**, *29*, 925–934. [CrossRef]
18. Sudha, M.; Selvarajan, S. Feature selection based on enhanced cuckoo search for breast cancer classification in mammogram image. *Circuits Syst.* **2016**, *7*, 327–338. [CrossRef]
19. Al-Tashi, Q.; Kadir, S.J.A.; Rais, H.M.; Mirjalili, S.; Alhussian, H. Binary optimization using hybrid grey wolf optimization for feature selection. *IEEE Access* **2019**, *7*, 39496–39508. [CrossRef]
20. Tu, Q.; Chen, X.; Liu, X. Multi-strategy ensemble grey wolf optimizer and its application to feature selection. *Appl. Soft Comput.* **2019**, *76*, 16–30. [CrossRef]
21. Canayaz, M. MH-COVIDNet: Diagnosis of COVID-19 using deep neural networks and meta-heuristic-based feature selection on X-ray images. *Biomed. Signal Process. Control.* **2021**, *64*, 102257. [CrossRef] [PubMed]
22. Medjahed, S.A.; Saadi, T.A.; Benyettou, A.; Ouali, M. Kernel-based learning and feature selection analysis for cancer diagnosis. *Appl. Soft Comput.* **2017**, *51*, 39–48. [CrossRef]
23. Zhang, Y.; Liu, R.; Wang, X.; Chen, H.; Li, C. Boosted binary Harris hawks optimizer and feature selection. *Eng. Comput.* **2021**, *37*, 3741–3770. [CrossRef]
24. Too, J.; Abdullah, A.R.; Mohd Saad, N. A new quadratic binary harris hawk optimization for feature selection. *Electronics* **2019**, *8*, 1130. [CrossRef]
25. Nadimi-Shahraki, M.H.; Banaie-Dezfouli, M.; Zamani, H.; Taghian, S.; Mirjalili, S. B-MFO: A binary moth-flame optimization for feature selection from medical datasets. *Computers* **2021**, *10*, 136. [CrossRef]
26. Kumar, L.; Bharti, K.K. An improved BPSO algorithm for feature selection. In Proceedings of the Recent Trends in Communication, Computing, and Electronics: Select Proceedings of IC3E 2018, Langkawi, Malaysia, 21–22 November 2018; Springer: Singapore, 2019; pp. 505–513.
27. Sayed, G.I.; Hassanien, A.E.; Azar, A.T. Feature selection via a novel chaotic crow search algorithm. *Neural Comput. Appl.* **2019**, *31*, 171–188. [CrossRef]
28. Hafez, A.I.; Zawbaa, H.M.; Emary, E.; Hassanien, A.E. Sine cosine optimization algorithm for feature selection. In Proceedings of the 2016 International Symposium on Innovations in Intelligent Systems and Applications (INISTA), Sinaia, Romania, 2–5 August 2016; pp. 1–5.
29. Sindhu, R.; Ngadiran, R.; Yacob, Y.M.; Zahri, N.A.H.; Hariharan, M. Sine–cosine algorithm for feature selection with elitism strategy and new updating mechanism. *Neural Comput. Appl.* **2017**, *28*, 2947–2958. [CrossRef]
30. Zivkovic, M.; Jovanovic, L.; Ivanovic, M.; Krdzic, A.; Bacanin, N.; Strumberger, I. Feature selection using modified sine cosine algorithm with COVID-19 dataset. In Proceedings of the Evolutionary Computing and Mobile Sustainable Networks: Proceedings of ICECMSN 2021, Bengaluru, Karnataka, 28–29 September 2021; Springer: Singapore, 2022; pp. 15–31.
31. Mafarja, M.M.; Mirjalili, S. Hybrid whale optimization algorithm with simulated annealing for feature selection. *Neurocomputing* **2017**, *260*, 302–312. [CrossRef]
32. Hussien, A.G.; Hassanien, A.E.; Houssein, E.H.; Bhattacharyya, S.; Amin, M. S-shaped binary whale optimization algorithm for feature selection. In *Recent Trends in Signal and Image Processing: ISSIP 2017*; Springer: Singapore, 2019; pp. 79–87.
33. Crawford, B.; Soto, R.; Astorga, G.; García, J.; Castro, C.; Paredes, F. Putting continuous metaheuristics to work in binary search spaces. *Complexity* **2017**, *2017*, 8404231. [CrossRef]
34. Ab. Aziz, N.A.; Ab. Aziz, K. Pendulum Search Algorithm: An Optimization Algorithm Based on Simple Harmonic Motion and Its Application for a Vaccine Distribution Problem. *Algorithms* **2022**, *15*, 214. [CrossRef]
35. Ab Rahman, T.; Ibrahim, Z.; Ab Aziz, N.A.; Zhao, S.; Abdul Aziz, N.H. Single-Agent Finite Impulse Response Optimizer for Numerical Optimization Problems. *IEEE Access* **2018**, *6*, 9358–9374. [CrossRef]
36. Beheshti, Z. UTF: Upgrade transfer function for binary meta-heuristic algorithms. *Appl. Soft Comput.* **2021**, *106*, 107346. [CrossRef]
37. Durgut, R.; Aydin, M.E. Adaptive binary artificial bee colony algorithm. *Appl. Soft Comput.* **2021**, *101*, 107054. [CrossRef]
38. Ghosh, K.K.; Singh, P.K.; Hong, J.; Geem, Z.W.; Sarkar, R. Binary social mimic optimization algorithm with x-shaped transfer function for feature selection. *IEEE Access* **2020**, *8*, 97890–97906. [CrossRef]
39. Sun, W.Z.; Zhang, M.; Wang, J.S.; Guo, S.S.; Wang, M.; Hao, W.K. Binary Particle Swarm Optimization Algorithm Based on Z-shaped Probability Transfer Function to Solve 0-1 Knapsack Problem. *IAENG Int. J. Comput. Sci.* **2021**, *48*, 294–303.
40. Kahya, M.A.; Altamir, S.A.; Algamal, Z.Y. Improving whale optimization algorithm for feature selection with a time-varying transfer function. *Numer. Algebr. Control. Optim.* **2021**, *11*, 87. [CrossRef]
41. Chantar, H.; Thaher, T.; Turabieh, H.; Mafarja, M.; Sheta, A. BHHO-TVS: A binary harris hawks optimizer with time-varying scheme for solving data classification problems. *Appl. Sci.* **2021**, *11*, 6516. [CrossRef]
42. García, J.; Lemus-Romani, J.; Altimiras, F.; Crawford, B.; Soto, R.; Becerra-Rozas, M.; Moraga, P.; Becerra, A.P.; Fritz, A.P.; Rubio, J.M.; et al. A binary machine learning cuckoo search algorithm improved by a local search operator for the set-union knapsack problem. *Mathematics* **2021**, *9*, 2611. [CrossRef]
43. Crawford, B.; Soto, R.; Lemus-Romani, J.; Becerra-Rozas, M.; Lanza-Gutiérrez, J.M.; Caballé, N.; Castillo, M.; Tapia, D.; Cisternas-Caneo, F.; García, J.; et al. Q-learnheuristics: Towards data-driven balanced metaheuristics. *Mathematics* **2021**, *9*, 1839. [CrossRef]

44. Lemus-Romani, J.; Becerra-Rozas, M.; Crawford, B.; Soto, R.; Cisternas-Caneo, F.; Vega, E.; Castillo, M.; Tapia, D.; Astorga, G.; Palma, W.; et al. A novel learning-based binarization scheme selector for swarm algorithms solving combinatorial problems. *Mathematics* **2021**, *9*, 2887. [CrossRef]
45. Wolpert, D.H.; Macready, W.G. No free lunch theorems for optimization. *IEEE Trans. Evol. Comput.* **1997**, *1*, 67–82. [CrossRef]