International Journal of Machine Learning and Cybernetics (2021) 12:303–342
https://doi.org/10.1007/s13042-020-01174-8

ORIGINAL ARTICLE

Feature selection schema based on game theory and biology migration algorithm for regression problems

Mohammad Masoud Javidi[1]

Received: 23 September 2019 / Accepted: 1 August 2020 / Published online: 12 August 2020

Abstract

Many real-world datasets nowadays are of regression type, while only a few dimensionality reduction methods have been developed for regression problems. On the other hand, most existing regression methods are based on the computation of the covariance matrix, rendering them inefficient in the reduction process. Therefore, a BMA-based multi-objective feature selection method, GBMA, is introduced by incorporating the Nash equilibrium approach. GBMA is intended to maximize model accuracy and minimize the number of features through a less complex procedure. The proposed method is composed of four steps. The first step involves defining three players, each of which is trying to improve its objective function (i.e., model error, number of features, and precision adjustment). The second step includes clustering features based on the correlation therebetween and detecting the most appropriate ordering of features to enhance cluster efficiency. The third step comprises extracting a new feature from each cluster based on various weighting methods (i.e., moderate, strict, and hybrid). Finally, the fourth step encompasses updating players based on stochastic search operators. The proposed GBMA strategy explores the search space and finds optimal solutions in an acceptable amount of time without examining every possible solution. The experimental results and statistical tests based on ten well-known datasets from the UCI repository proved the high performance of GBMA in selecting features for solving regression problems.

Keywords Feature selection · Nash equilibrium · Multi-objective optimization · Biology migration algorithm · Game theory

1 Introduction

Datasets are almost constantly generated in different areas, such as industry, social networking, and business. They can be represented in the form of networks to gain insights from many perspectives, including patterns of information diffusion. Nowadays, a lot of attention is being paid to the study of network analysis technologies because of their considerable impact on many fields, such as classification [1], clustering [2], and link prediction [3]. For example, sociologists classify individuals into various social media groups [4], and biologists predict protein–protein links in interaction networks and discover missing relations between proteins [5]. However, real-world networks have grown to such a large-scale, so researchers are trying to embed them into a low-dimensional space while preserving the useful features [6]. Shi et al. [7, 8] proposed two network embedding (NE) procedures for node classification and a multi-label network embedding (MLNE) to learn feature representations where each node contains some feature values and a set of class labels. These large datasets indeed contain irrelevant, redundant, or erroneous features. Computational burden, increasing storage requirements, overfitting, decreasing prediction performance, and poor output comprehensibility are some of the disadvantages engendered by applying large-scale datasets with misleading features. Dimensionality reduction is a crucial preprocessing step in the analysis of high-dimensional datasets, generally including two strategies: feature extraction and feature selection [8]. Feature extraction tries to define new features by transforming or combining the original features. Two conventional feature extraction

Electronic supplementary material The online version of this article (https://doi.org/10.1007/s13042-020-01174-8) contains supplementary material, which is available to authorized users.

✉ Mohammad Masoud Javidi
javidi@uk.ac.ir

[1] Department of Computer Science, Shahid Bahonar University of Kerman, Kerman, Iran

techniques are principal component analysis (PCA) and independent component analysis (ICA) [9]. Feature selection tries to eliminate redundant or unnecessary features and selects only those features that are sufficient for solving a problem. Hence, it makes the model simpler and comprehensible by removing insignificant features.

However, finding the best-performing feature subset is a complicated process, especially for a high-dimensional problem due to the large search space. In the last few decades, many traditional feature selection algorithms like sequential forward selection (SFS) [10] and sequential backward selection (SBS) [11] have been utilized to provide promising results in dealing with feature selection problems. Nevertheless, they suffer several issues, such as computational complexity [12] and nesting effects [13]. We can consider the feature selection challenge as a multi-objective optimization (MOO) problem. There is a trade-off between some objectives (e.g., the minimum number of features and minimum error value); thus, optimization techniques can be used to strike the right balance between objectives. In some datasets, the error rate of the classifier can be reduced by removing irrelevant and redundant features.

Nonetheless, if a dataset only consists of key features required by the classifier, removing any feature may reduce the accuracy of this classifier and so the two objectives conflict with each other. Notably, investigating the importance of different features requires various levels of costs like time, storage, or other resources. That is, too many features entail higher costs.

Today, the metaheuristic method is becoming increasingly popular in feature selection and machine intelligence [14, 15]. Several metaheuristic based feature selection strategies have been developed over the years. The basic process of such methods is to optimize an objective function that is defined for feature selection to find the optimal feature subset. The metaheuristic-based feature selection algorithms randomly generate new solutions and compute their fitness values. In subsequent iterations, new solutions are created by the best agents of the current iteration. Hence, these methods prevent the creation of a solution similar to the previous one, thereby reducing the computational time in determining the best subset of features. A majority of feature selection methods have been developed to handle classification problems, and there has been less focus on regression problems [16]. Regression involves predicting a real value of output parameter (i.e., continuous target variable) for the given values of other parameters, which is based on previous input–output observations. The focus of this study is to adopt Biology Migration Algorithm (BMA) in a feature selection method to solve regression problems. The experimental results in [17] with several test functions and four real-life engineering problems proved that BMA is more effective compared with other related optimization techniques. These include Whale Optimization Algorithm (WOA) [18], Bat Algorithm (BA) [19], Animal Migration Optimization (AMO) [20], Cuckoo Search (CS) [21], Gravitational Search Algorithm (GSA) [22], Biogeography-Based Optimization (BBO) [23], and Particle Swarm Optimization (PSO) [24].

The proposed strategy applies BMA for solving the dimensionality reduction problem to minimize the number of features and errors. It also introduces a new approach for feature clustering based on the correlation between features and game theory. Game theory is a branch of modern mathematics that has recently received much attention from researchers in the study of cross-discipline [25]. It quantifies interactions among incentive structures or various players for strategic interdependence. When there is more than one agent who independently performs their own behavior in the process, game theory can present a suitable framework for the analysis of interactions between these decisions [26]. In this paper, we apply game theory to deal with the conflict and competition among multiple objectives in the feature selection problem.

The remainder of the paper is structured as follows. Section 2 briefly reviews some related works. Section 3 lists the main contributions of the paper. Section 4 illustrates basic concepts about multi-objective optimization and the Nash heuristic method. Section 5 presents the proposed dimensionality reduction strategy. Section 6 focuses on reducing the computational complexity of the proposed algorithm. Section 7 reports the experimental results based on UCI datasets and some discussions. Section 8 concludes the paper and presents future work.

2 Related works

Although much research has been conducted on dimensionality reduction, regression problems are rarely studied compared with classification problems. One of the reasons may be a simple formulation for feature selection criteria like the broad margin framework with considering class discriminability [27, 28]. Some feature selection algorithms for classification can be extended for regression problems, while others may not. Straightforward adaptation by discretizing (or binning) the output variable into several classes is not always a suitable solution since this can lead to the loss of important information [16]. Moreover, most feature clustering techniques compute the conditional probability of a feature belonging to a particular class. Thus such probabilities cannot be defined for the regression problem.

Consequently, these algorithms cannot be used directly for reducing dimensions of regression problems. On the other hand, most existing methods of regression are based on the computation of the covariance matrix, eigenvalues, and eigenvectors, rendering them inefficient for large problems. For example, the mRmR [29], relief [30], and LDA [31] methods can be used for classification and with some changes for regression problems. These methods are explained in detail as follows.

- *Linear discriminant analysis* (*LDA*): It is originally introduced for supervised learning, especially classification problems, applicable for finding the optimal linear discriminant functions (OLDFs) [31]. LDA attempts to provide linear combinations of features in such a way that the ratio of between-class scatter and within-class scatter be maximized. Unlike classification problems that have discrete output or classes, it is challenging to consider between- and within-class scatter matrices in regression applications because the continuous target variable is defined. One solution for presenting a regressional version of LDA is that we segment the given dataset into several intervals (i.e., virtual classes) according to the output values with the fixed boundaries. The original LDA for classification problems can now be used, yet the results will depend on the number of intervals and the boundary approach. Moreover, this modified LDA strategy does not consider the degrees of similarity among various classes. Kwak and Lee [32] proposed a regressional version of LDA and extended ICA-FX for regression problems [33].
- *Relief*: The original version of the Relief method [30] is used only for binary problems. It ranks features according to the determination of feature value differences between the nearest neighbor example pairs. Then, Sikonja and Kononenko [34] proposed a new modified version (ReliefF) for regression problems. Nevertheless, the nearest cannot be used for regression problems because the predicted variable (i.e., class) is continuous. Therefore, Relief-F [35] for regression is introduced. It defines a probability based on the relative distance between the predicted and actual values to determine whether or not the predicted values of two objects are different.
- *Minimum redundancy, maximum relevance* (*mRmR*): The mRmR method [29] tries to choose some features that show the highest correlation with a class, and there is the least correlation between themselves. Obviously, the relevance and redundancy measures for classification and regression must be different. For the calculation of relevance, "F-statistic" and "mutual information (MI)" are suitable for continuous features and discrete features, respectively. To calculate redundancy, Pearson's correlation coefficient and MI are applied for continuous features and discrete features, respectively [36].

Now, we present a brief literature review on the new feature selection and feature clustering approaches. Xu and Lee [16] developed a feature clustering method to reduce the dimensionality of regression problems. The proposed algorithm creates a group of clusters based on similarity tests such that similar items fall into the same cluster. Then, it constructs a new feature from each cluster using three weighting methods, namely hard, soft, and mixed, which are based on the weighted combination of training samples. One strength of this method is that the number of clusters is automatically obtained during the process without requiring to be adjusted by the user. By providing the experimental result, they demonstrated that the proposed algorithm could significantly reduce dimensionality while the main properties of the primary dataset are retained. The main disadvantage of this method is that it shows the high computational complexity for high-dimensional datasets.

Rao et al. [37] presented a feature selection algorithm by using the bee colony and gradient boosting decision tree. The proposed algorithm preprocesses and reduces the irrelevant features of the original dataset by the bee colony. Then, it tries to forcibly reduce the initial feature set and create a decision tree for determining the weight of features and supporting the accuracy of the model. The experiments indicated that the proposed algorithm could reduce dimensionality without sacrificing classification accuracy. However, the employment of a decision tree in the weighting process is time-consuming and computationally expensive.

Zhang et al. [38] modified the firefly optimization algorithm for feature selection. In population initialization, it uses a chaotic logistic map to increase swarm diversity. In the search process, it applies Simulated Annealing (SA) and optimal global signals. To improve convergence, the proposed algorithm diverts weak solutions to optimal regions by swarm leaders. The experiments based on several classification and regression problems indicated that the proposed firefly model outperforms other classical methods. However, they did not consider outlier detection.

Ghimatgar et al. [39] proposed a new feature selection strategy, i.e., Modified Graph Clustering based Ant Colony Optimization (MGCACO), according to the graph clustering and ant colony optimization. The MGCACO algorithm consists of three main steps. The first step involves investigating the relevance of features to classes to improving pheromone initialization and assigning a higher priority to the more relevant features. The second step includes finding the redundancy of feature subsets based on the multiple discriminant analysis (MDA). Finally, the third step comprises arranging features according to relevance and redundancy analysis by applying a cost function. The experimental results indicated the superior performance of MGCACO in terms of accuracy. However, the main weakness of MGCACO is that there is no specific way of determining the number of clusters.

Table 1 compares general characteristics, main idea, and limitations of the methods reviewed previously. Most

Table 1 Summary of dimension reduction methods

References	[16]	[37]	[38]	[39]
Year	2015	2019	2018	2018
Classification/regression	Regression	Regression	Classification/regression	Classification
Datasets	4 UCI datasets	8 UCI datasets	33 UCI datasets	7 UCI datasets
Single/multi-objectives	Single objective	Single objective	Multi-objective	Single objective
Feature selection/extraction	Feature selection and extraction	Feature selection	Feature selection	Feature selection
Main idea	Design three feature weighting approaches	Apply gradient boosting and decision tree	Utilize a chaotic map function and update agents' positions with SA and firefly algorithm	Apply graph clustering and ant colony optimization
Weakness	(1) High computational complexity (2) Avoiding the order of features during the clustering process	(1) The artificial bee colony has some drawbacks such as: (I) lack of use of secondary information; (II) The population of solutions increases the computational cost (2) Boosting approaches are from empirical evidence and particularly vulnerable to noises	(1) Lack of method for outliers detection (2) High computational complexity (3) Sensitive to noise	(1) Consume a lot of time to generate clusters (2) Lack of method for outliers detection (3) Filter_based approach may increase the error rate of classifier

dimensionality reduction techniques presented in the literature are feature selection methods, and a few feature extraction approaches are considered. Based on theoretical and empirical results [40], traditional algorithms suffer from three problems. Firstly, they have problems with overhead memory space for large datasets and discovering the relationship between features that significantly affect the efficiency and accuracy of the model. The second drawback of previous feature selection methods is that they are easily get trapped into local optima during search processes. Therefore, these behaviors cause substantial growth in feature selection delay in exploiting the correlation among lots of relevant features. Thirdly, traditional algorithms may not be efficient and robust for high-dimensional heterogeneous datasets with low quality and noisy points. Nevertheless, robustness is an essential characteristic of satisfactory feature selection methods (SFSMs) because the presence of noise in big data is inevitable, and the processing time may increase by high noise levels. Consequently, an effective feature selection algorithm is necessary to overcome several issues in existing algorithms, such as problems of algorithm implementation, convergence, and computational complexity for regression applications.

Although there has been a lot of research on feature selection based on metaheuristics as well as simultaneous feature selection and clustering, little attention has been paid to using game theory in feature selection. The main reason is that metaheuristic techniques have performed remarkably in solving complex problems, thanks to flexibility, faster convergence by focusing on intensification, and local optima avoidance by emphasizing diversification. Furthermore, game theory can help determine an efficient solution by analyzing conflict and cooperation among various objectives. To the best of our knowledge, most game-theoretic feature selection algorithms are proposed for use in classification problems. Consequently, it would be meaningful to investigate the issue of metaheuristic based game theoretic feature selection for identifying the best subset of features, substantially affecting target prediction. In this paper, we propose a new game theory-based dimensionality reduction algorithm to deal with conflict and competition among multiple objectives of data regression. Thanks to its versatility and robustness, game theory has been applied to solve some complex problems in different research areas like task scheduling and resource management [41]. Moreover, algorithms, according to game theory, show low computational complexity and high computational speed.

3 Contributions

Below is a list of some issues raised in Table 1 that require further attention:

- Feature selection strategies presented in the literature have been focused on classification problems, and these algorithms cannot be used quickly and efficiently to reduce the dimensionality of regression problems. However, the proposed algorithm can be used for both regression and classification problems with a small modification.

- All studies have referred to the multi-objective nature of the feature selection problem, though they have indeed dealt with it as single-objective. To address this issue, the proposed algorithm solves the feature selection problem to realize three objectives. These are minimizing the regression error rate, minimizing feature cardinality, and adjusting method precision. For this purpose, it takes advantage of BMA, which is a new metaheuristic method inspired by the biology migration phenomenon.
- The sequence of features, weighting methods for feature extraction, and fitness values are used as heuristic information for the BMA method. Less important features are suppressed based on the information they contribute to decision-making using the BMA algorithm.
- In the proposed algorithm, a coalitional game occurs between features to generate appropriate clusters. Put differently, features are considered as players, each of which wants to join a group of features and generate a cluster with high correlation among players (i.e., the correlation coefficient is considered as a coalitional function). The majority of previous strategies have user-defined parameters (e.g., number of clusters and threshold value). However, in the proposed method, clusters are generated automatically without determining the number of clusters by the user. Furthermore, three populations are defined as players to play a non-cooperative game, and each of which aims to improve its own fitness function (i.e., model error, number of features, and Mallows' Cp).
- Most feature selection strategies are extremely time-consuming, rendering them inefficient for large problems. Therefore, the proposed method uses MapReduce and space reduction techniques to reduce time complexity. It is also used to perform preprocessing (i.e., handling missing values and detecting outliers) to improve data quality and increase overall productivity.
- We have performed extensive experiments based on ten regression datasets. The proposed algorithm (i.e., GBMA) is compared with conventional state-of-the-art methods in two parts. Firstly, various swarm-based techniques, including MGWO, ABCoDT, and CPSOS, are evaluated. Secondly, the non-swarm intelligence-based techniques, including well-known feature extraction algorithms (e.g., LDAr, WPCA, and FC-C-S), are compared. Moreover, a statistical analysis validates the efficacy and superiority of the proposed method.

4 Preliminaries

In this section, we briefly review the basic information about BMA,□ MOO, and Nash equilibria that are used in the proposed algorithm (i.e., GBMA).

4.1 Biology Migration Algorithm (BMA)

One of the major techniques for solving complex optimization problems is nature-inspired metaheuristic algorithms. Zhang et al. [17] proposed a metaheuristic algorithm called Biology Migration Algorithm (BMA) that is based on the biology migration behavior in nature. BMA consists of three steps as follows:

- Population generation: This step models the movement of different biotic species to a new location. Population P with N search agents is randomly generated from the problem space [17].
- Migration phase: It contains two rules: (1) The positions of agents are modified based on the best position of the solution space, and (2) The agent moves to a new location according to its neighborhood solutions.
- Updating phase: If a solution cannot be improved during a pre-defined number of cycles (Max_C), it will be replaced by a new one in the solution space. The pseudocode of this phase is presented by Algorithm 1 [17], where X_i (t) indicates the i-th search agent of the population at the t-th iteration.

Algorithm 1. Updating phase

```
Evaluate $X_i (t+1)$;
if $X_i (t+1)$ is better than $X_i (t)$
 {
   $X_i (t) = X_i (t+1)$ ;
   $Cycle (i) = 0$;
 }
else
 {
    $Cycle (i) = Cycle (i+1)$;
    if ($Cycle (i) \geq Max\_C$) then
       $X_i (t)$= randomly initialize agent;
 }
```

4.2 Multi-Objective Optimization (MOO)

Generally, MOO is defined as finding a vector of variables to minimize (or maximize) a vector of objective functions in a feasible region. Thus, it can be modeled as Eq. (1) [42]:

$$Minimize \;\; F(x) = [f_1(x), f_2(x), \ldots, f_d(x)], \quad Subject\,to \quad q(x) \leq 0, \tag{1}$$

where x indicates the vector of variables, $f_d(x)$ shows the d-th objective function, and $q(x)$ indicates the constraint vector.

One of the main challenges in MOO is the conflicting objectives (i.e., a trade-off between objectives). In other words, one objective is achieved only by making concessions to another objective; thus, the optimal solution cannot be found for all m objective functions simultaneously. Consequently, a "compromise solution" is acceptable for these problems.

One of the primary ways to deal with the multi-objective problem is to consider scalar objectives and define the linear combination of weighted objectives [43]. However, this method will be applicable and efficient once the weights can be determined correctly. This method also has some weaknesses, such as sensitivity to weights and sometimes engendering the loss of information [43]. Two obvious contradictory objectives in the feature selection problem are minimizing the number of features and minimizing the error rate. The feature selection process attempts to select a few significant features to obtain similar or even better regression performance, rather than considering all features. Hence, we consider feature selection as a bi-objective minimization problem. In this paper, we apply a non-cooperative schema with the notion of a player to solve the multi-objective feature selection problem.

4.3 Nash heuristic method

In 1950, Nash introduced the fundamental concept of determining equilibria in cases where several competing players seek to optimize their objective functions [43]. If an optimization problem consists of d objectives, the Nash strategy will define d players, each of which with its own optimization criterion. Each player seeks to optimize its own criterion by assuming that the criteria of the other players are fixed. When none of the players can improve their criteria, the system reaches a state of equilibrium called the Nash equilibrium [39]. The idea of combining Nash with heuristic methods is to bring together a heuristic method and the Nash strategy to find the Nash equilibrium as a solution for MOO problems.

5 The proposed method

This paper proposes a novel dimensionality reduction technique based on BMA and game strategies, i.e., GBMA. GBMA is hybridized with two specific game strategies, i.e., the Nash strategy and coalitional game, to find the best solution and solve the multi-objective feature selection problem. Firstly, the preprocessing steps are taken to transform the original dataset into a format that is easily and more effectively applicable to future processing phases. The main processes taking place here handle the missing values and detect outliers [44, 45]. Using interquartile range (IQR) [43], we attempted to detect and remove outliers to provide a model that fits well the reality and undeviated by outliers. IQR indicates how the data spreads about the median and is used for outlier detection. In other words, outliers are defined as objects that fall below $Q1-1.5\ IQR$ or above $Q3+1.5\ IQR$ [46]. Handling the missing values is an essential step during model development since they affect conclusions derived from the data. Missing data can be removed when it is limited to a small number of objects, followed by processing the remaining objects. This method is called listwise deletion and considered as a default option in most statistical software packages [47].

5.1 GBMA framework

Figure 1 shows the framework of the proposed method that consists of four major steps as follows:

- In the first step, the population of agents is initialized as players (see Sect. 5.2).
- In the second step, the features corresponding to each agent are clustered based on their correlation coefficients. That is, these features are considered as players that play a coalitional game, each of which desires to join the features that increase the correlation coefficient of extracted features (see Sect. 5.3).
- In the third step, the new features are extracted from each cluster based on the weighting methods (see Sect. 5.4).
- In the fourth step, the best agent of each population is determined, followed by updating the agents of two populations (see Sect. 5.5).

Moreover, steps 2–4 are repeated until the stopping criterion is satisfied; otherwise, the algorithm is stopped, and the best feature set is reported.

5.2 Definition of populations as players

This section initially defines the fitness function and then introduces the agent representation. For each objective in the fitness function, players (i.e., populations) are defined that participate in a game, each of which attempting to improve their objective. This game is continued until none of the players can improve its objective.

5.2.1 Fitness function

The proposed algorithm deals with the feature selection problem as a MOO problem. We attempt to optimize three different objectives with three players by computing the population of agents for each criterion. The fitness function is calculated by Eq. (2):

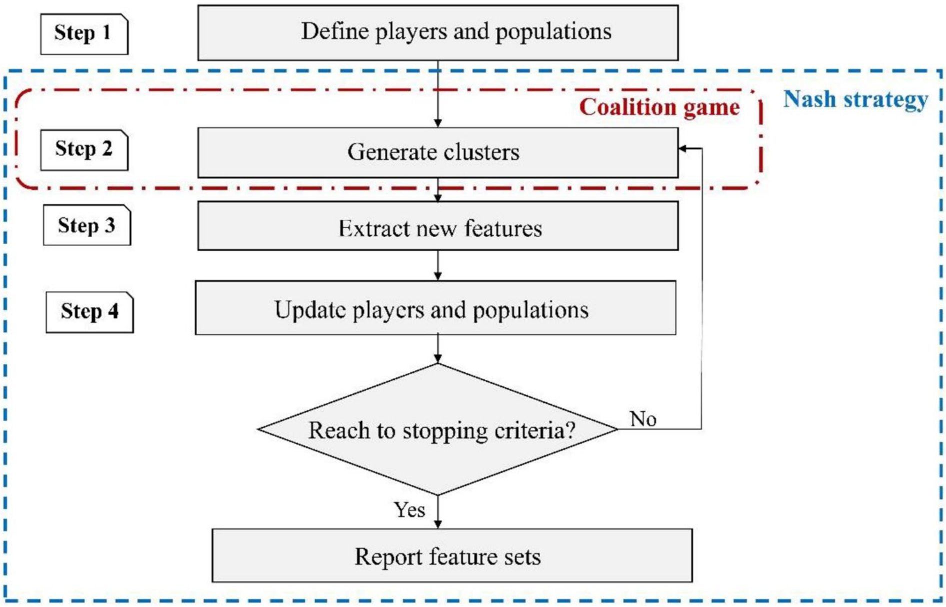


Fig. 1 The framework of GBMA

$$fitness = W_r \times Er(R) + W_f \times \frac{|L|}{|T|} + W_p \times \left(\frac{SSE_p}{MSE_{FULL}} - All\_s + 2h \right), \tag{2}$$

where $Er(R)$ is the regression error rate, $|L|$ indicates the length of the feature, $|T|$ is the total number of features, MSE_{FULL} is the MSE for the full model (i.e., the model containing all features). SSE_h indicates the residual sum of squares (RSS) for the subset model containing h features (i.e., h is the number of features after the feature selection process). Moreover, All_s is the sample size. In Eq. (2), W_r, W_f, and W_p are three parameters corresponding to the importance of regression quality, subset length, and model precision, respectively, where $W_r,\ W_f,\ W_p \in [0, 1][0, 1]$.

Figure 2 presents the attempts made to optimize three different objectives participating in fitness calculation (Eq. 2) and how such merging can be achieved with three players. We show the candidate solution for a triple objective function by Eq. (3):

$$S = XX'X'' \tag{3}$$

where X X', and X'' show the first, second, and third criteria.

The optimization processes of X,X', and X'' are assigned to $Player_1$, $Player_2$, and *Player3*, respectively. Based on the Nash theory, $Player_1$ tries to optimize S according to the first criterion by changing X, while X' and X'' are fixed by $Player_2$ and $Player_3$, respectively. Similarly, $Player_2$ seeks to optimize S according to the second criterion by changing X', while X and X'' are fixed by $Player_1$ and $Player_3$, respectively. In this paper, a population is created for each player (i.e., three populations). *Player* 1's optimization process is carried out by the first population, whereas $Player_2$'s and $Player_3$'s optimization processes are performed by the second and third populations, respectively.

Let X_{k-1} be the best value found by $Player_1$ at generation $k-1$, X'_{k-1} and X''_{k-1} be the best values found by $Player_2$ and $Player_3$, respectively, at generation $k-1$. At generation k, $Player_1$ optimizes X using X'_{k-1} and X''_{k-1} to evaluate S (i.e., $S = X_k X'_{k-1}$). Likewise, $Player_2$ optimizes X'_k using X_{k-1} and X''_{k-1} (i.e., $S = X_{k-1} X'_k X''_{k-1}$).

After the optimization process, $Player_1$ forwards the best value X_k to $Player_2$ and $Player_3$, which will apply it at generation $k+1$. Likewise, $Player_2$ forwards the best value X'_k to $Player_1$ and $Player_3$, which will apply it at generation $k+1$. This process is repeated for X''_k as the best value for $Player_3$. Nash equilibrium is reached when the three players (i.e., $Player_1$, $Player_2$, $Player_3$) cannot improve their criteria.

5.2.2 Agent representation

In our schema, each agent in a population consists of two parts. The first part is named *Sequencing,* whose size is equal to the number of features in a dataset. Each agent represents the order of features affecting clustering. The second part determines the weighting method used to extract new features. Each cell of agents in the first part consists of a constant value between [*0*, *M*1], where *M*1 shows the total number of features. In the second part, each cell consists of a constant value between [*0*, *M*2], where *M*2 indicates the total number of weighting methods. Therefore, the problem dimension in this paper is equal to *M*, which is the sum of *M*1 and *M*2. Each cell is rounded to the next highest integer, after which the index of intended features and weighting method is determined.

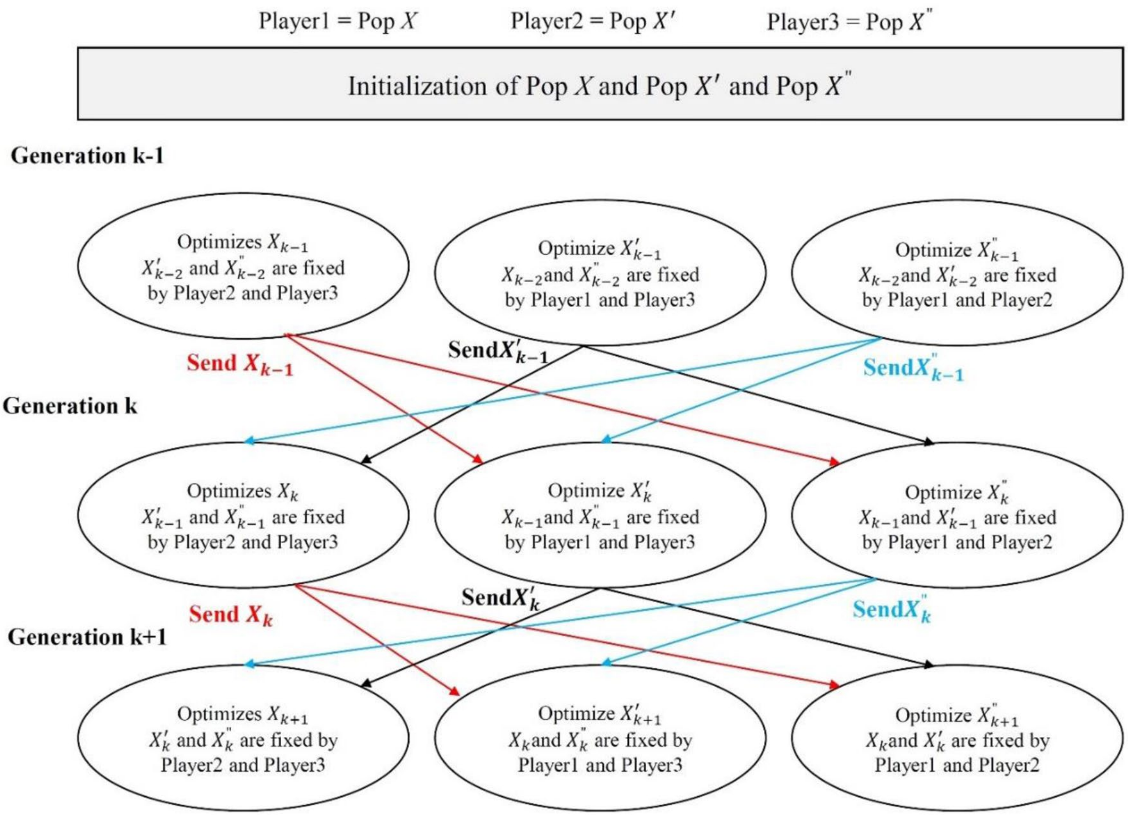
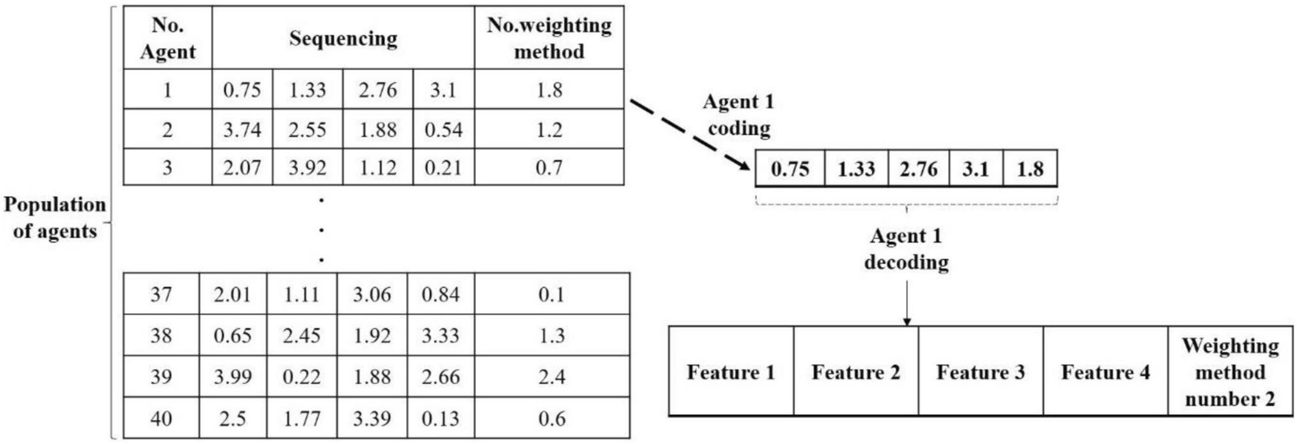

Player1 = Pop X
Player2 = Pop X'
Player3 = Pop X''
Initialization of Pop X and Pop X' and Pop X''
Generation k-1
Optimizes X_{k-1} X'_{k-2} and X''_{k-2} are fixed by Player2 and Player3
Optimize X'_{k-1} $X_{k-2}$and X''_{k-2} are fixed by Player1 and Player3
Optimize X''_{k-1} $X_{k-2}$and X'_{k-2} are fixed by Player1 and Player2
Send X_{k-1}
SendX'_{k-1}
SendX''_{k-1}
Generation k
Optimizes X_k X'_{k-1} and X''_{k-1} are fixed by Player2 and Player3
Optimize X'_k $X_{k-1}$and X''_{k-1} are fixed by Player1 and Player3
Optimize X''_k $X_{k-1}$and X'_{k-1} are fixed by Player1 and Player2
Send X_k
SendX'_k
SendX''_k
Generation k+1
Optimizes X_{k+1} X'_k and X''_k are fixed by Player2 and Player3
Optimize X'_{k+1} $X_k$and X''_k are fixed by Player1 and Player3
Optimize X''_{k+1} $X_k$and X'_k are fixed by Player1 and Player2

Fig. 2 A block diagram of Nash strategy for three objectives

Figure 3 shows an example of a population and its agents. Here, we can see the agent coding and decoding for a dataset with four features. For example, in the first agent, the indices of *Feature*1, *Feature*2, *Feature*3, and *Feature*4 are indicated by 0.75, 1.33, 2.76, and 3.1, respectively. At the end of this section, two populations are generated, similar to Fig. 3.

5.3 Cluster generation

This section illustrates the correlation measure and explains how clusters are generated based on coalitional games.



No. Agent	Sequencing				No.weighting method
1	0.75	1.33	2.76	3.1	1.8
2	3.74	2.55	1.88	0.54	1.2
3	2.07	3.92	1.12	0.21	0.7
		⋮			
37	2.01	1.11	3.06	0.84	0.1
38	0.65	2.45	1.92	3.33	1.3
39	3.99	0.22	1.88	2.66	2.4
40	2.5	1.77	3.39	0.13	0.6

0.75	1.33	2.76	3.1	1.8

Feature 1	Feature 2	Feature 3	Feature 4	Weighting method number 2

Fig. 3 An example of a population and its agents

5.3.1 Correlation coefficients of features

The proposed algorithm seeks to group the most similar features in the same cluster by a coalitional game. In our method, clustering mainly aims at finding relationships between features where the most distinct and informative features are used in feature extraction. Pearson's correlation coefficient is applied to estimate the correlation between different features. Hence, the correlation coefficient between two features fe_i and fe_j is obtained using Eq. (4) [48]. The features in each agent are taken as players, where they collaborate to attain a higher correlation coefficient:

$$Cp_{ij} = \frac{\sum_{l=1}^{All\_S} (x_i(l) - \bar{x}_i)(x_j(l) - \bar{x}_j)}{\sqrt{\sum_{l=1}^{All\_S} (x_i(l) - \bar{x}_i)^2}\sqrt{\sum_{l=1}^{All\_S} (x_j(l) - \bar{x}_j)^2}}, \tag{4}$$

where $x_i(l)$ and $x_j(l)$ are values of feature vector fe_i and fe_j for the l-th sample, respectively. $\bar{x}_i$ is the mean of x_i and $\bar{x}_j$ is the mean of x_j over all All_S samples. According to Eq. (4), high Cp values indicate high similarity between the two features.

Upon determining correlation coefficients between features, the correlation value for feature i can be defined by Eq. (5) [48]:

$$Corr_i = \frac{\sum_{j=1}^{T} \left|Cp_{ij}\right|}{T - 1} \quad i \neq j, \tag{5}$$

where T denotes the total number of features. The low value of this parameter for a feature suggests a low similarity between this feature and other features (i.e., this is a distinctive feature among other features).

5.3.2 A coalitional game among features

This section explains a coalitional game among features. Each feature is considered as a player in the game. A payoff function is defined by Eq. (6):

$$v_i(j) = \left|Cp_{ij}\right| \quad \forall i, j \in N. \tag{6}$$

It is assumed that $v_i(i) = 0 \quad \forall i.$

There is a preference relation ($\succ_i$) between each feature and the feature subsets based on Eq. (7):

$$K_1 \succ_i K_2 \Leftrightarrow \sum_{j \in C_1} v_i(j) \geq \sum_{j \in C_2} v_i(j) \quad \forall K_1, K_2 \in fe_i \quad i = 1, 2, \ldots, T, \tag{7}$$

where K_1 and K_2 are two features, C_1 and C_2 are two clusters, T denotes the total number of features, and v_i is a payoff function.

While forming a coalition, each feature wants to join the feature group that maximizes its payoff function.

This process is regarded as a coalitional game. Each feature seeks to find a cluster contributing to its high correlation with the features of that cluster while decreasing the correlation between feature and output. Figure 4 depicts an example of a coalitional game envisaged in the proposed method. In this type of game, players try to improve the payoff function. For example, consider a dataset with four original features (i.e., four participating players) and one target. There is an agent with five cells, the first four cells of which indicate the order of features for clustering, and the last one indicates the weighting method for feature extraction. First, a cluster ($G1$) with $Feature4$ that is the first feature number in the agent is generated. Then, the relation (i.e., correlation) between the next feature in the solution (i.e., $Feature2$ in Fig. 4) with the generated cluster and other remaining features are examined.

If the correlation is not improved with joining this feature to cluster $G1$, the aforesaid feature (i.e., $Feature2$) and feature with the highest correlation (i.e., $Feature3$ with a correlation value of 0.8) generate a new cluster called $G2$. So far, the state of three features is determined (i.e., $Feature4$ is in cluster $G1$, and $Feature2$ and $Feature3$ are in cluster $G2$). Now, the correlation of the last feature in the agent (i.e., $Feature1$) with cluster $G1$ and cluster $G2$ is checked. Since there is more than one feature in cluster $G2$, the average correlation between these two features and $Feature1$ should be determined. Ultimately, the final feature is placed in the cluster with the highest degree of correlation with its members. In other words, cluster $G2$ with 0.68 is selected for $Feature1$ in Fig. 4.

5.4 New feature extraction

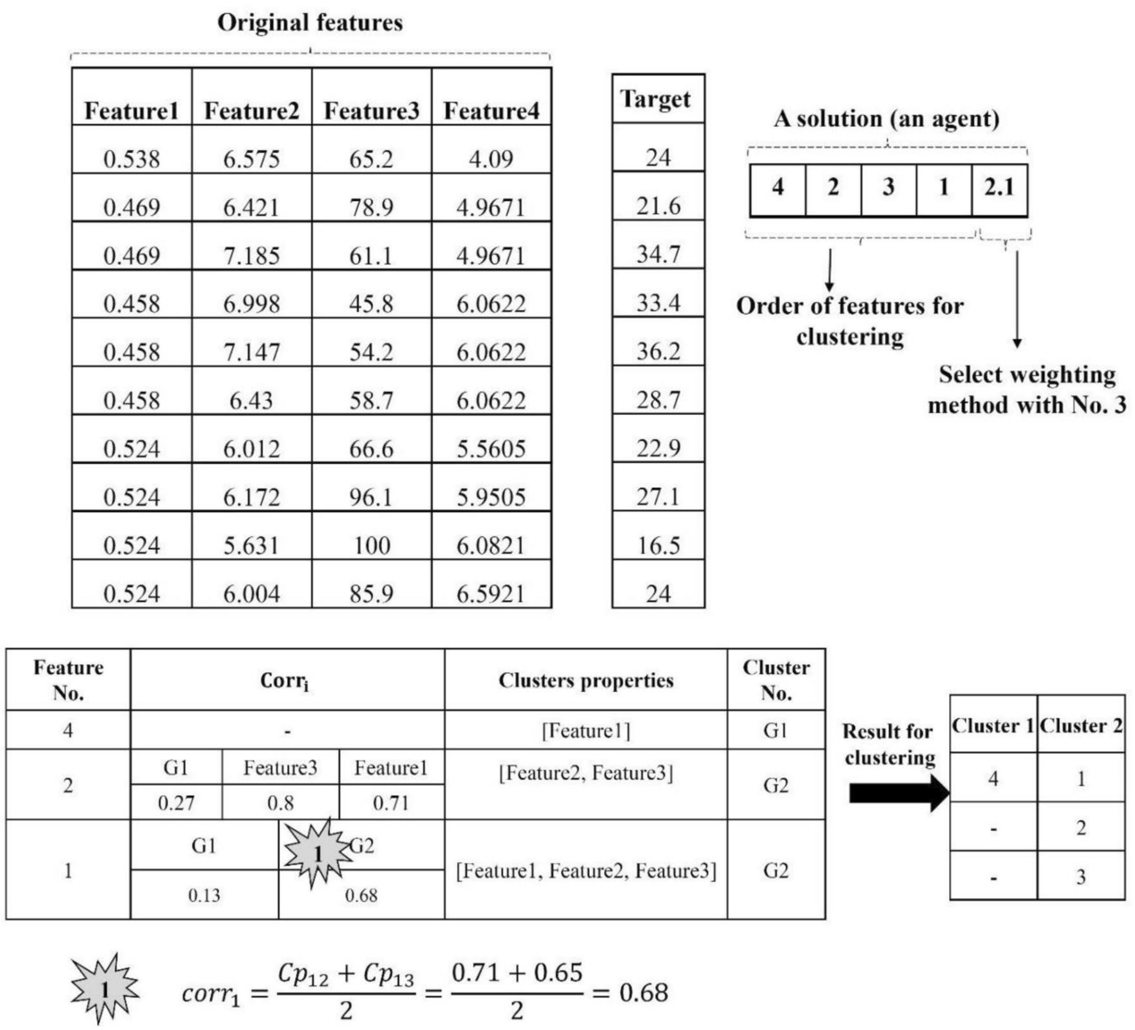
Upon cluster generation, a new feature is extracted from each cluster. If T original features are grouped into C clusters, they can be represented by C new features since all features within a cluster are quite similar. Let us assume that original features are represented with vector $fe_j = < fe_{1j}, fe_{2j}, \ldots, fe_{All\_sj} >^T$, $1 \leq j \leq T$, where All_s denotes the total number of samples in the dataset. Then, the new features are shown with vector $D'_h = < D'_{1h}, D'_{2h}, \ldots, D'_{All\_sh} >^T$, $1 \leq h \leq Number\_new\_features$ and defined as follows:

$$D' = Z * Fe, \tag{8}$$

where

$$D' = [D'_1 D'_2 \ldots D'_h] \tag{9}$$

$$Fe = [fe_1 fe_2 \ldots fe_T]^T \tag{10}$$



Feature1	Feature2	Feature3	Feature4
0.538	6.575	65.2	4.09
0.469	6.421	78.9	4.9671
0.469	7.185	61.1	4.9671
0.458	6.998	45.8	6.0622
0.458	7.147	54.2	6.0622
0.458	6.43	58.7	6.0622
0.524	6.012	66.6	5.5605
0.524	6.172	96.1	5.9505
0.524	5.631	100	6.0821
0.524	6.004	85.9	6.5921

Target
24
21.6
34.7
33.4
36.2
28.7
22.9
27.1
16.5
24

Feature No.	Corr_i			Clusters properties	Cluster No.
4	-			[Feature1]	G1
2	G1	Feature3	Feature1	[Feature2, Feature3]	G2
	0.27	0.8	0.71		
1	G1	G2		[Feature1, Feature2, Feature3]	G2
	0.13	0.68			

Cluster 1	Cluster 2
4	1
-	2
-	3

$$corr_1 = \frac{Cp_{12} + Cp_{13}}{2} = \frac{0.71 + 0.65}{2} = 0.68$$

Fig. 4 An example for coalitional game among features

$$Z = \begin{bmatrix} S_{11} & \dots & S_{1T} \\ \dots & \ddots & \dots \\ S_{h1} & \dots & S_{hT} \end{bmatrix}. \tag{11}$$

Thus, Z shows the weighting matrix. From Eq. (8), we have:

$$D'_h = S_{h1}fe_1 + S_{h2}fe_2 + \cdots + S_{hT}fe_T\,, \quad 1 \le h \le Number\_new\_features. \tag{12}$$

The proposed method assumes three weighting methods as strict, moderate, and hybrid as follows:

Strict method: Here, each feature only affects the creation of a new feature for its own cluster. Hence, for $1 \le j \le T$, it is defined as follows [24]:

$$S^{st}_{hj} = \begin{cases} 1, & if\ \ h = \arg \max\limits_{1 \le h \le number\_clusters} Corr_{st}(fe_j, G_h) \\ 0, & otherwise \end{cases}, \tag{13}$$

where $Corr_{st}$ represents the correlation in the strict weighting method. According to Eq. (13), if fe_j belongs to cluster G_h ,S^{st}_{hj} is 1, and S^{st}_{vj} is 0 for $v \neq h$. However, in this method, a feature cannot contribute to constructing more than one new feature.

Moderate method: Here, each original feature contributes to extracting features according to its correlation value. For moderate method, we have:

$$S^{m}_{hj} = Corr_{mo}(fe_j, G_h) \quad 1 \le j \le T, \quad 1 \le h \le number\_clusters, \tag{14}$$

where $Corr_{mo}$ denotes correlation in the moderate weighting method for generating new features, and T denotes the total number of features.

Based on Eq. (14), feature fe_j has the greatest effect on the construction of feature D'_h when it has the most strong correlation with cluster G_h, while other features are less effective.

Hybrid method: It is a combination of the two methods above. Thus, it can be represented as follows:

$$S_{hj} = \beta \times S_{hj}^{st} + (1 - \beta) \times S_{hj}^{m}, \tag{15}$$

where S_{hj}^{st} and S_{hj}^{m} are determined based on Eqs. (13) and (14), respectively. The value of β is set by the user in the range [0–1]. According to Eq. (15), the hybrid method operates as a strict method when β is set to 1. Likewise, the hybrid method is identical to a moderate method when β is set to 0.

Once D' is calculated, the new dataset can be obtained as follows:

$$X_i' = < D_{i1}', D_{i2}', \ldots, D_{ih}' >, \quad 1 \le i \le All\_s, \quad 1 \le h \le Number\_new\_features. \tag{16}$$

As indicated, the dimension of the input vector for each sample is reduced to h $(h < T)$. Therefore, the new dataset may be stated as follows:

$$d_i' = (X_i', y), \ \ 1 \le i \le All\_s. \tag{17}$$

Using the previous example, a new feature generation procedure can be illustrated. In Fig. 4, the hybrid weighting method used for extracting new features is selected based on the rounded value of the last cell. Now, S_{hj} value is calculated based on Eq. (15). Figure 5 shows an example of the hybrid method. Figure 6 indicates the hybrid values and the newly constructed features.

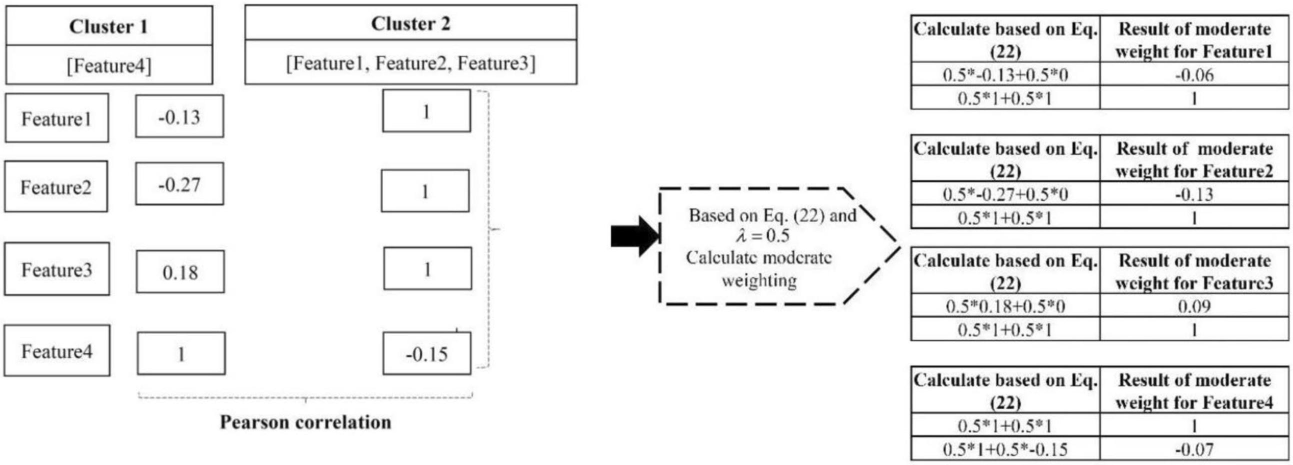
From Fig. 4, it can be observed that the last agent cell is 2.1. Upon rounding up, the third method (i.e., hybrid method) is used for generating new features. The hybrid weighting method is obtained by Eq. (15). It combines strict and moderate weighting methods (Eqs. 13, 14). Hence, it is necessary to calculate the strict and moderate weights of each feature in all clusters. For example, the moderate weight for *Feature*1 based on cluster *G*1 is equal to − 0.13 that is calculated based on Eq. (14). Its strict weight is 0 since *Feature*1 is not in cluster *G*1. The moderate and strict weights based on cluster *G*2 for *Feature*1 are equal to 1; hence, the hybrid weight is set to 1. This process is repeated for each feature. Upon obtaining the weighting matrix for all features in each cluster, the new features are calculated based on Eq. (12). In Fig. 6, a new feature *D*1 based on cluster *G*1 is calculated as follows:

$$(-0.06 \times Feature1) + (-0.13 \times Feature3) + (0.09 \times Feature3) + (1 \times Feature4).$$

This process is repeated for cluster *G*2 like *D*1.

5.5 Updating players and populations

In this work, there are three populations, each of which with different objectives for optimization. The first and second populations seek to minimize the number of features and model errors, respectively. The third one attempts to find a good trade-off between a model with all features and a model with selected features in terms of precision value. Thus, the fitness of each agent is evaluated based on its population objective. Three phases for fitness evaluation are presented in Fig. 7. In summary, steps one and two involve agent initialization and the generation of new features based on the agents in populations, respectively. New features are evaluated in step three.



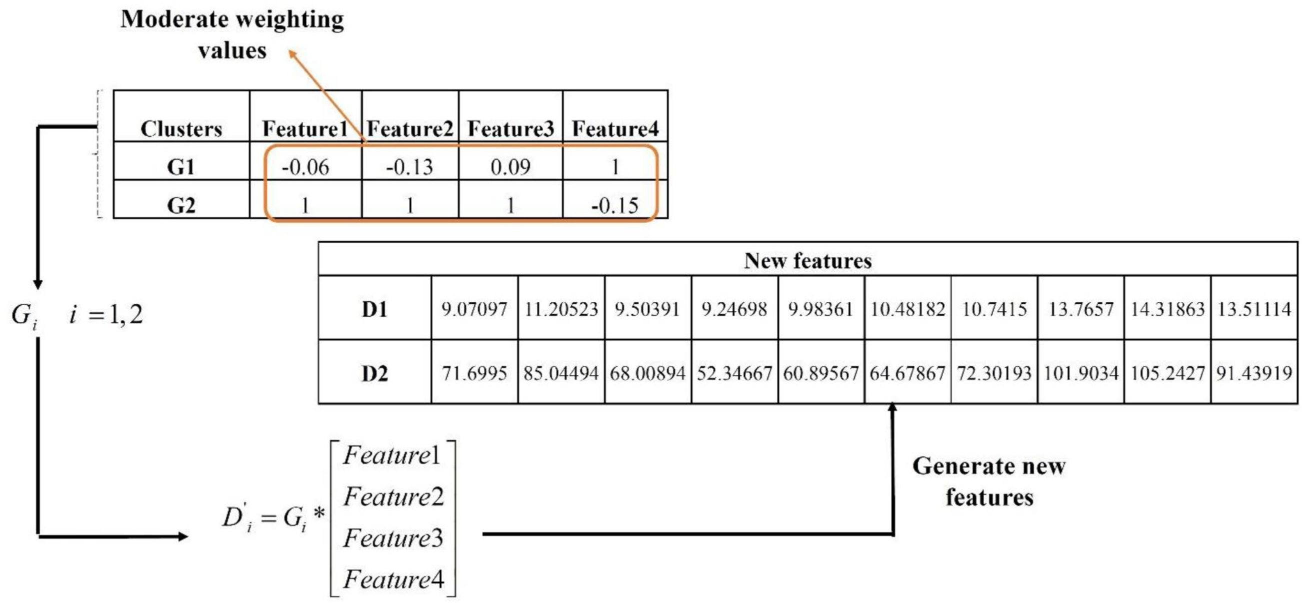
Calculate based on Eq. (22)	Result of moderate weight for Feature1
0.5*-0.13+0.5*0	-0.06
0.5*1+0.5*1	1

Calculate based on Eq. (22)	Result of moderate weight for Feature2
0.5*-0.27+0.5*0	-0.13
0.5*1+0.5*1	1

Calculate based on Eq. (22)	Result of moderate weight for Feature3
0.5*0.18+0.5*0	0.09
0.5*1+0.5*1	1

Calculate based on Eq. (22)	Result of moderate weight for Feature4
0.5*1+0.5*1	1
0.5*1+0.5*-0.15	-0.07

Fig. 5 An example for the hybrid method



Clusters	Feature1	Feature2	Feature3	Feature4
G1	-0.06	-0.13	0.09	1
G2	1	1	1	-0.15

	New features									
D1	9.07097	11.20523	9.50391	9.24698	9.98361	10.48182	10.7415	13.7657	14.31863	13.51114
D2	71.6995	85.04494	68.00894	52.34667	60.89567	64.67867	72.30193	101.9034	105.2427	91.43919

$$D'_i = G_i * \begin{bmatrix} Feature1 \\ Feature2 \\ Feature3 \\ Feature4 \end{bmatrix}$$

Fig. 6 The new generated features

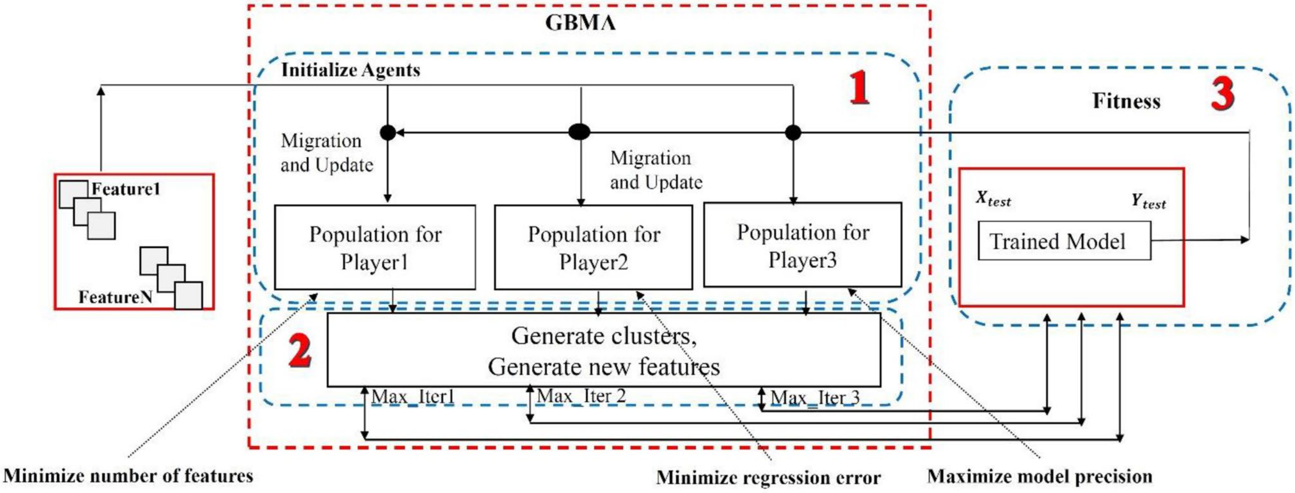


Fig. 7 The general schema for evaluating populations

Evaluation of the first population: To evaluate the agents in the first population, the number of generated features is considered as a basic metric. Since the number of features affects the cost of acquiring sample values and model training, the agent with the lowest number of features is determined as the best agent.

Evaluation of the second population: To evaluate the agents in the second population, the obtained dataset with new features from the second population is trained and tested through Support Vector Regression (SVR). A regression version of SVM with a surrogate loss function is commonly used to solve regression problems. Structural risk minimization (SRM) aims at establishing SVR and minimizing the upper bound on the generalization error. SVR is an effective technique in real-valued function estimation. It applies a nonlinear mapping to map input data x into a high-dimensional feature space Fe and then solves a linear regression problem in the new space.

Regression approximation tries to predict the output value based on a given dataset such as $G = \left\{\left(x_i, y_i\right)\right\}_i^n$, where x_i is the input vector (containing n features $\{fe_1, fe_2, \ldots, fe_n\}$),$y_i$ is the output value, and tot_s is the total number of samples. We seek to find a regression function like $y = f(x)$ to estimate the outputs $\left\{y_i\right\}$ based on a new set of input–output samples like

$\{(x_i, y_i)\}$. In the second population, an agent with the lowest regression error is considered as the best agent.

Evaluation of the third population: To evaluate the agents in the third population, before the clustering procedure, the MSE of features in each agent is calculated using SVR. Once new features are generated, their RSS is calculated by SVR. Finally, the trade-off between the precision of the model with all features and model with selected features is calculated by Mallows Cp, and the agent with the lowest value is taken as the best agent. After evaluating the fitness of each agent, the best agents of the three populations are exchanged for the next iteration. In a situation with two players (i.e., two populations), this process has occurred with the best agents of two populations.

Upon determining the best agents for the two populations and exchanging their places, the process of updating agents is triggered. In BMA, the updating process of agents consists of two main phases: the migration phase and the updating phase.

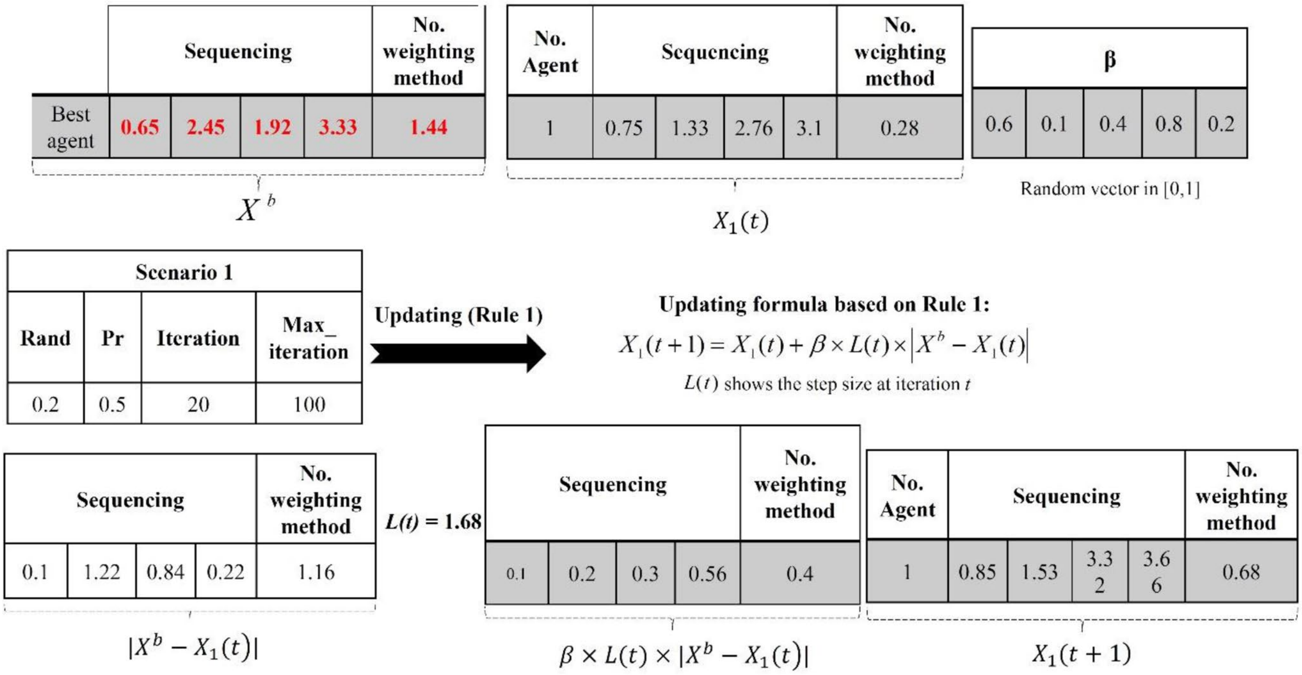
(1) *Migration phase:* It has two stochastic search operators, namely Rule 1 and Rule 2. In Rule 1, each agent moves to the best agent, and thus the optimal solution is found as quickly as possible. In Rule 2, each agent randomly moves to other positions, and hence exploration ability is improved. Two examples of the migration phase are shown in Figs. 8 and 9.

In Fig. 8, attempts have been made to update agent with number one ($X_1(t)$) based on Rule 1 in the 20-th iteration, where the maximum number of iterations is set to 100. Furthermore, *Rand* and *Pr* values are assumed as 0.2 and 0.5, respectively. Firstly, the best agent (i.e., *Best-agent* in Fig. 8) must be determined from the previous iteration. For example, in Fig. 8, *Best-agent* equals 0.65, 2.45, 1.92, and 3.33 in the *Sequencing* part and 1.44 in the last part. Once updated, it would be 0.85, 1.53, 3.32, and 3.66 for the *Sequencing* part and 0.68 for the last part of the best agent.

In Fig. 9, it is attempted to update agent with number one ($X_1(t)$) based on Rule 2 in the 20-th iteration, where the maximum number of iterations is set to 100. Additionally, *Rand* and *Pr* values are set to 0.2 and 0.5, respectively. First, two random agents must be specified. For example, in Fig. 9, an agent with number two ($X_2(t)$) and an agent with number forty ($X_{40}(t)$) are randomly selected.

Consider an agent with number one equal 0.75, 1.33, 2.76, and 3.1 in the *Sequencing* part and 0.28 in the last part. Then, it is updated with updating formulas based on $X_2(t)$ and $X_{40}(t)$. Finally, the values in the *Sequencing* and the last part are replaced with new values. Hence, it is equal to 1.49, 1.4, 3.36, and 3.42 in the *Sequencing* part and 0.51 in the last part of the agent with number one.

(2) *Updating phase:* The updating phase can be used to enhance the diversity of populations. If an agent cannot be improved after a certain number of iterations, it must be replaced by another. Once updated, the validation form of the agents of the two populations must be checked. An agent has a valid form if (1) all integral part of the value in the *Sequencing* part falls within the range (0, *T*), where *T* denotes the total number of original features, (2) the value of *No. Weighting* falls within the range (0, 3), and (3) the



	Sequencing				No. weighting method
Best agent	0.65	2.45	1.92	3.33	1.44

No. Agent	Sequencing				No. weighting method
1	0.75	1.33	2.76	3.1	0.28

β				
0.6	0.1	0.4	0.8	0.2

Scenario 1			
Rand	Pr	Iteration	Max_ iteration
0.2	0.5	20	100

Updating formula based on Rule 1:

$$X_1(t+1) = X_1(t) + \beta \times L(t) \times \left|X^b - X_1(t)\right|$$

$L(t)$ shows the step size at iteration t

Sequencing				No. weighting method
0.1	1.22	0.84	0.22	1.16

Sequencing				No. weighting method
0.1	0.2	0.3	0.56	0.4

No. Agent	Sequencing				No. weighting method
1	0.85	1.53	3.3 2	3.6 6	0.68

Fig. 8 The process of updating with Rule 1

integral part of the value in the *Sequencing* part is different from each other.

An invalid agent is transformed into a valid form in two steps as follows:

Step 1: The values in *Sequencing* and *No.weighting* parts outside the range are modified based on Eq. (18):

$$fixed\_x = \begin{cases} \operatorname{mod}(x, N) & if\ x \in\ Sequencing\ part \\ \operatorname{mod}(x, 3) & if\ x \in\ No.weighting\ part \end{cases}. \tag{18}$$

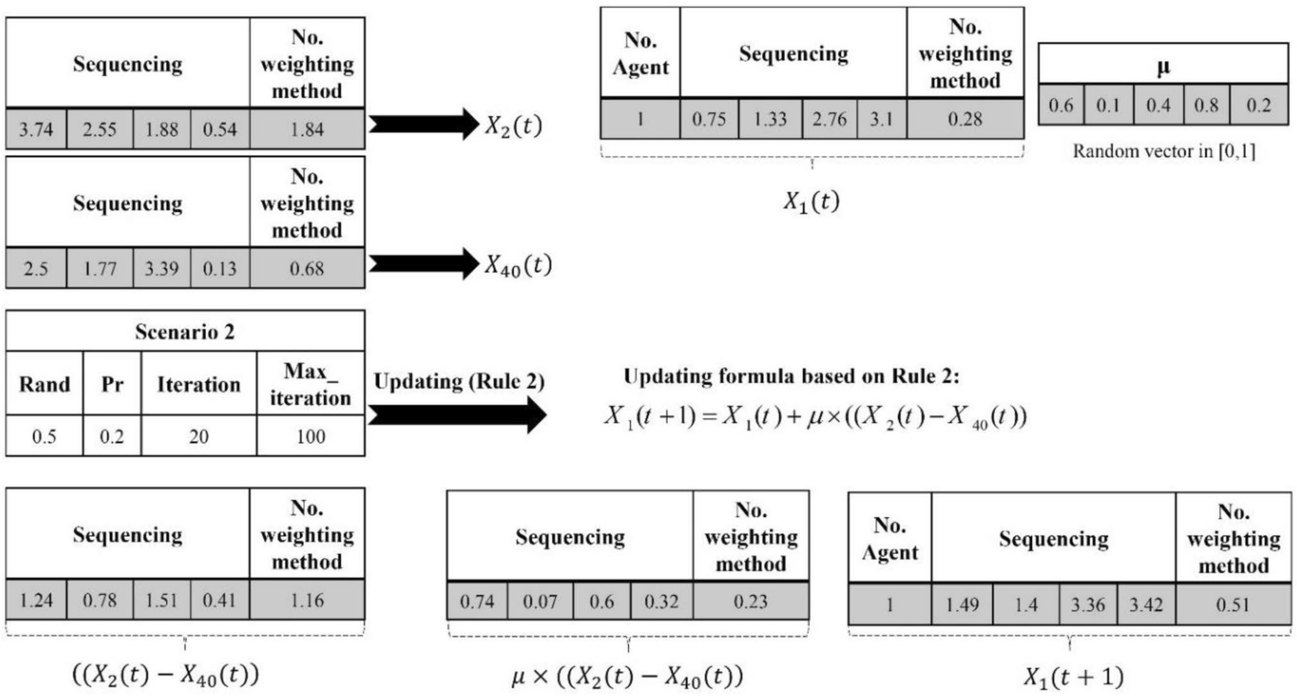
where *fixed_x* shows the corrected form of value *x* and function *mod*(*a, b*) returns the remainder after *a* is divided by *b*.

Step 2: First, the values with the same (i.e., repeated) integral parts and the values with missing integral parts are identified. Then, one of the values with the repeated integral part is randomly replaced with the value with missing integral part.

Algorithm 2 shows the main steps of the proposed method.

Algorithm 2. Main()

```
Input: Population_size, Max_iteration, Dataset
Output: Best features
s=1;
for (int j= 1; j<= 3; j++)
  {
 for (int k= 1; k<= Population_size; k++)
   {
     $Agent_k^j$ = Generate_agents(Dataset); //Initialize agents of three populations
     $Agent_k^j$ = Check_valid( $Agent_k^j$ ); //Check whether the generated agents are valid or not
     $MSE_k^j$ =Cross_validation( $Agent_k^j$ , Dataset); //Calculate $MSE_{FULL}$ for all agents
     Clustering= Generate_cluster( $Agent_k^j$ ); //Cluster features
     [new_features, new_features_number]= Extract_features (Clustering, $Agent_k^j$ , Dataset); //Generate new features by
clustering features and a weighting method that suggested by $Agent_k^j$
     RMSE= Cross_validation (new_features, Dataset); //Calculate RMSE for new features with 10 fold_cross validation
     Calculate $f$ ( $Agent_k^j$ ); //Calculate the fitness of j-th agent in k-th population
  } //end for
  Pop1best= Find_pop1best( $Agent^1$ ); //Find the best agent among all other agents in the first population
  Pop2best= Find_pop2best( $Agent^2$ ); //Find the best agent among all other agents in the second population
 Pop3best= Find_pop3best( $Agent^3$ ); //Find the best agent among all other agents in the third population
} //end for
While (Pop1best $\neq$ Pop2best $\neq$ Pop3best || s< Max_iteration) do //Pop1best is the best agent in the first population, Pop2best
is the best agent in the second population and Pop3best is the best agent in the third population
{
  for (int j=1; j<= 3; j++)
       for (k=1; k<= Population_size; k++)
        {
         $Agent_k^j$ = Update_agents( $Agent^j$ , Pop1best, Pop2best, Pop2best); //Update agents of three populations
        $Agent_k^j$ = Check_valid( $Agent_k^j$ );
        $MSE_k^j$ =Cross_validation( $Agent_k^j$ , Dataset);
        Clustering= Generate_cluster( $Agent_k^j$ );
        [new_features, new_features_number]= Extract_features(clustering, $Agent_k^j$ , Dataset);
        RMSE= Cross_validation(new_features, Dataset);
        Calculate $f$ ( $Agent_k^j$ );
       }
   Pop1best=Find_pop1best( $Agent^1$ );
   Pop2best=Find_pop2best( $Agent^2$ );
   Pop3best= Find_pop3best( $Agent^3$ );
   Update Pop1best;
   Update Pop2best;
   Update Pop3best;
} //end while
s=s+1;
return the best features generated by Pop1best or Pop2best or Pop3best;
```



Sequencing				No. weighting method
3.74	2.55	1.88	0.54	1.84

Sequencing				No. weighting method
2.5	1.77	3.39	0.13	0.68

No. Agent	Sequencing				No. weighting method
1	0.75	1.33	2.76	3.1	0.28

μ				
0.6	0.1	0.4	0.8	0.2

Scenario 2			
Rand	Pr	Iteration	Max_ iteration
0.5	0.2	20	100

$$X_1(t+1)=X_1(t)+\mu\times((X_2(t)-X_{40}(t))$$

Sequencing				No. weighting method
1.24	0.78	1.51	0.41	1.16

Sequencing				No. weighting method
0.74	0.07	0.6	0.32	0.23

No. Agent	Sequencing				No. weighting method
1	1.49	1.4	3.36	3.42	0.51

Fig. 9 The process of updating with Rule 2

6 Complexity reduction

The high computational complexity of an evolutionary algorithm is attributed to two issues: (1) the scale of the population in the swarm and (2) the size of the search space [49]. This section initially illustrates how a swarm population can be scaled by a parallel technique (i.e., MapReduce). Obviously, the more agents evaluated, the more computation time will be needed, and so the population of the swarm should be cut down. Then, the space reduction is explained.

6.1 GBMA based on MapReduce

A parallel model is designed for BMA using the MapReduce technique. There are two dependencies between agents in the migration phase as follows:

(1) Exchanging the global best.
(2) Moving toward the direction of neighbors.

Now, we explain how GBMA is parallelized (see Fig. 10). In each iteration of GBMA, there are two main phases (i.e., migration and updating). It evaluates the agent at the new point and updates its global best after being compared with its neighbors. Each agent operates independently of the rest of the swarm, except for updating its global best and moving toward neighbors. Due to the limited communication among agents, updating a swarm can be formulated as a MapReduce operation. Once mapped, an agent receives a new position, value, and information of its right and left neighbors. In the reduction phase, it incorporates information from other agents into the swarm to update its global best. In the parallel version of BMA, the global best is communicated only between right and left neighbors rather than all the agents, rendering MapReduce more efficient.

6.2 Dynamic varying search area (DVSA)

The complexity of the optimization problem is not only linked to its objective functions but also the search space is one of the essential factors [49]. To accelerate the processing of the algorithm, the search area of each population is modified or reduced dynamically. In each population, assume that there are N cooperative agents in the search space. When the minimal distances between agents of each population (i.e., first and second populations) reach a threshold, the real optimal solution should be found in the region around these agents based on the maximum likelihood estimation (MLE). Therefore, the previous search area S is reduced to S'. Now, there are two new populations with the same number of agents, and the search space is reduced. Figure 11 indicates the case in which four cooperative agents reduce their search spaces in the second population. First, they search the solution in S and get the best agents R_1^*, R_2^*, R_3^*, and R_4^* included in S'. Accordingly, the search area becomes S'. The same procedure applies to the subsequent reductions to S''.

6.3 Complexity analysis

In GBMA, assume a dataset with N features, and each population with M agents. Since GBMA is implemented in parallel form (e.g., like Fig. 10), the time complexity of the whole method is equal to maximum time complexity for the execution of one agent. The time complexity of the proposed algorithm is estimated based on the pseudo-code (i.e., Algorithm 2) as follows:

Step 1. Generate populations and check agents—lines: 4–9:

For N features and M agents: $O(3 \times 2 \times M(N+1))$,

Step 2. Calculate MSE—line 10:

For N features and p training samples: $O(N^2p + N^3)$,

Step 3. Cluster features—line 11:

For N features: $O(N \log N)$,

Step 4. Extract new features (calculate weights and generate new features)—line 12:

For N features and d clusters: $O(2Nd)$,

Step 5. Calculate RMSE—line 13:

For k extracted features and p training sample: $O(k^2p + k^3)$,

Step 6. Calculate fitness—line 14:

For M agents, in population one: $O(M)$ and in populations two and three: $O(M \times (k^2p + k^3))$.

Step 7. Find the best agent: $O(M)$.

These six main steps are repeated from line 22 until the end. Consequently, the total computational complexity of the proposed algorithm is $O(MM + N^3 + Nd + MN^3 + N \log N)$.

7 Experimental results and comparisons

The structure of the experimental results is divided into three main subsections. The first subsection (8.1) describes the characteristics of the regression datasets (all datasets are accessible from the UCI data repository [50]) and the parameter setting of the compared algorithms. The second subsection (8.2) investigates the performance of the proposed method in terms of the feature evaluation criteria in three parts (i.e., swarm-based methods, non-swarm based methods, and statistical tests). The third subsection (8.3) studies the appropriate values for parameter L that adjusts the exploration and exploitation in GBMA for different types of datasets. The applicability of the proposed method is also investigated for classification problems. The details of classification datasets and the comparison results are given in Appendix 1.

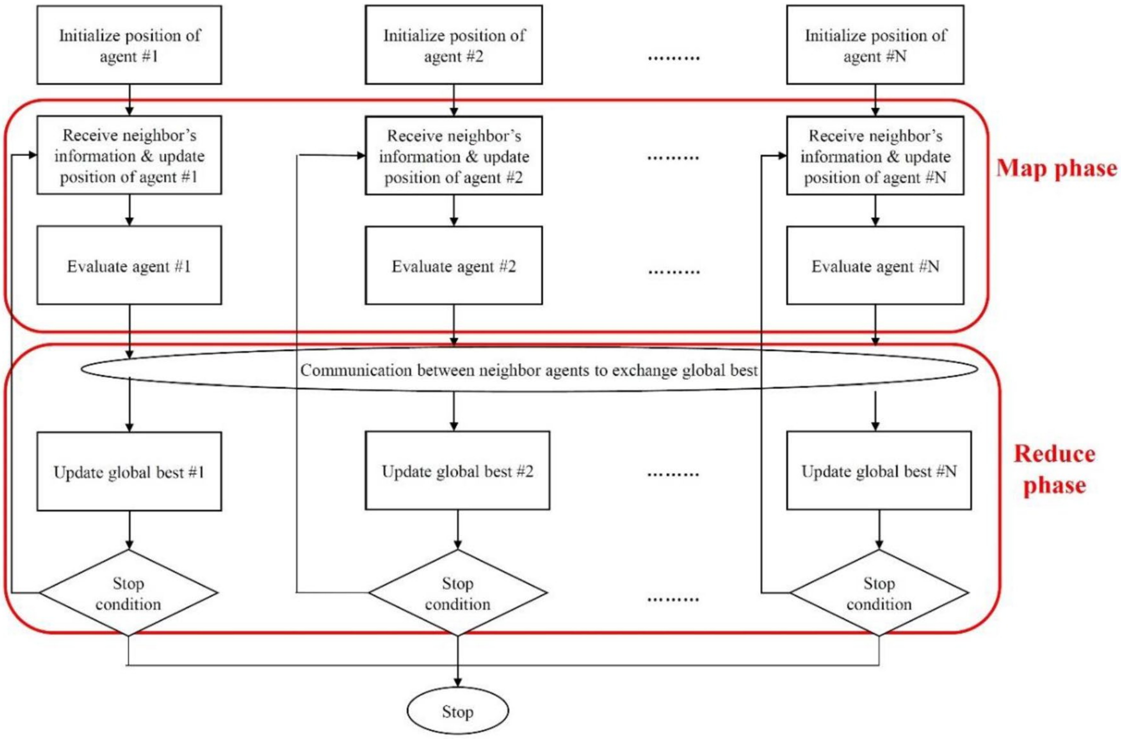

Initialize position of agent #1
Initialize position of agent #2
....
Initialize position of agent #N
Receive neighbor's information & update position of agent #1
Receive neighbor's information & update position of agent #2
Receive neighbor's information & update position of agent #N
Map phase
Evaluate agent #1
Evaluate agent #2
Evaluate agent #N
Communication between neighbor agents to exchange global best
Update global best #1
Update global best #2
Update global best #N
Reduce phase
Stop condition
Stop condition
Stop condition
Stop

Fig. 10 Diagram of parallel BMA based on MapReduce

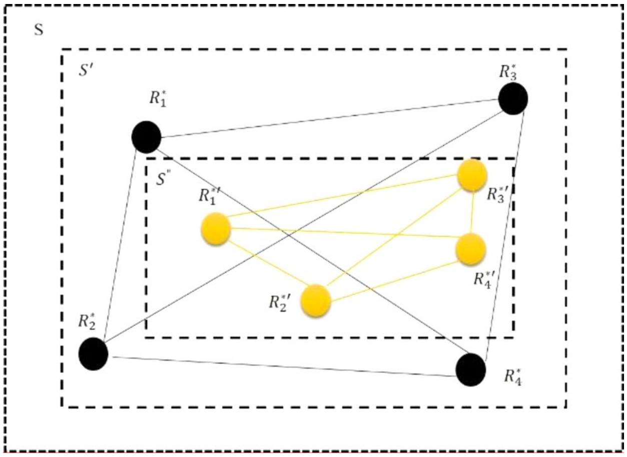


Fig. 11 An example for search space reduction

7.1 Datasets and parameter setting

The proposed method is evaluated using ten regression datasets. Table 2 shows the details of these regression datasets. Table 3 demonstrates the parameter setting procedure of the compared methods.

7.2 Performance evaluation of the compared methods

7.2.1 Swarm-based methods

This section evaluates the convergence and efficiency of the proposed algorithm (i.e., GBMA) using three swarm-based feature selection methods as follows:

- *Artificial bee colony and gradient boosting decision tree* (*ABCoDT*) [37]: It combines bee colony and decision tree to enhance the quality of the selected features.
- *Modified chaotic particle swarm optimization* (*CPSOS*) [51]: It tries to solve feature selection problems by chaotic particle swarm optimization and strikes a balance between exploration and exploitation by the sigmoid-based acceleration coefficients.
- *Modified grey wolf optimization* (*MGWO*) [52]): It modifies the GWO algorithm to set its parameters and control the exploration and exploitation capabilities.

They are compared in terms of the number of selected features (NSF), regression error (RMSE), and fitness value by ten regression datasets. RMSE and the average NSF are two important criteria to identify how well a feature selection method can choose the most relevant features. Generally, a suitable method obtains a low number of features along with a low RMSE value. In our study, SVR with ten-fold cross-validation is used as a regression model to predict the observed value. The RMSE of each method is calculated by the subset of selected features using Eq. (19) as follows:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n} (p_i - o_i)}{tot\_s}}, \tag{19}$$

where p_i and o_i indicate the predicted and observed values of sample i, respectively. The total number of samples in the training set is represented by tot_s.

Figure 12 shows the RMSE value along with the number of selected features for four metaheuristic methods based on the *Bh* dataset. As seen, the proposed method (i.e., GBMA) has outperformed other methods and achieved a lower number of features. For 30 iterations, the proposed method selects, on average, 4.4 features, outperforming CPSOS, MGWO, and ABCoDT by 32%, 38%, and 50%. Moreover, it shows, on average, 3.2 for RMSE and so reduces the error rate by 33%, 41%, and 51% compared to CPSOS, MGWO, and ABCoDT, respectively. This improvement can be attributed mainly to the fact that GBMA considers the order of features during clustering and then extracts the new features. Therefore, the proposed method can investigate different

Table 2 Regression datasets details

Dataset	#Instances	#Features	Class
Boston housing (Bh)	506	13	Median value of owner-occupied homes in $1000s
Airfoil self-noise (Af)	1503	6	Scaled sound pressure level in decibels
Breast cancer Wisconsin (Bc)	569	32	Diagnosis value
Communities and crime (Cc)	1994	128	Total number of violent crimes per 100 K population
Relative location of CT slices on axial axis (Rct)	53,500	386	Location of a CT slice on the axial axis of the human body
Compressive strength of concrete (Cs)	1030	9	Concrete compressive strength
YearPredictionMSD (Yp)	515,345	90	Prediction of the release year of a song from audio features
Computer hardware (Ch)	209	9	Relative CPU performance
Real estate valuation (Re)	414	7	House price of unit area
Orange juice (Oj)	218	700	Determination of saccharose concentration on orange juice

arrangements of features and appropriately clusters all features.

Tables 4 and 5 show the average RMSE values and the NSF for different methods based on various datasets, respectively. From Table 4, it can be seen that the GBMA method outperforms the three other approaches for all test datasets, except for two datasets on which MGWO and CPSOS outperform the other methods with a slight performance difference from the proposed approach. GBMA realizes a much lower RMSE value than other similar methods in almost all regression problems. It shows an average RMSE of ≈ 2.9, while the closest method (i.e., CPSOS) indicates an RMSE of 3.2. Tables 4 and 5 show that CPSOS has better performance in terms of the NSF and RMSE value compared with ABCoDT (on average, CPSOS selects 10% smaller number of features and shows 12% lower RMSE value). This is because CPSOS possesses improved global and local search capabilities thanks to its chaotic function and searches the potential high-performance regions of the feature space.

For subsequent experiments, we consider *R*-squared (R^2), a statistical measure for a regression model that investigates the scattering pattern of data points along the fitted regression line. Once the regression model (e.g., SVR) has been fit, we must investigate how well the model fits the data. One of the goodness-of-fit (GoF) statistics is *R*-squared for regression analysis. *R*-squared is defined as Eq. (20) ranging from 0 and 100%:

$$R^2 = \frac{Explained\ variation}{Total\ variation}. \tag{20}$$

If a model can explain all the variability of response data around its mean, R^2 is 100%. For the same dataset, high values of this criterion would indicate a small difference between actual and predicted data. Figure 13 illustrates the performance of the compared methods in terms of *R*-squared for the *Bh* dataset. We can see from Fig. 13 that GBMA performs well for the *Bh* dataset and improves *R*-squared about 7%, 12%, and 5% compared to MGWO, ABCoDT, and CPSOS, respectively. When a feature selection method achieves the high value for *R*-squared compared to other methods means that it can find more influential independent variables. We try to determine features that are independent of each other, each of which with highly dependent on the output variable.

Table 3 Parameter setting of the compared algorithms

Method	Parameter	Value
GA	Crossover rate	0.6
	Mutation rate	0.2
PSO	c_1	2
	c_2	2
	Inertia weight	0.7
FA	Absorption coefficient	1
	Leves index	1.5
GBMA	$Cycle_{up}$	80
	L	Linearly decrease from 2 to 0.3
	P_r	0.2
General	Population size	50
	Number of iterations	100

Table 6 demonstrates the *R*-squared values achieved by different methods for nine datasets. Here, we can see that GBMA and CPSOS approaches obtain higher *R*-squared values (18% and 13%, on average) compared to the other methods. In other words, there is a slight difference between the actual and predicted values, and thus a regression model fits the data well. The advantageousness of GBMA can be mainly attributed to the updating phase. The solution is checked in terms of improvement. If a solution cannot achieve better results (i.e., lower error regression in the regression problem), it is removed from the population and replaced with another one. Hence, the wide areas of search space have been investigated to find the best features. The good performance (behavior) of the CPSOS method is associated with its chaotic function that improves the search process and contributes to escape from local optimum.

The objective function that is defined based on Eq. (3), is applied to evaluate the fitness of each selected feature subset. The individual (i.e., agent) with low regression error and the small number of features and low Mallows' Cp value shows a better fitness value. The results of fitness for different metaheuristic methods based on various datasets are shown in Fig. 14. It can be observed that the fitness characteristics are almost the same for GBMA and CPSOS from 35 to 85 iterations for the *Af* dataset. However, afterward, the fitness of GBMA suddenly reduced since GBMA suddenly jumped out of the local optimal where CPSOS was trapped. From Fig. 14, the best performance is achieved by the proposed algorithm (i.e., GBMA) in the fitness value obtained. Thus, it proves the capability of GBMA in adaptively searching the feature space. GBMA shows 19%, 17%, and 11% improvement in terms of fitness value compared to ABCoDT, MGWO, and CPSOS, respectively. As shown in Fig. 14, GBMA outperforms CPSOS on 9 datasets i.e., *Af*, *Bh*, *Bc*, *Rct*, *Ch*, *Cc*, *Re*, and *Oj*, with improved fitness values of about 6.7%, 5.2%, 2.7%, 3.1%, 13%, 17%, 2.3%, and 9%, respectively. One of the disadvantages of CSPSO is that it may not reach global optima and improve its best solution regularly. This is because its mutation update procedure for both local best and global best lacks a mechanism to maintain the best previous solution of each firefly. Thus, they move regardless of their previous best situation.

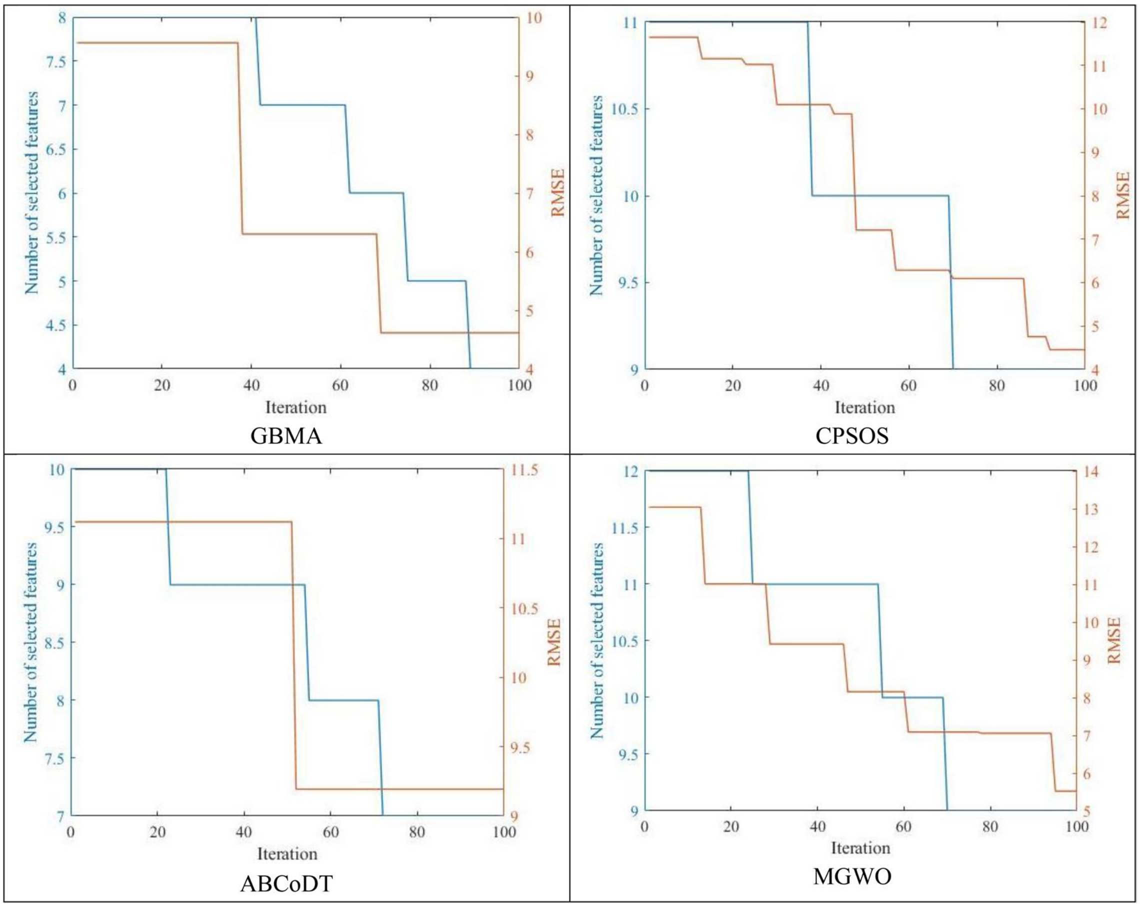


Fig. 12 RMSE and NSF for different methods

Table 4 Average of RMSE for different methods

Method Dataset	GBMA	CPSOS	MGWO	ABCoDT
Af	**1.82**	1.95	1.93	2.13
Bh	**3.2**	5.2	6.0	6.6
Bc	**4.7**	4.95	5.1	5.5
Rct	**1.56**	1.94	2.05	1.83
Ch	**2.01**	2.07	2.28	2.5
Cs	3.56	**3.42**	3.71	3.71
Cc	**4.3**	4.5	6.12	5.09
Re	4.62	4.62	**4.27**	4.94
Yp	**2.11**	2.44	2.35	2.26
Oj	**1.42**	1.8	1.76	1.5

The lowest average RMSE value among different methods is indicated in bold values

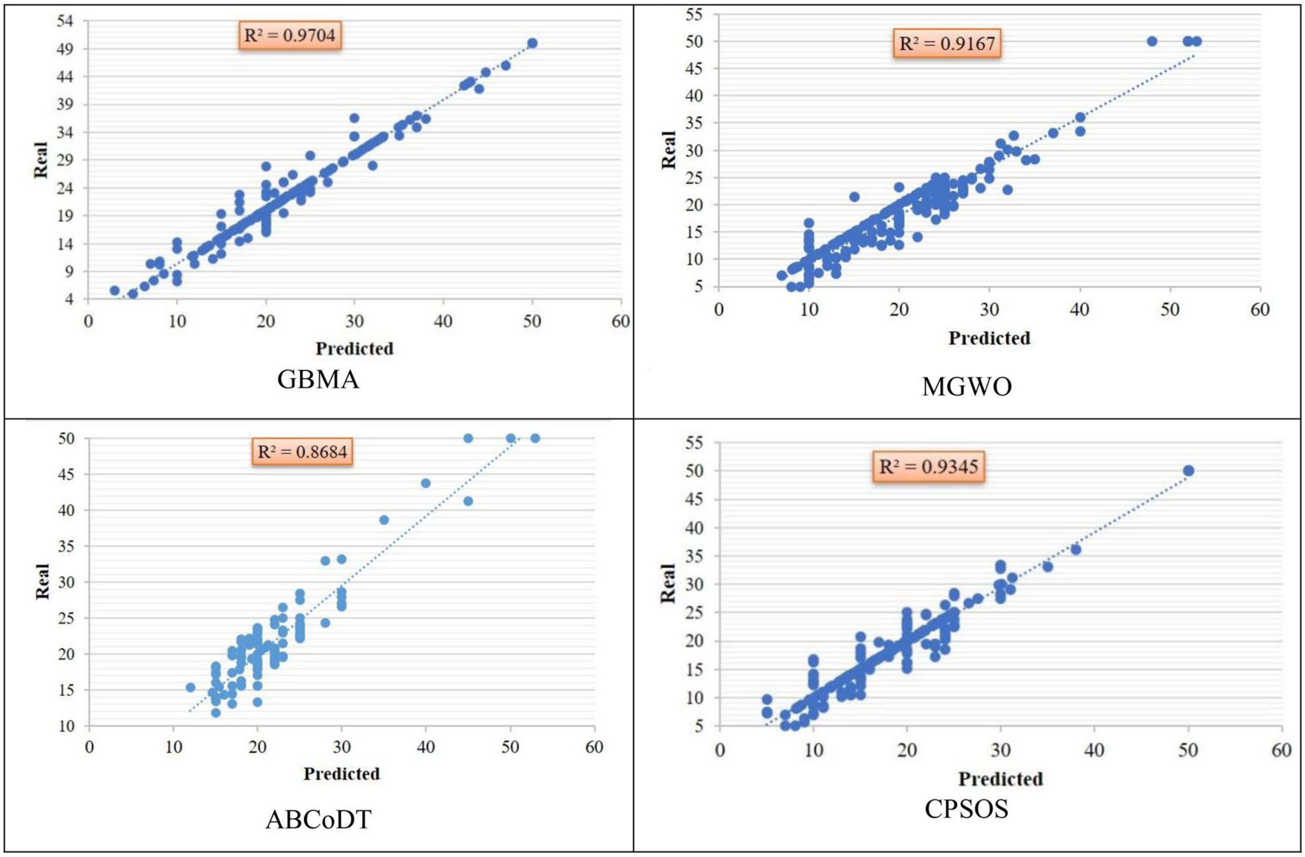
Table 5 Average of NSF for different methods

Method Dataset	GBMA	CPSOS	MGWO	ABCoDT
Af	**3.5**	4.5	4.9	5.8
Bh	**4.4**	7.1	8.3	9.6
Bc	13.5	15.1	18.3	**10.8**
Rct	**85**	103.7	99.4	113.7
Ch	**4.5**	6.4	6.1	5.6
Cs	**2.3**	4.64	4.25	5.87
Cc	**41.6**	57.1	49.3	45
Re	**2.4**	4.5	4.6	5.2
Yp	**24.5**	54.6	43.7	74.5
Oj	**252.3**	280.6	285	311.4

The lowest average NSF value among different methods is indicated in bold values

Nonetheless, BMA accepts the new (i.e., the updated) position of an individual when the previous fitness improved by changing its position. If fitness does not improve, a new individual is generated; thus, BMA can escape local optimum.

From Table 7, we can see that GBMA has the best performance in most cases (bold numbers). For instance, it has an average fitness value of 4.5, the lowest among the four algorithms on the *Re* dataset. MGWO also has an acceptable



GBMA

MGWO

ABCoDT

CPSOS

Fig. 13 R-squared of different methods based on *Bh* dataset

Table 6 R-squared of different methods

Method Dataset	GBMA	CPSOS	MGWO	ABCoDT
Af	**0.976**	0.95	0.94	0.93
Bh	**0.97**	0.93	0.916	0.86
Bc	**0.955**	0.92	0.89	0.93
Rct	**0.975**	0.94	0.91	0.9
Ch	**0.87**	0.83	0.825	0.82
Cs	0.88	**0.93**	0.83	0.83
Cc	**0.95**	0.9	0.85	0.87
Re	0.88	0.85	**0.93**	0.83
Yp	**0.982**	0.94	0.945	0.95
Oj	**0.94**	0.91	0.9	0.88

The highest R-squared value among different methods is indicated in bold values

fitness value, 3.7. It is also indicated that ABCoDT and CPSOS are fairly similar in performance on the *Ch* dataset. Their average fitness values are nearly the same (8 and 8.1, respectively). Having the highest fitness values, ABCoDT does not exhibit excellent performance compared to other algorithms on *Oj*, *Re*, *Cc*, *Bh*, and *Af* datasets.

7.2.2 Non-swarm-based methods

This section compares the proposed method with non-swarm-based methods (i.e., methods without heuristic search algorithms like BMA). The feature reduction methods are divided into two types: filter methods and feature extraction methods. To show the generality of the proposed method, it was also compared with some filter methods.

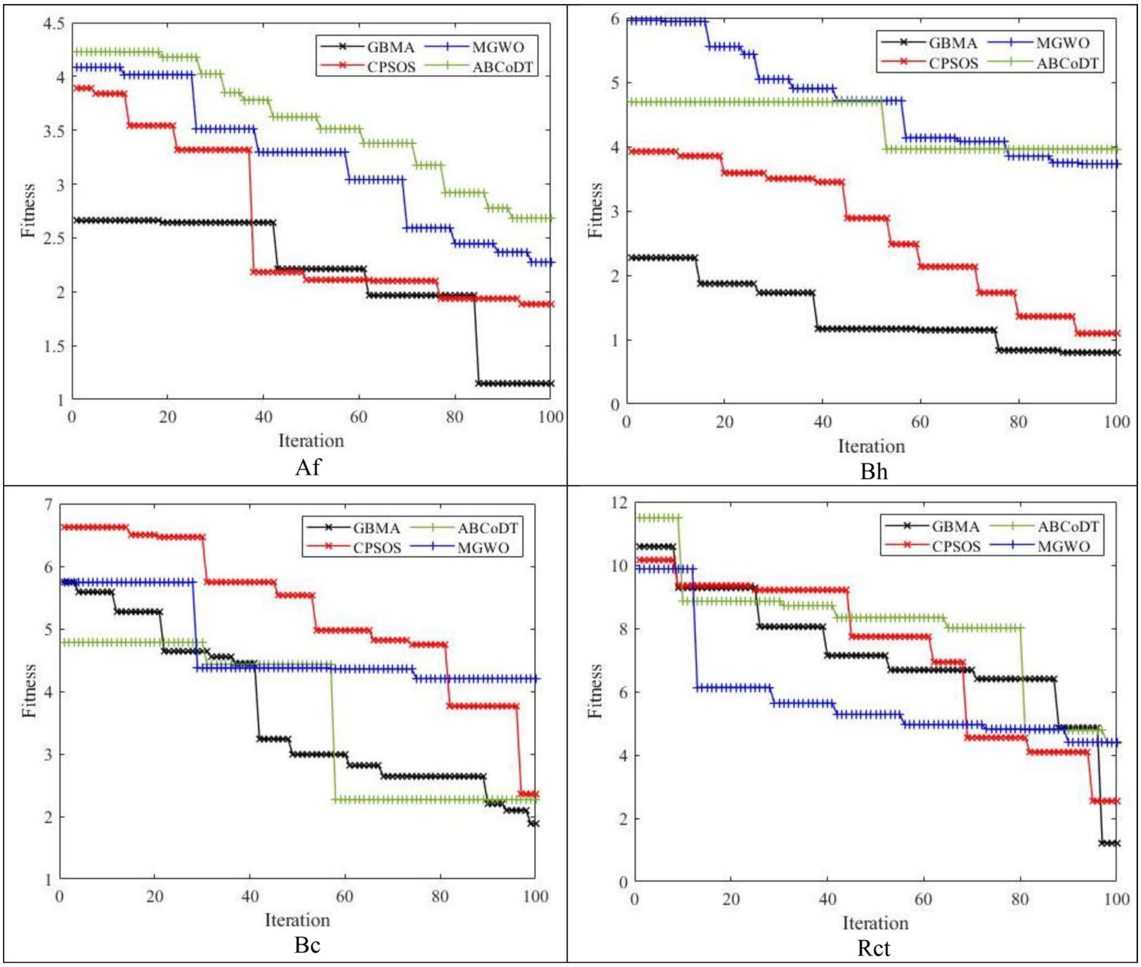


Fig. 14 Fitness values for different datasets

Filter methods, along with their performance results in classification datasets, are explained in detail in Appendix 2. The three basic feature extraction algorithms consist of three methods, namely FC-C-S [16], weighted principal component analysis (WPCA) [16], and linear discriminant analysis for regression (LDAr) [16].

In this section, the compared methods are evaluated in terms of average *R*-squared, average NSF, and average RMSE. The performance of the compared methods on different regression datasets can be seen from Fig. 15. As reflected, the proposed method achieves better results in most of the datasets. For example, GBMA improves *R*-squared and NSF by 18% and 28%, respectively, compared to FC-C-S. Furthermore, GBMA achieves high *R*-squared value in comparison with other methods since it can find independent variables that are more influential. This advantage can be attributed partly to the fact that the proposed method applies BMA as a search method to detect the appropriate order of features for generating clusters such that features in the same cluster have a high correlation.

In Fig. 15, GBMA improves *R*-squared and RMSE by 35% and 38%, respectively, in comparison with LDAr. Since GBMA defines three weighting methods (i.e., moderate, strict, and hybrid) for generating new features, it can obtain the information of features from other clusters through these weighting methods. While LDAr transforms the input space into a high-dimensional feature space by a kernel trick and tries to maximize the ratio of distances of samples with significant differences in the output variable, it does not consider the relativeness between features.

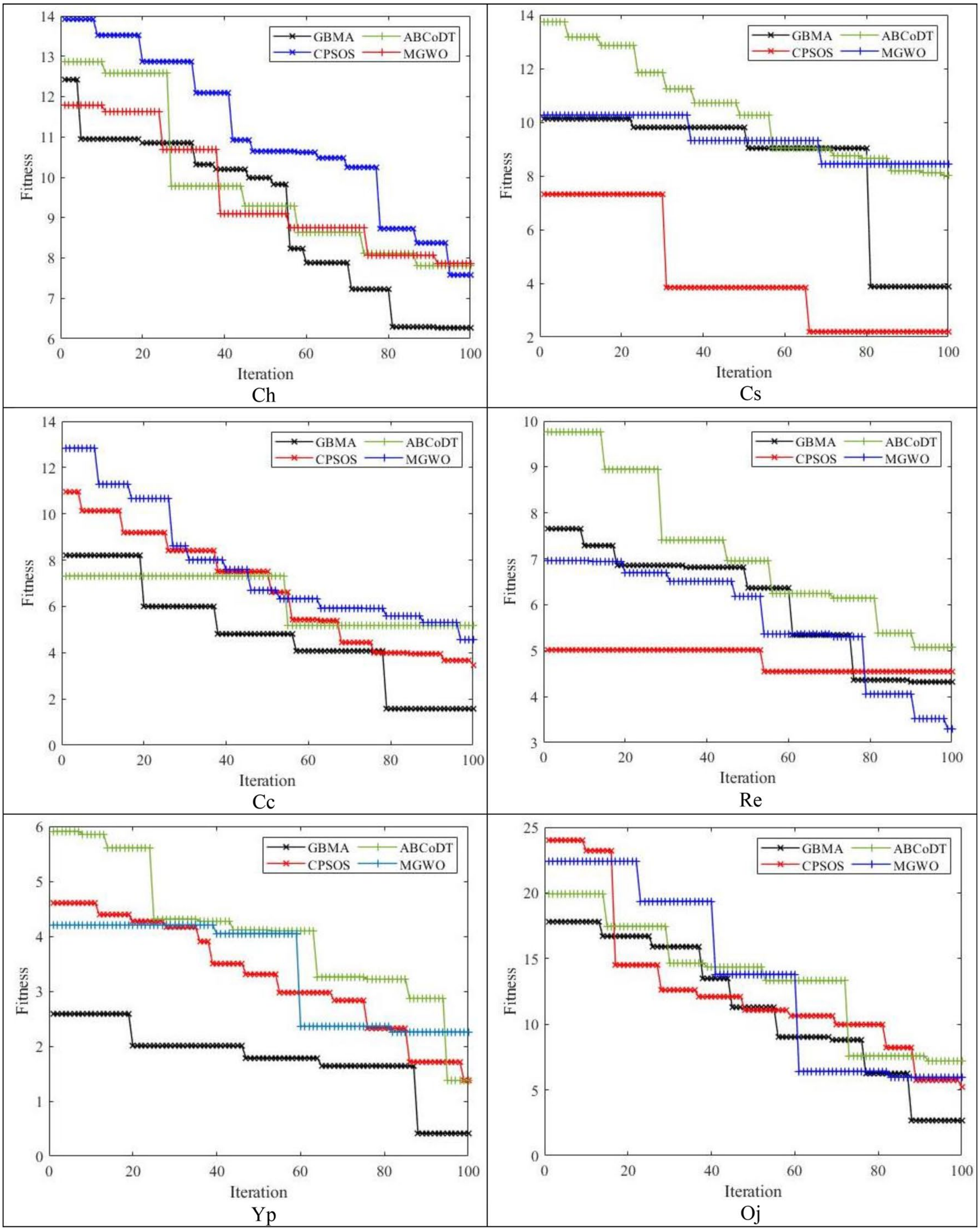


Fig. 14 (continued)

7.2.3 Statistical tests

In this section, several statistical tests are applied to provide a comparative analysis of the proposed method. The Friedman test is a nonparametric statistical test that functions based on the average rank ($rank_i$) of each strategy [53]. Nonparametric testing means that no particular distribution is assumed for the data. Moreover, it can evaluate the results of *N* different strategies for *M* datasets. The null hypothesis (i.e., there is no statistical difference in the performance of each strategy) is accepted or rejected according to the *P* value that is determined by chi-square distribution. Holm's

Table 7 Average the best fitness value for the compared methods

Method Dataset	GBMA	CPSOS	MGWO	ABCoDT
Af	**1.3**	1.6	2.8	2.88
Bh	**1.1**	2.43	4.2	4.78
Bc	**2.4**	3.78	4.33	3.2
Rct	**1.7**	3.03	4.8	4.17
Ch	**6.1**	8.1	7.75	8
Cs	4.58	**2.61**	9.5	8.75
Cc	**1.83**	3.78	4.41	6.03
Re	**4.5**	4.7	3.7	5.3
Yp	**0.7**	2	2.4	2.3
Oj	**2.28**	5.63	5.78	6.6

The lowest average fitness value among different methods is indicated in bold values

sequential Bonferroni posthoc test [54] is a step-down procedure that applies a minimum rank strategy to specify whether its performance is statistically significant based on other strategies. The test is performed by pairwise comparisons as follows:

$$z = \frac{(rank_i - rank_j)}{\sqrt{\frac{N(N+1)}{6M}}}, \qquad (21)$$

where N is the number of strategies, and the strategy with minimum Friedman rank is defined as a control strategy. The P value is obtained based on the value of z and normal distribution. In Holm's test, in case the smallest P value is lower than $\frac{\alpha}{N-1}$, the hypothesis is rejected, and the next higher value is checked with $\frac{\alpha}{N-2}$. This process continues

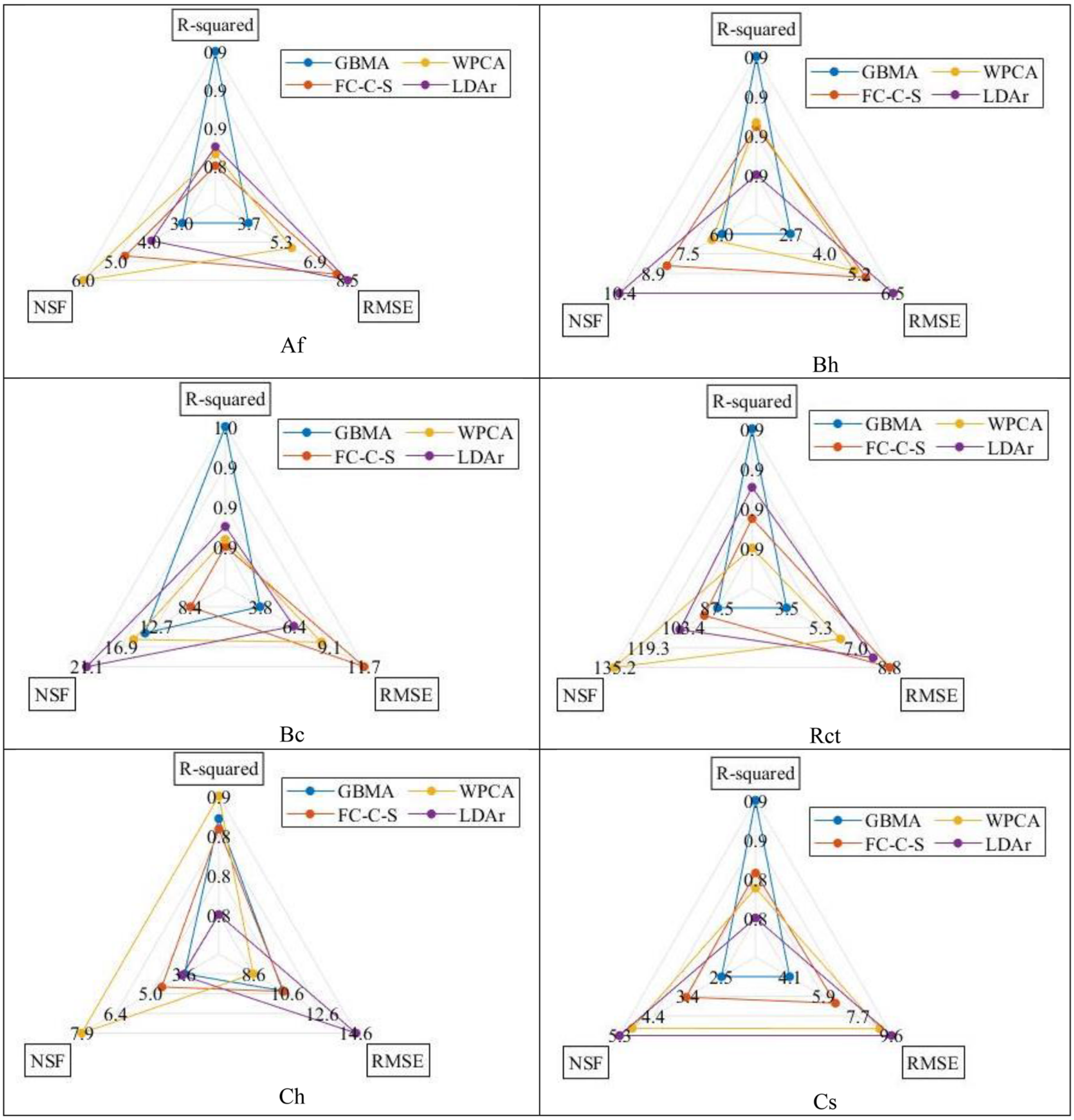


Fig. 15 Spider web diagrams for different regression datasets

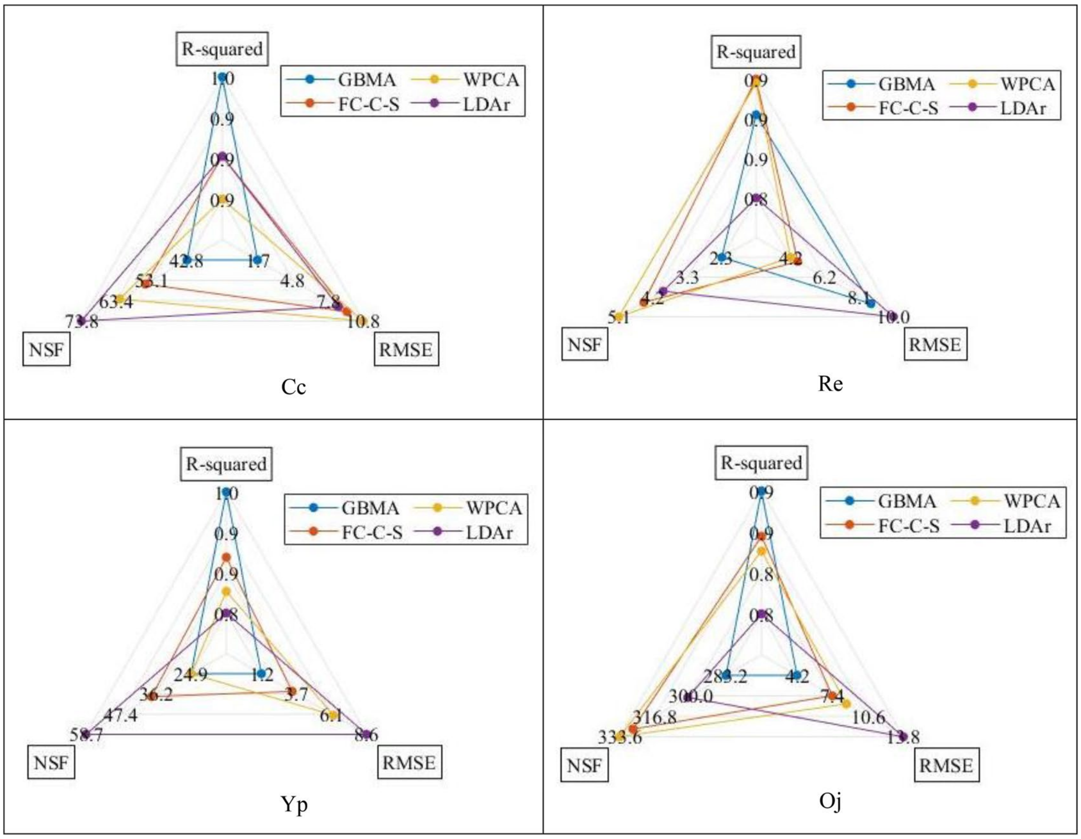


Fig. 15 (continued)

until the hypothesis is accepted or all *P* values are checked. Statistic tests are performed for the results of the compared methods based on two evaluation parameters (i.e., RMSE, and NSF). The statistical tests for classification accuracy are given in Appendix 3.

The compared methods (i.e., CPSOS, MGWO, ABCoDT, WPCA, FC-C-S, and LDAr) are ranked based on their RMSE. Consequently, the method with the lowest rank would perform best. Figure 16 displays the ranking of the compared feature selection methods based on the Friedman test. As seen, GBMA, FC-C-S, and CPSOS ranked first (1.5), second (2.61), and third, respectively, among all the methods. According to Fig. 17, the P values for the error rate values of all feature reduction methods are less than 0.05 (i.e., 0.04, 0.03, 0.001, 0.0003, 0.00004, and 0.00002 for GBMA vs. FC-C-S, GBMA vs. CPSOS, GBMA vs. MGWO, GBMA vs. ABCoDT, GBMA vs. WPCA and GBMA vs. LDAr, respectively). Hence, these results are statistically significant.

For subsequent experiments, the Friedman and Holm tests are investigated for the NSF of all the compared methods (i.e., GBMA, LDA, FC-C-S, WPCA, CPSOS, MGWO, and ABCoDT). Each of the compared methods is run 30 times; the average NSF of each is calculated. Figure 18 shows the Friedman ranks of the 13 methods based on NSF. From Fig. 18, we can see that the NSF through all strategies is significantly different since the attained *P* value of the test is 3.33E−07, which is lower than the desired significance level (i.e., $\alpha = 0.05$). The proposed feature selection method (i.e., the highest-ranking method) is considered as a control strategy in Holm's test to check if its performance statistically differs from other strategies. The results of Fig. 19 demonstrate that the GBMA strategy is significantly better than LDA, FC-C-S, WPCA, CPSOS, and ABCoDT.

7.3 Studying GBMA parameters

A set of extensive experiments have been performed to study the impact of the standard parameters of the GBMA. We test different values for the main parameters in the algorithm (i.e., number of search agents, the maximum number of iterations, and *L* that indicates the step size at each iteration in the migration phase (see Fig. 8). In this section, the datasets are divided into three groups (i.e., small, medium, and large) based on the size of the dimension. The small dataset includes *Af*, *Ch*, Re, *Cs*. The medium dataset

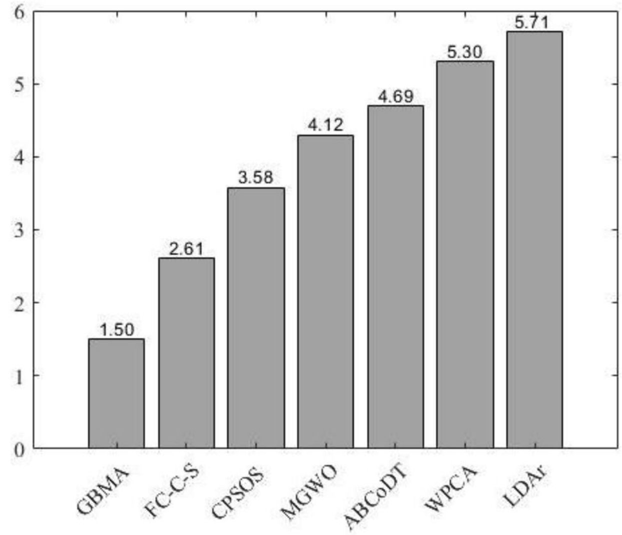


Fig. 16 Friedman ranks for RMSE

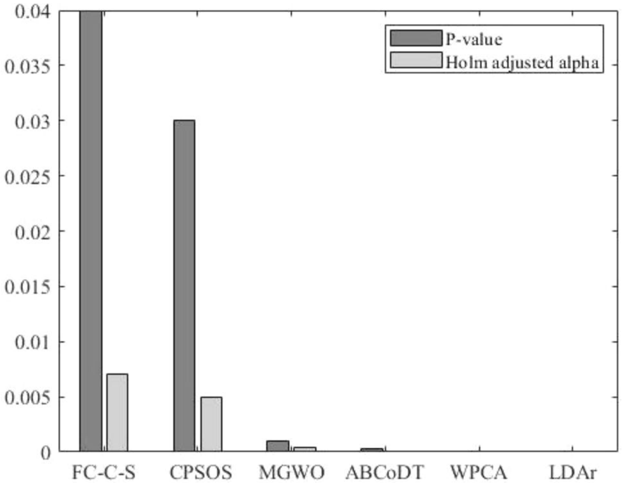


Fig. 17 Holm test for RMSE

includes *Bh*, *Bc*, *Yp,* and the large dataset includes *Cc* and *Rct*. To carry out the experiments, two datasets are randomly selected (i.e., two small datasets, two medium datasets, and two large datasets) to evaluate the different combinations of these parameters.

The parameter "number of search agents (i.e., population size)" is allowed to take five different values (i.e., 30, 40, 50, 60, and 70). Furthermore, the number of iterations is allowed to take four different values (i.e., 30, 50, 70, and 100). The last parameter (i.e., vector *L*) is linearly changed in three different intervals, namely [0.3, 1], [0.3, 2], and [0.3, 4]. A set of independent experiments are performed for each dataset by varying the number of population sizes, number of iterations, and *L* simultaneously to illustrate the effect of these parameters on the performance of the GBMA algorithm. A total of 60 parameter combinations are adopted for each dataset. The algorithm was run ten times for every set of parameter values and each dataset. Then, we calculate the average RMSE and fitness value and compare the results. Figures 20a–f shows RMSE values in terms of the number of iteration and population size for *AF* and *Cs* as two small datasets. Figures 21a–f illustrates RMSE values in terms of the number of iterations and population size for *Bh* and *Bc* as two medium-sized datasets. Figures 22a–f depicts RMSE values in terms of the number of iterations and population size for *Cc* and *Rct* as two large datasets. Figures 20, 21 and 22 show how the performance of GBMA is changed by setting different population sizes while varying other parameters (i.e., number of iteration and vector *L*) for small, medium, and large datasets. The study of main parameters for GBMA on different types of datasets shows that 100 iterations are sufficient to obtain the best results in most cases. Moreover, GBMA with a medium number of agents, between 50 and 60, and the parameter *L* in [0.3, 2] can obtain the lowest RMSE value for most datasets.

All the source codes developed in this research are available via a GitLab repository [55].

8 Conclusions

Dimensionality reduction is a primary data-preprocessing technique in statistics, machine learning, and related fields. Since selecting a compact subset of features can reduce the computational cost and achieve good generalization abilities. Feature selection seeks to minimize two conflicting objectives, namely error rate and the number of features. Many feature selection strategies have been developed to deal with this problem, but the focus is less on regression than on classification. Motivated by this, we presented a novel game-theoretic feature selection strategy, which obtains the optimal solution via BMA optimization in less time. The proposed method puts the most correlated features in one cluster and considers the appropriate order for checking correlations among input vectors. Moreover, three weighting methods (i.e., moderate, strict, and hybrid) were applied to extract new features based on the generated clusters. We tested the performance of the proposed multi-objective algorithm with several well-known datasets and recent feature

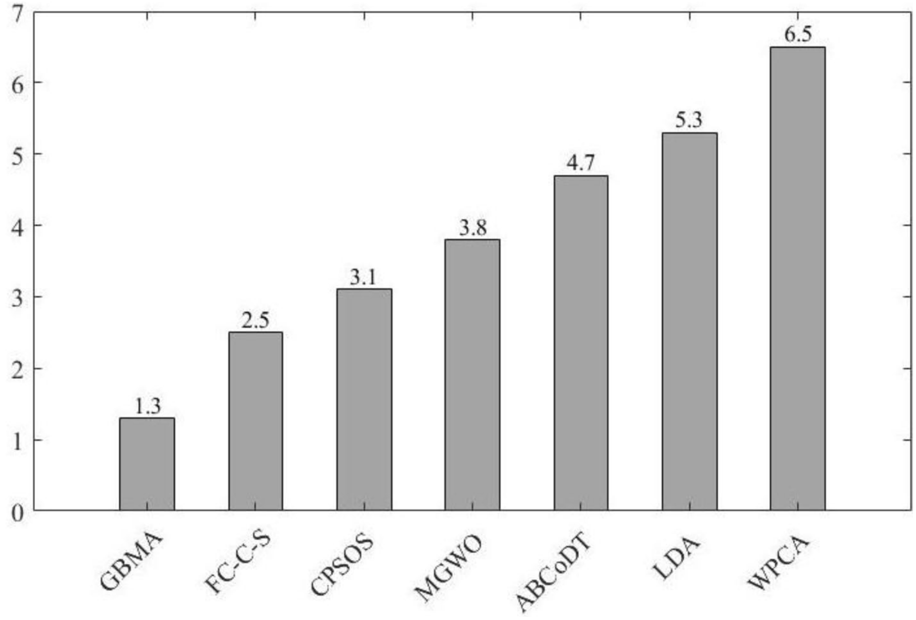


Fig. 18 Friedman ranks for NSF

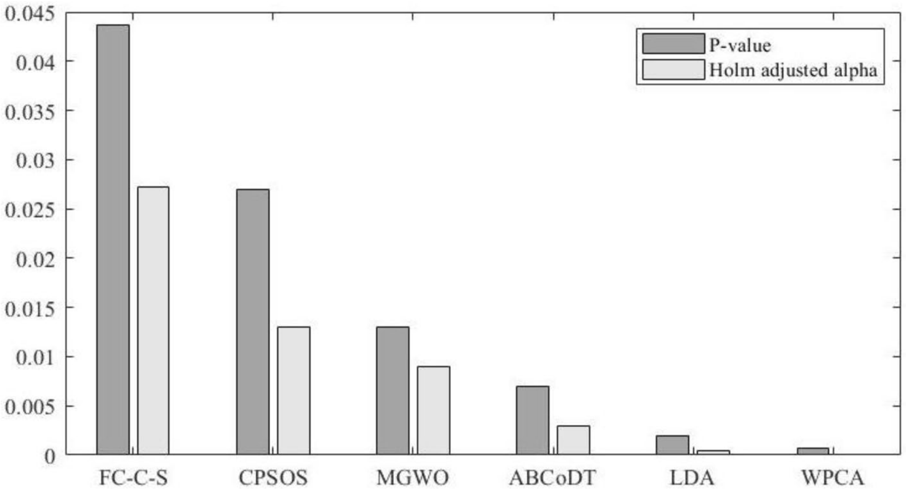


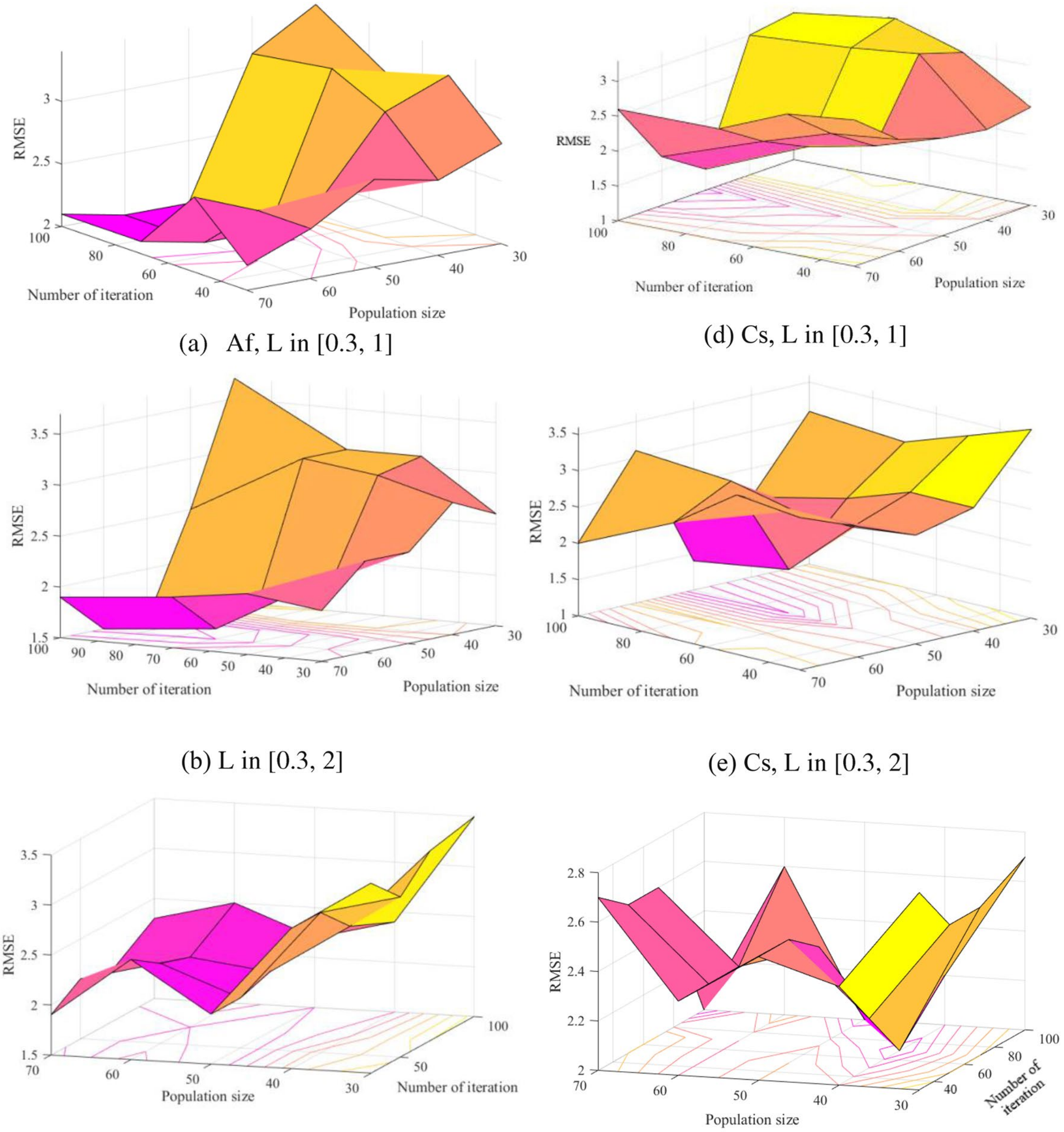
Fig. 19 Holm test for NSF

selection methods. In almost all cases, the proposed method achieves better performance than all the other strategies both in terms of accuracy and the number of features. In other words, it provides a compact subset of features with a high predictive capability. Future work may focus on exploring feature selection issues for an imbalanced and noisy dataset. The proposed feature selection can be tested by some real industry applications, such as the fault diagnosis.

Appendix 1: The performance of swarm-based methods on classification datasets

To evaluate the performance of GBMA for classification problems, ten classification datasets are assumed that are described in Table 8.

GBMA is compared with three swarm-based feature selection strategies for classification datasets as follows:



(a) Af, L in [0.3, 1]

(d) Cs, L in [0.3, 1]

(b) L in [0.3, 2]

(e) Cs, L in [0.3, 2]

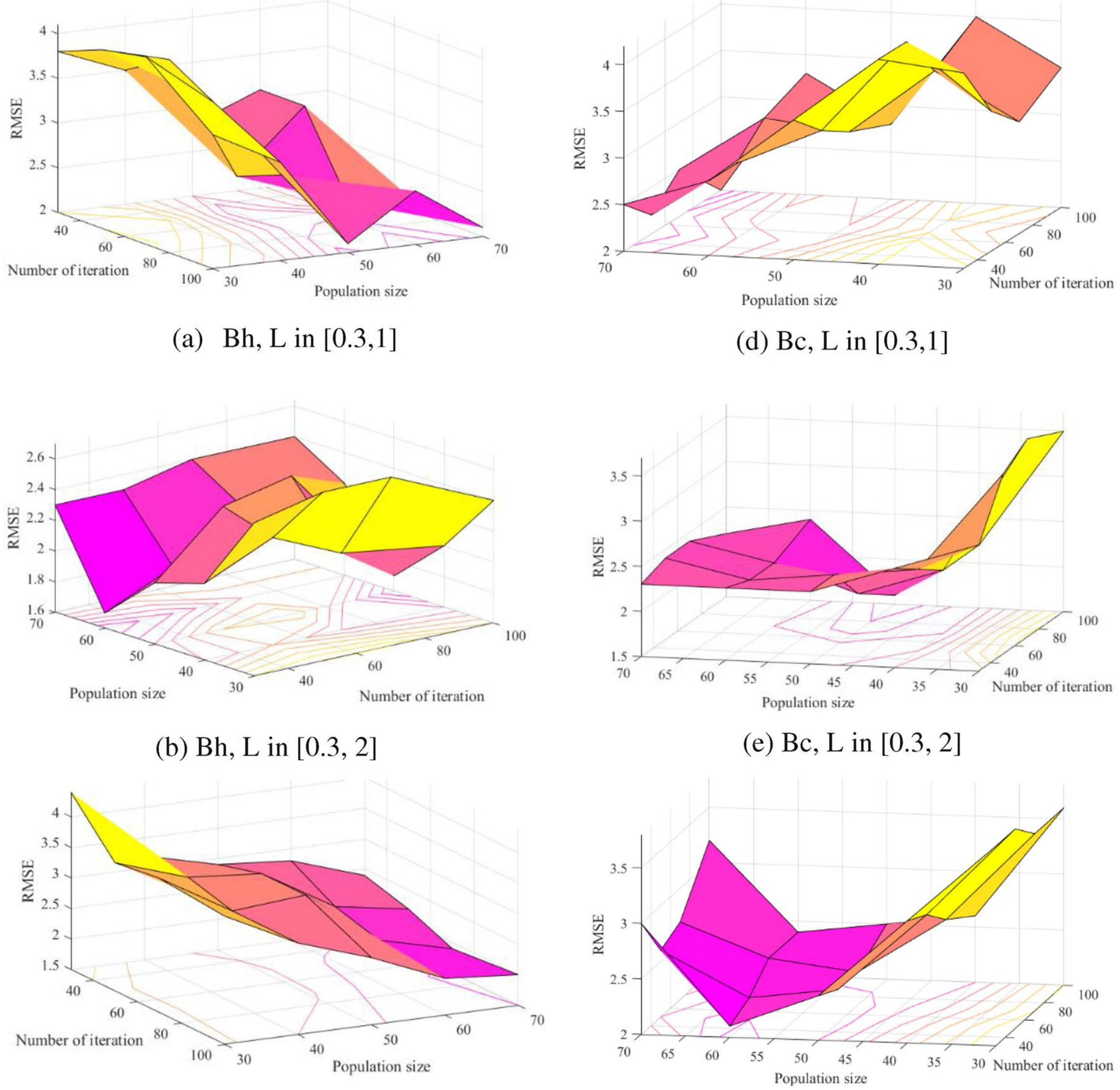
(c) L in [0.3, 4]

(f) Cs, L in [0.3, 4]

Fig. 20 Small datasets investigation

- *Hybrid genetic algorithm* (*HGA*) [56]: It integrates the exploration capability of a genetic algorithm into the exploitation capability of neighborhood local search.
- *Graph clustering-based ant colony optimization* (*GCACO*) [39]: It represents the feature space by dividing the features into some clusters based on a community detection strategy. It then determines the appropriate subset of features using the ant colony-based search approach.
- *Modified firefly algorithm* (*MFA*) [38]): It improves the firefly algorithm by Simulated Annealing (SA) and chaotic diversified search approach to select optimal feature subsets.

They are compared in terms of classification accuracy (CA), the number of selected features (NSF), F-measure, and area under curve (AUC) by ten classification datasets. In this evaluation, the fitness function includes two objectives: (1) the accuracy that is selected by the support vector machine (SVM) and the number of selected features. The fitness function is determined using Eq. (22):



(a) Bh, L in [0.3,1]

(d) Bc, L in [0.3,1]

(b) Bh, L in [0.3, 2]

(e) Bc, L in [0.3, 2]

(b) Bh, L in [0.3, 4]

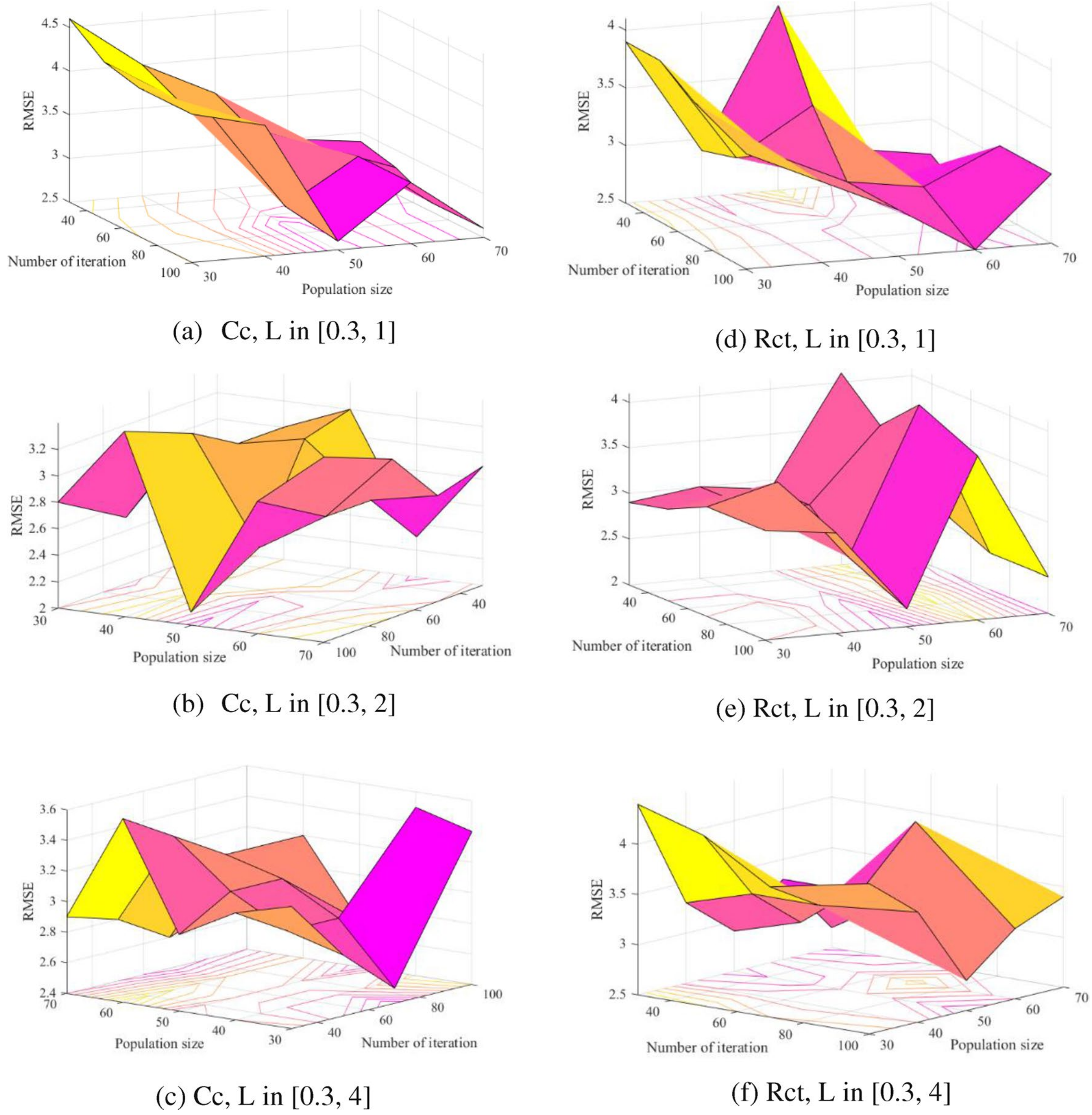
(f) Bc, L in [0.3, 4]

Fig. 21 Medium datasets investigation

$$Fitness(x) = w_1 \times accuracy + w_2 \times NSF, \tag{22}$$

where w_1 and w_2 denote the weights of classification accuracy and $w_1 + w_2 = 1$. Likewise, *NSF* represents the number of selected features. Since the classification accuracy is more important than the NSF, we assign $w_1 = 0.9$ and $w_2 = 0.1$ as recommended in related studies [57].

We must modify players and the fitness function of GBMA to use it for classification problems. Since the original proposed algorithm (i.e., GBMA) is introduced for regression datasets, the number of players is reduced to two. Equation (22) is used as a fitness function. Figure 23 indicates the comparison of the classification performance between GBMA with three feature selection algorithms based on accuracy and the number of features. The horizontal axis and vertical axis show the number of features in rectangular coordinates and accuracy, respectively. The top-left points are the best results since they have higher accuracy and a smaller number of features. The comparison of GBMA with three methods is categorized into three parts. The first part is excellent results obtained for *Ionosphere*, *SpectEW*, *Madelon*, and *KrvskpEW* datasets as GBMA obtains higher accuracy and a smaller number of features. On these datasets, several solutions of GBMA are located at the upper and upper-left of optimal solutions of other algorithms. For the *KrvskpEW* dataset, the optimal solution is the point with 0.925 accuracy and six features created by GBMA. We can see several black points at the upper, meaning the



(a) Cc, L in [0.3, 1]

(d) Rct, L in [0.3, 1]

(b) Cc, L in [0.3, 2]

(e) Rct, L in [0.3, 2]

(c) Cc, L in [0.3, 4]

(f) Rct, L in [0.3, 4]

Fig. 22 Large datasets investigation

Table 8 Classification datasets details [50]

Dataset	#Instances	#Features	#Classes
Zoo	101	16	7
WDBC	569	30	2
KrvskpEW	3196	36	2
Musk version 1	476	166	2
SpectEW	267	22	2
Ionosphere	351	34	2
Satellite	6435	36	6
Madelon	2000	500	2
SonarEW	208	60	2
WaveformEW	5000	40	3

better results of GBMA in comparison with other feature selection algorithms. Put differently, GBMA has better convergence ability for the first objective and better diversity searching process for the second objective. There is one red point belonging to MFA locating under the left part of the figure with an accuracy of 0.85 and 5 number of features.

The second part of competitive results represents that GBMA only achieves higher accuracy value with the same number of features. *Zoo*, *Satellite*, and *Musk* datasets yield these results. For the *Musk* dataset, GBMA finds the optimal solution (i.e., black point) with a 0.96 accuracy and 62 generated features. It can be seen that three black points located at the upper part of blue and red points in the left part of the rectangle, where they are not on the upper left. It proves that

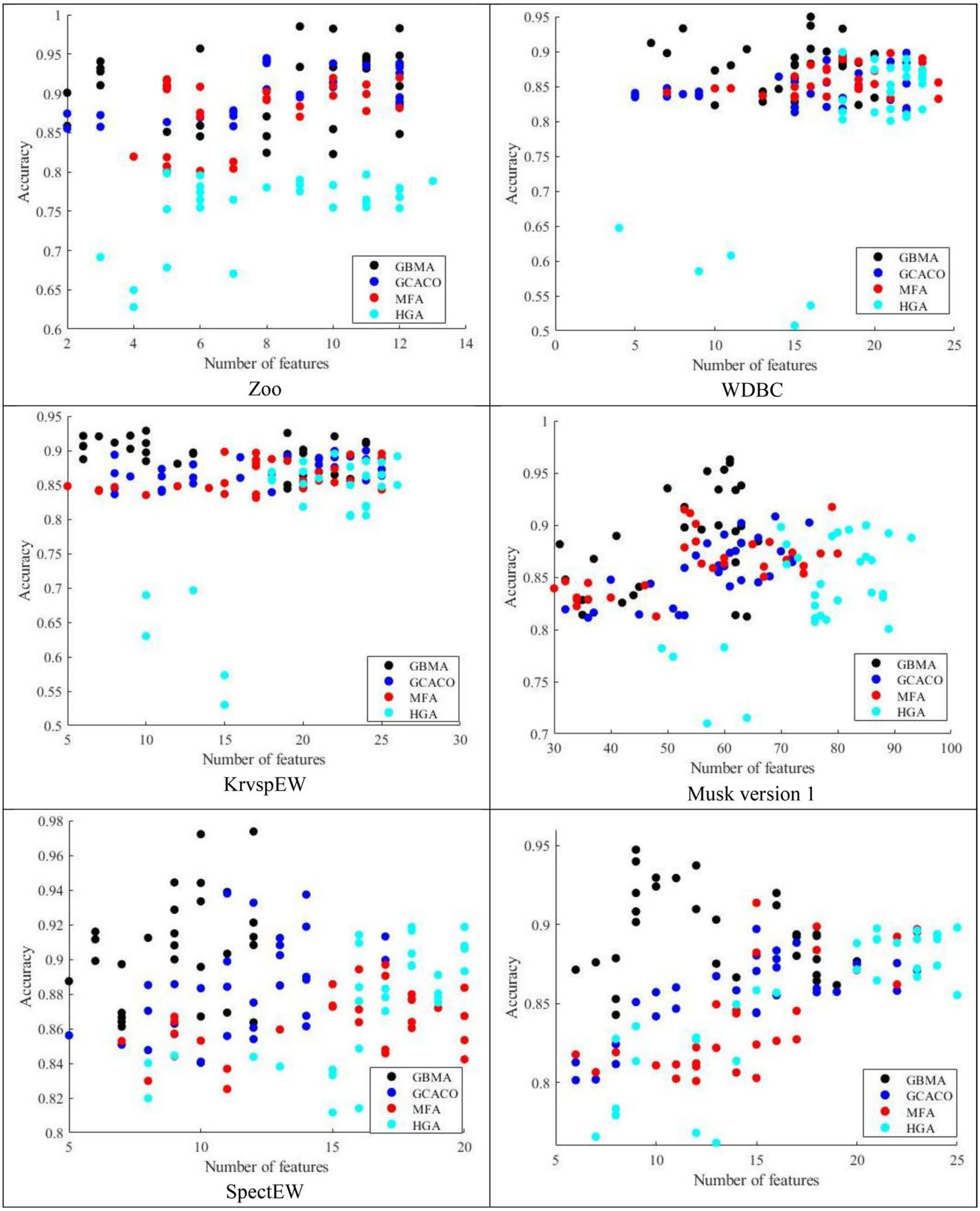


Fig. 23 Plots of the number of the selected features and the corresponding accuracy

GBMA has three solutions better than GCACO and MFA. Although GBMA cannot indicate the solutions with lower dimensions and higher accuracy, it still can search higher accuracy solutions with the same dimensions by the powerful diversity capability.

The third part of the analysis describes the failure results of GBMA. For the *SonarEW* and *WaveformEW* datasets, GBMA has lower accuracy and a larger number of features. This can be mainly ascribed to its inability to conduct a comprehensive evaluation in limited iteration times due to its computation. In

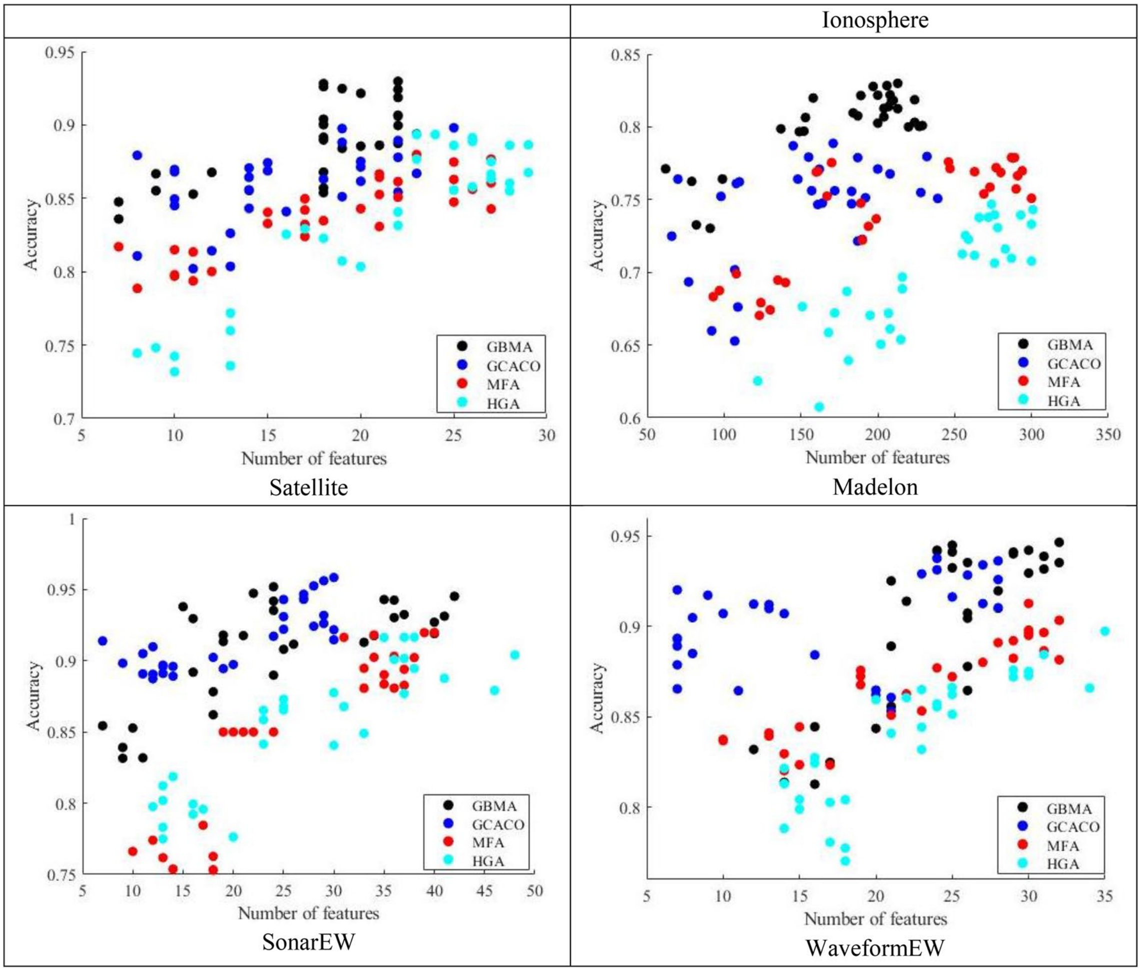


Fig. 23 (continued)

short, GBMA presents better results than other feature selection algorithms through an analysis of this section.

In the next experiment, we depict the areas under ROC curves (AUCs) and F-measure of four methods for ten datasets by the boxes in Fig.

24. AUC is one of the main evaluation metrics for classification problems that shows AUCs. Moreover, F-measure is a commonly used performance metric obtained by the weighted harmonic mean (WHM) of the precision (i.e., positive predictive value or PPV) and recall (i.e., sensitivity). F-measure with value 1 and 0 indicate the best and worst results, respectively. Box plots are used to provide a stability analysis of the proposed feature selection method (i.e., GBMA) and other compared methods. The empirical distribution, variation information, median, symmetry, and skewness on the dataset can be easily represented by box plots.

From Fig. 24, we can observe that the proposed algorithm gives superior and consistent results for all the evaluated datasets except *SonarEW* and *WaveformEW,* where GCACO has better performance. GCACO improves AUC and F-measure by 7% and 8%, respectively, compared to GBMA. This is because the migration phase after two random neighbors is used for updating the solution. Then, the positions of solutions (sequencing and weighting parts) are close to each other, and a new solution has not been practically produced. Hence, the probability of being trapped in local optimum has been increased.

The proposed algorithm (i.e., GBMA) outperforms HGA on all datasets and improves AUC and F-measure by 34% and 38%, respectively. The main reason is that HGA cannot determine near-optimal solutions and cannot perform exploitation very well, and it may get stuck in local optima. The results obtained from Fig. 24 are listed as follows:

- The median of F-measure and AUC for GBMA is often higher than the other feature selection algorithms.
- The AUC and F-measure of GBMA are higher than 85% for all evaluated datasets.

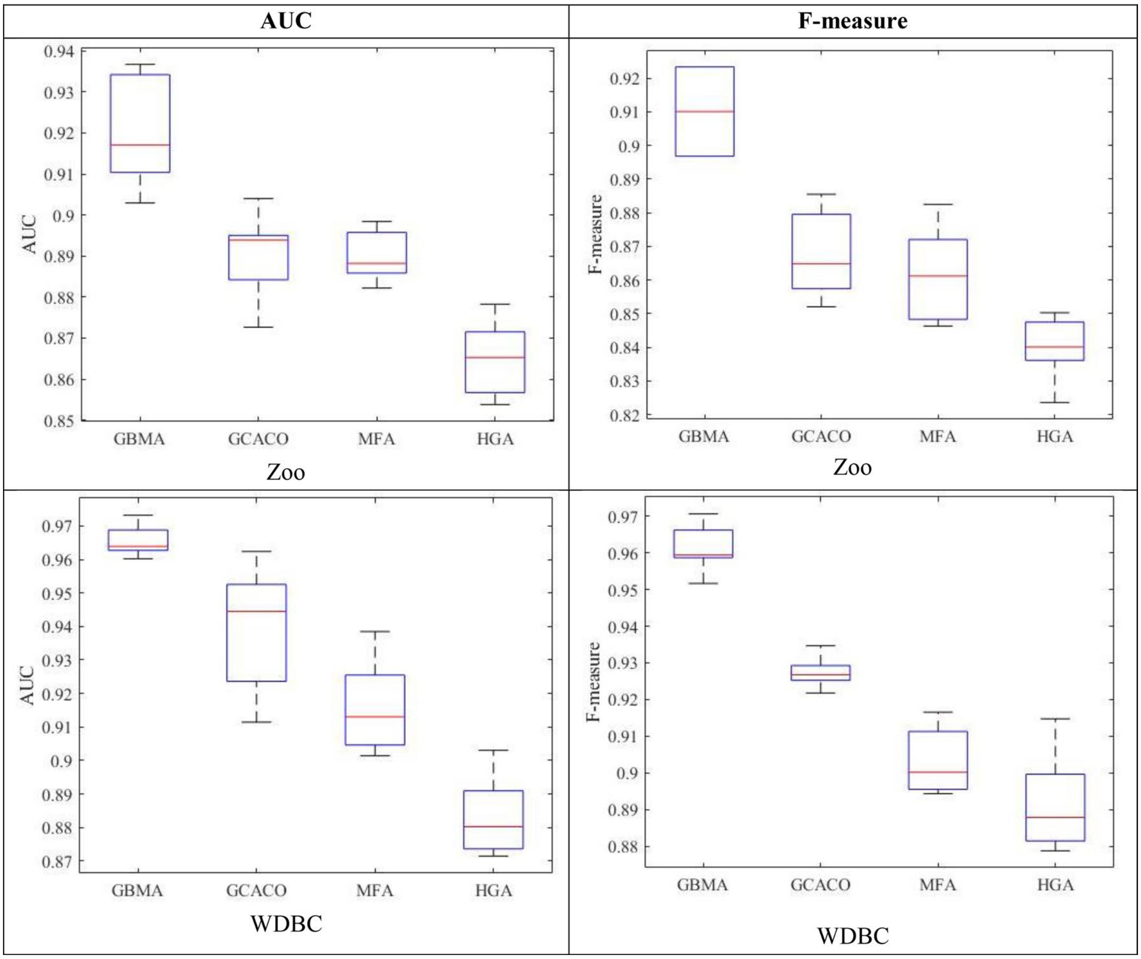


Fig. 24 AUC and F-measure for classification datasets

- The difference between the minimum and maximum AUC and F-measure for GBMA is very insignificant (less than 10%) in most datasets.
- These results prove the superiority of GBMA in selecting the prominent feature set.

Table 9 presents the average computation times of four algorithms (i.e., GBMA, GCACO, MFA, and HGA) over 30 independent runs to indicate how fast an algorithm implements the search process. As indicated, GBMA has the lowest average computation time (bold numbers) compared to the other methods for seven datasets. On two of the ten datasets, HGA requires less computation time than the other methods, and MFA requires less time for one dataset.

Additionally, Table 9 ranks the performance of the methods based on their average computation times. We can observe that GBMA exhibits the lowest average computation time overall, followed by MFA, HGA, and GCACO. Table 9 represents the results of three groups of Wilcoxon rank-sum tests to compare two related methods. Frank Wilcoxon [58] introduced the Wilcoxon rank-sum test that assigns ranks to all the scores of one set and then sums the ranks in each set. This test is defined as the nonparametric version of the *t* test for two independent sets, and any difference in the two rank sums appears due to sampling error.

According to Table 9, GBMA outperforms GCACO, MFA, and HGA. The proposed method has superiority over all other methods on all datasets except for MFA on *KrvskEW* and HGA on *SpectEW* at a significance level of 0.05 and similar results for HGA on the *Zoo* dataset. Moreover, the results of Table 9 are obtained over 30 runs for each evaluated strategy. While the number of runs in which GBMA outperforms GCACO, MFA, and HGA in terms of CA is 12, 14, and 19, respectively, these results are represented in the last row of the Wilcoxon rank-sum table.

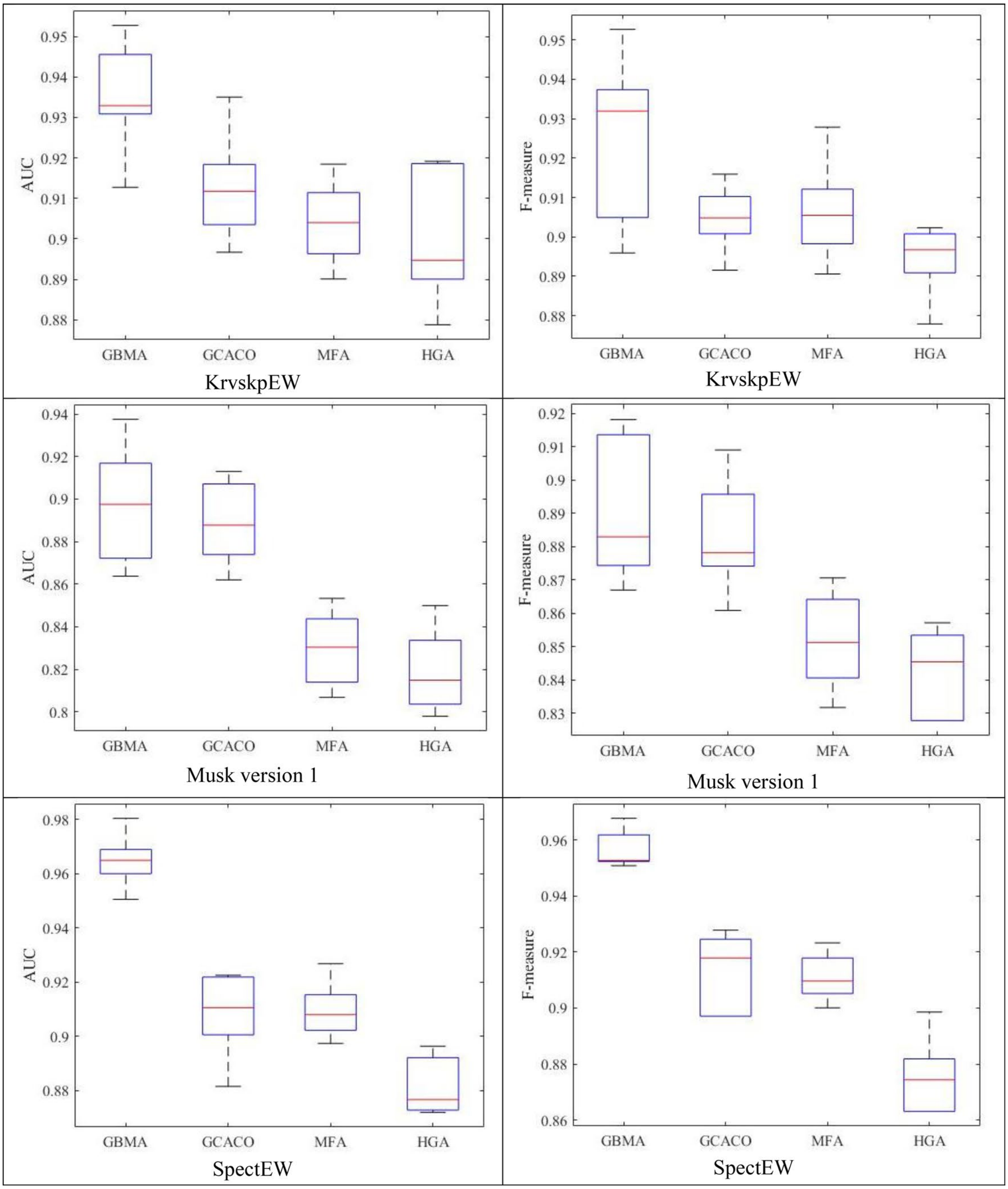


Fig. 24 (continued)

Appendix 2: The performance of filter methods on classification datasets

The proposed method is compared with three filter methods for classification datasets as follows:

- *Cooperative game theory with relief approach* (*CGTR*) [59]: It combines the game theory with a relief method to evaluate the contribution of features.
- *Fisher-score* (*F-score*) [60]: It selects a feature subset such that the between-class scatter is maximized while the within-class scatter is minimized.

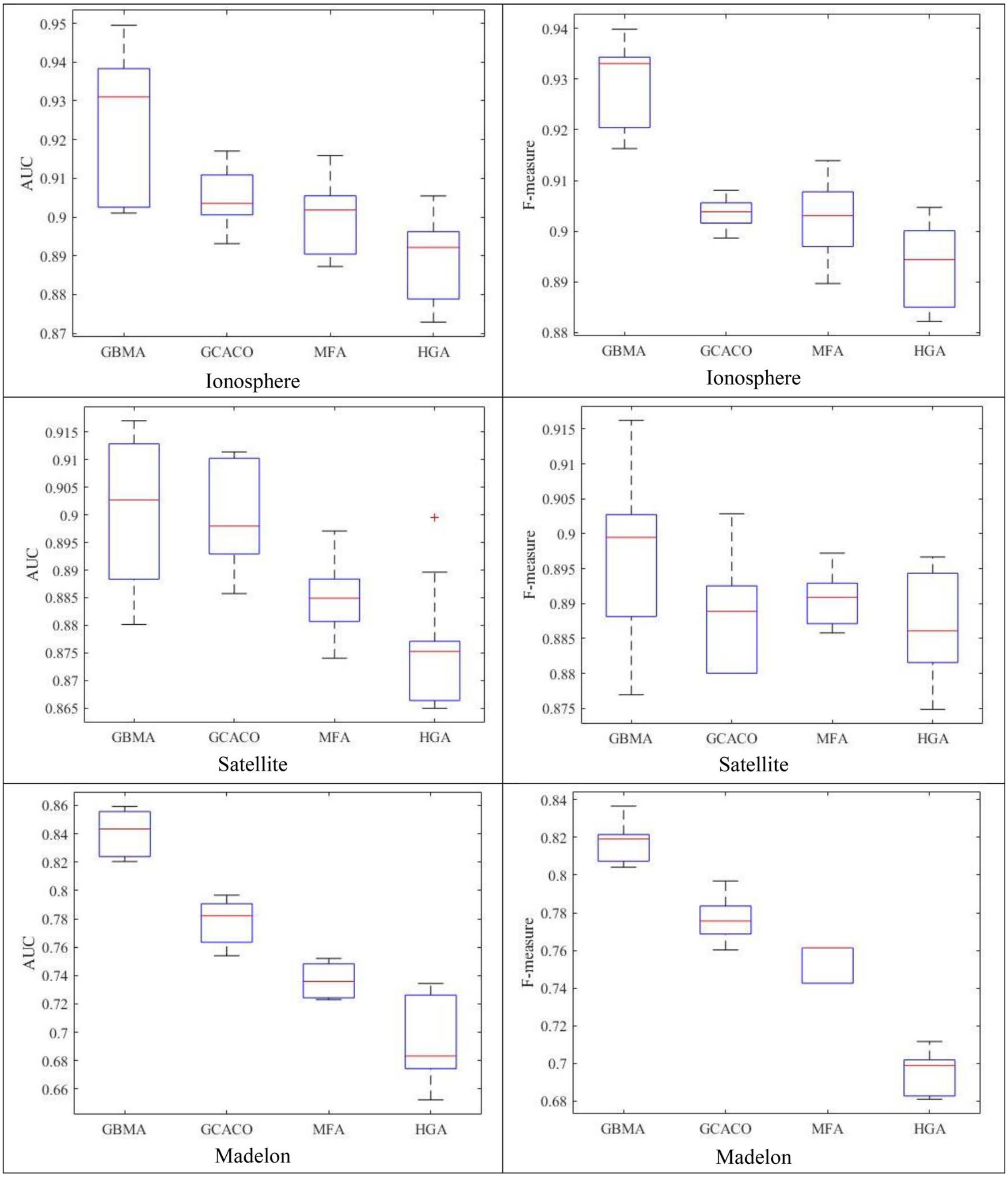


Fig. 24 (continued)

- *Correlation-based feature selection* (*CFC*) [61]: It applies a correlation-based strategy with less computation to evaluate the importance of features.

The proposed algorithm is evaluated in terms of classification accuracy (CA), the number of selected features (NSF), F-measure, and area under curve (AUC). The performance of the compared methods on different classification datasets can be seen from Fig. 25. For most datasets, the proposed method (i.e., GBMA) achieves better results. For example, on *KrvspEW*, GBMA obtains 0.98, 13.2, 0.976, and 0.982 for CA, NSF, F-measure, and AUC, respectively. While CGTR obtains 0.91, 18.2, 0.92, and 0.915. Thus, GBMA improves CA and NSF by 8% and 30%, respectively, compared to CGTR. The main weakness of the CGTR algorithm is that it uses the information gain as an evaluation parameter of the correlation between features. Hence, the relevant features cannot be

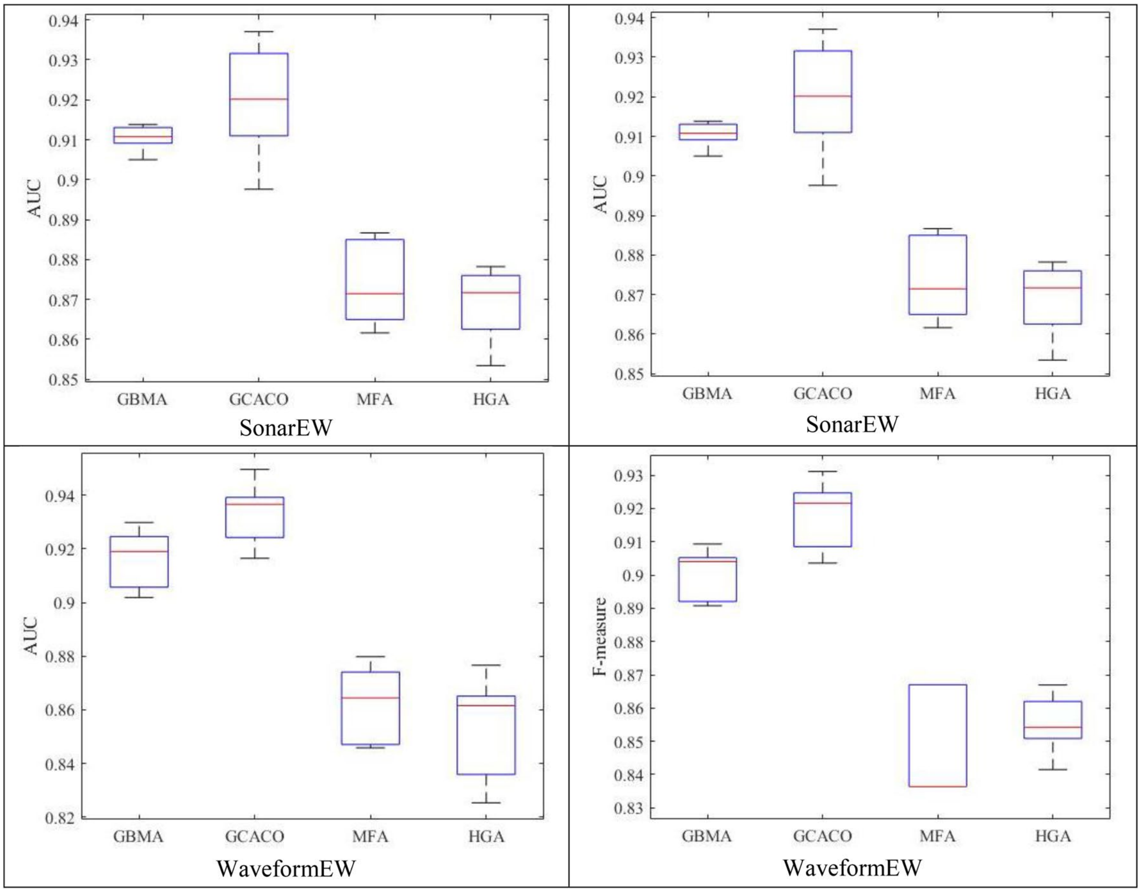
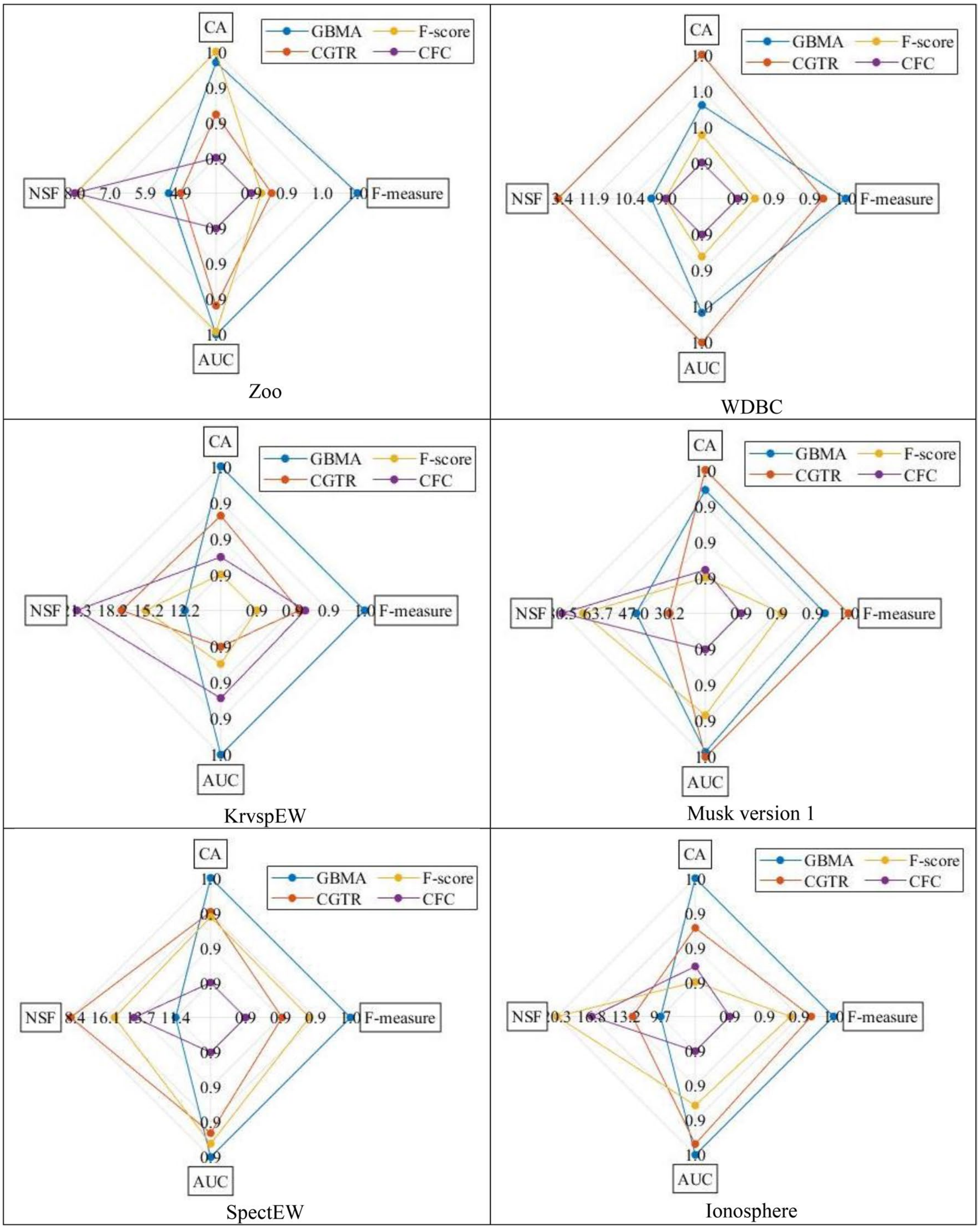


Fig. 24 (continued)

Table 9 Average computation time and Wilcoxon rank-sum for results of methods

Method	Computational time (s)				Wilcoxon rank-sum		
nnDataset	GBMA	GCACO	MFA	HGA	GCACO	MFA	HGA
Zoo	2.50	2.73	3.00	**2.50**	+	+	=
WDBC	**3.1**	3.15	3.22	3.18	+	+	+
KrvskpEW	38.31	52.7	**37.50**	48.64	+	−	+
Musk version1	**6.08**	6.28	6.84	6.5	+	+	+
SpectEW	2.72	2.78	2.81	**2.64**	+	+	−
Ionosphere	**3.03**	3.1	3.14	3.22	+	+	+
Satellite	**170.50**	198.20	191.60	175.70	+	+	+
Madelon	**240.4**	339.7	250.2	258.7	+	+	+
SonarEW	**76.3**	78.2	78.23	79.4	+	+	+
WaveformEW	**32.78**	37.82	35.1	34.25	+	+	+
Average	57.58	72.46	61.16	61.47	+	+	+
Rank	1	4	2	3	19	14	12

specified correctly on datasets that have features with a large number of distinct values such as *Musk* and *KrvskpEW*. Besides, GBMA improves AUC and F-measure by 22% and 25%, respectively, in comparison with F-score. The disadvantage of the F-score is that it does not reveal the mutual information among features. In other words, it considers the discriminative power of each feature independently rather than together, rendering it unable to select appropriate features.



Zoo

WDBC

KrvspEW

Musk version 1

SpectEW

Ionosphere

Fig. 25 Spider web diagrams for different classification datasets

Appendix 3: Statistical tests for swarm-based and filter methods

The Friedman and Holm tests are performed for swarm-based classification methods (i.e., GBMA, HGA, GCACO, and MFA) and filter methods (i.e., GBMA, CGTR, and F-score) in terms of classification accuracy (CA) and the number of selected features (NSF). Each compared method is run 30 times, and the average classification accuracy for each method is calculated.

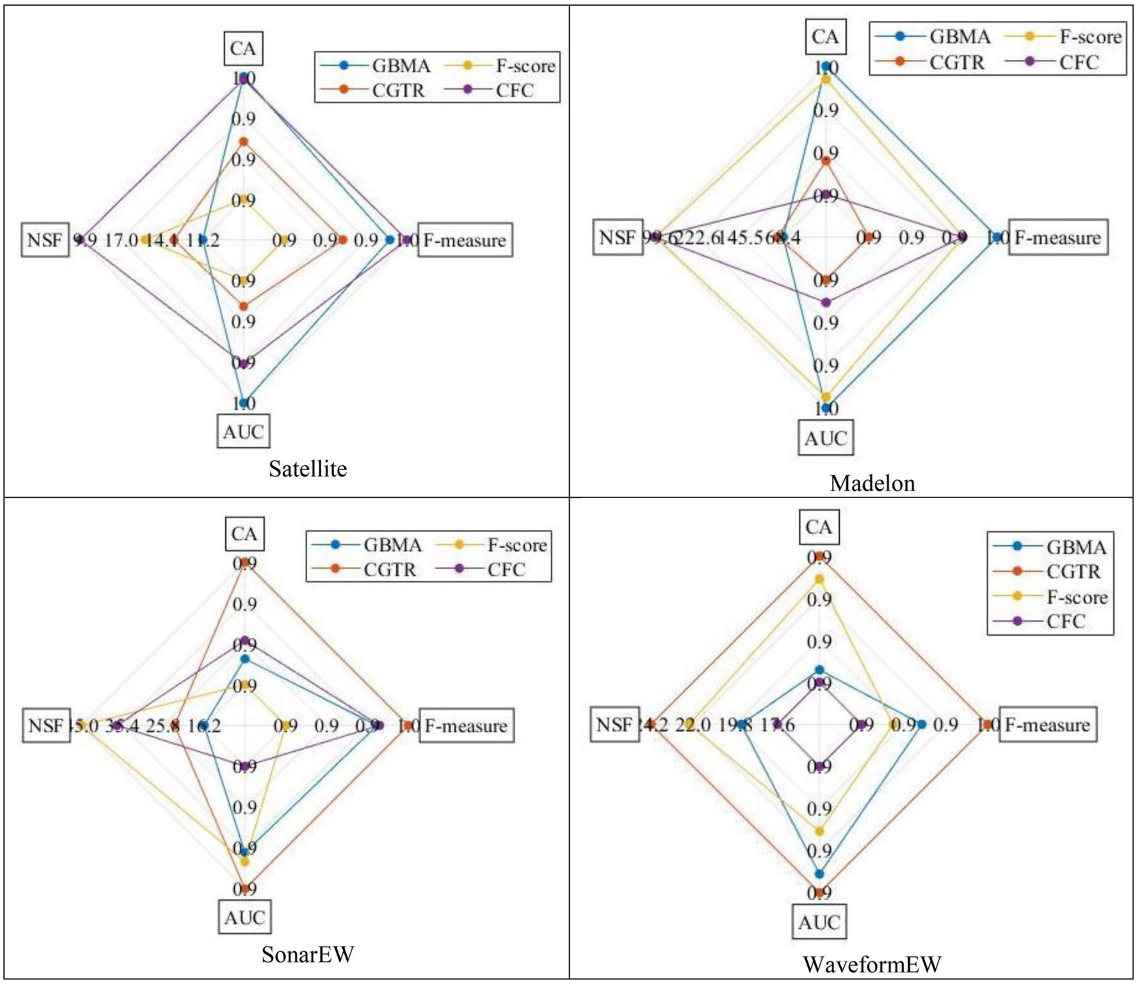


Fig. 25 (continued)

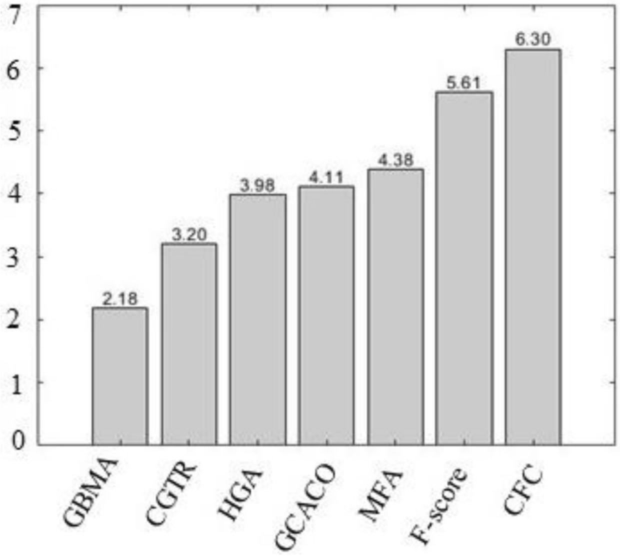


Fig. 26 Friedman ranks based on CA

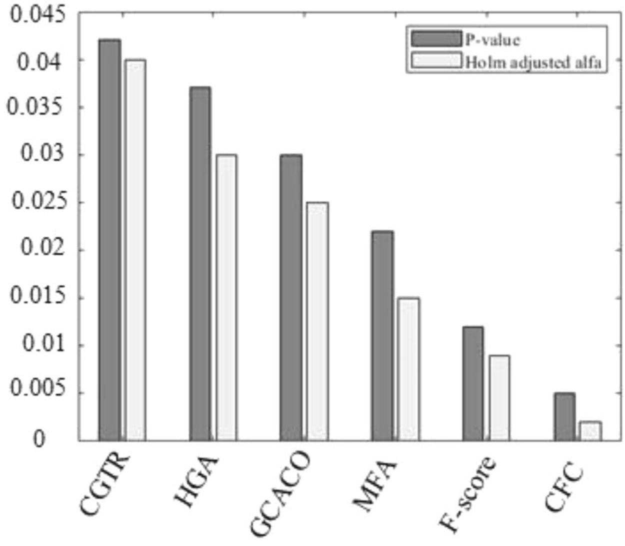


Fig. 27 Holm test based on CA

Fig. 28 Friedman ranks based on NSF

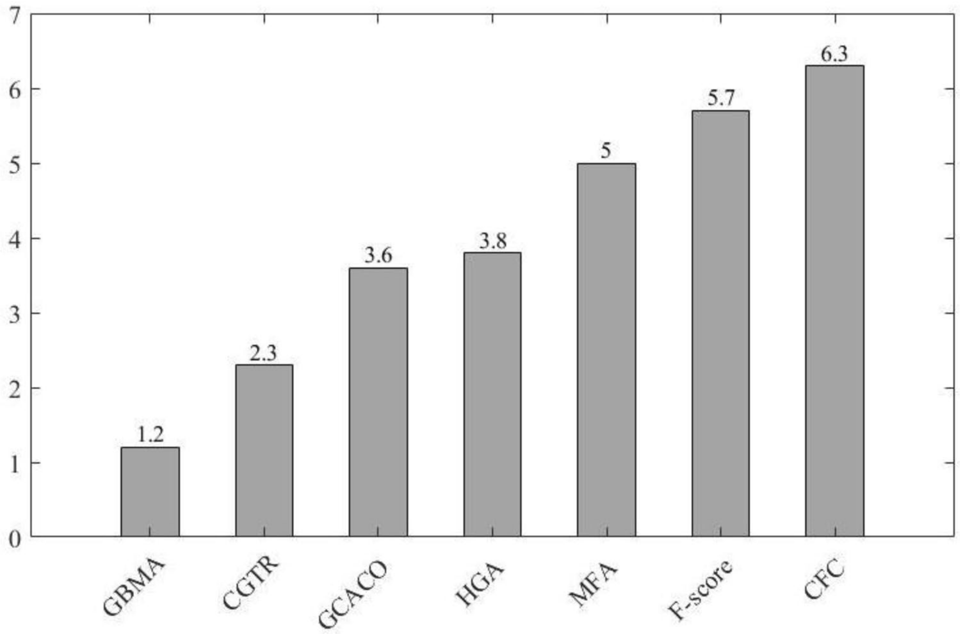


Fig. 29 Holm test based on NSF

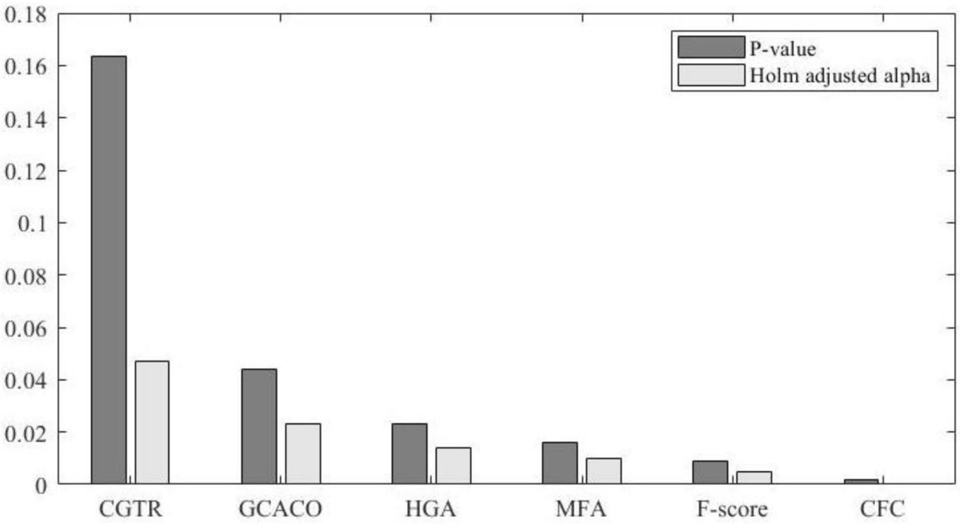


Figure 26 depicts Friedman's ranks (i.e., the vertical axis) of seven methods to analyze the outcomes of experiments. There are significant differences between the classification accuracies of the methods since the obtained *P* value is 8.70E−8, which is lower than the desired significance level (i.e., α = 0.05). The GBMA strategy achieves the highest rank with a difference of 0.95 from the subsequent strategy. Figure 27 displays Holm's sequential Bonferroni posthoc test, revealing that the method with the best performance (i.e., GBMA) performs as a control method. Notably, GBMA has yielded significantly better results than CGTR, HGO, GCACO, MFO, F-score, and CFC.

Furthermore, Friedman and Holm's sequential Bonferroni posthoc tests are applied for the NSF in Fig. 25. Figure 28 exhibits Friedman ranks of the seven methods. The results show that GBMA has a minimum number of features, whereas CFC has a maximum NSF. The proposed method selects a lower number of features by 36% compared to CFC. This significant difference in selected features could play a vital role in the running time of these methods. The P value obtained for the test is 4.13E−08, which is less than the presumed significance level, i.e., α = 0.05. Thus, the NSF by all methods is significantly different. Obtaining the best rank, GBMA generally acts as a control method in Holm's sequential Bonferroni posthoc test. The results in Fig. 29 suggest that the proposed method is significantly better than all methods save for CGTR.

References

1. Sen P, Namata G, Bilgic M, Getoor L, Gallagher B, Eliassi Rad T (2008) Collective classification in network data. AI Mag 29(3):93–106
2. Bhagat S, Cormode G, Muthukrishnan S (2011) Node classification in social networks. In: Aggarwal C (ed) Social network data analytics. Springer, Boston, pp 115–148
3. Liben-Nowell D, Kleinberg J (2007) The link-prediction problem for social networks. J Assoc Inf Sci Technol 58(7):1019–1031
4. Ou M, Cui P, Pei J, Zhang Z, Zhu W (2016) Asymmetric transitivity preserving graph embedding. In: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining, pp 1105–1114
5. Grover A, Leskovec J (2016) node2vec: Scalable feature learning for networks. In: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining, pp 855–864
6. Gong M, Yao C, Xie Y, Xu M (2020) Semi-supervised network embedding with text information. Pattern Recognit 104:107347
7. Shi M, Tang Y, Zhu X (2019) MLNE: Multi-label network embedding. IEEE Trans Neural Netw Learn Syst 1–14
8. Shi M, Tang Y, Zhu X, Liu J, He H (2020) Topical network embedding. Data Min Knowl Disc 34(1):75–100
9. Liu Y, Nie F, Gao Q, Gao X, Han J, Shao L (2019) Flexible unsupervised feature extraction for image classification. Neural Netw 115:65–71
10. Wang K-J, Chen K-H, Angelia M-A (2014) An improved artificial immune recognition system with the opposite sign test for feature selection. Knowl Based Syst 71:126–145
11. Marill T, Green D (1963) On the effectiveness of receptors in recognition systems. IEEE Trans Inf Theory 9:11–17
12. Hancer E, Xue B, Zhang M, Karaboga D, Akay B (2018) Pareto front feature selection based on artificial bee colony optimization. Inf Sci 422:462–479
13. Ma B, Xia Y (2017) A tribe competition-based genetic algorithm for feature selection in pattern classification. Appl Soft Comput 58:328–338
14. Mansouri N, Mohammad Hasani Zade B, Javidi MM (2019) Hybrid task scheduling strategy for cloud computing by modified particle swarm optimization and fuzzy theory. Comput Ind Eng 130:597–633
15. Mahdavi Jafari M, Khayati GR (2018) Prediction of hydroxyapatite crystallite size prepared by sol–gel route: gene expression programming approach. J Sol Gel Sci Technol 86(1):112–125
16. Xu RF, Lee SJ (2015) Dimensionality reduction by feature clustering for regression problems. Inf Sci 299:42–57. https://doi.org/10.1016/j.ins.2014.12.003
17. Zhang Q, Wang R, Yang J, Lewis A, Chiclana F, Yang S (2019) Biology migration algorithm: a new nature-inspired heuristic methodology for global optimization. Soft Comput 23(16):7333–7358. https://doi.org/10.1007/s00500-018-3381-9
18. Mirjalili S, Lewis A (2016) The whale optimization algorithm. Adv Eng Softw 95:51–67
19. Yang X-S (2010) A new metaheuristic bat-inspired algorithm. In Nature inspired cooperative strategies for optimization (NISCO 2010), studies in computational intelligence. Springer, Berlin, pp 65–74
20. Li XT, Zhang J, Yin MH (2014) Animal migration optimization: an optimization algorithm inspired by animal migration behavior. Neural Comput Appl 24(7–8):1867–1877
21. Yang X-S, Deb S (2010) Engineering optimization by cuckoo search. Int J Math Model Numer Optim 1(4):330–343
22. Rashedi E, Nezamabadi-Pour H, Saryazdi S (2009) GSA: a gravitational search algorithm. Inf Sci 179(13):2232–2248
23. Simon D (2008) Biogeograph-based optimization. IEEE Trans Evol Comput 12(6):702–713
24. Eberhart RC, Kennedy J (1995) A new optimizer using particle swarm theory. In: Proceedings of the sixth international symposium on micro machine and human science, pp 39–43
25. Yasini S, Sitani M B N, Kirampor A (2016) Reinforcement learning and neural networks for multi-agent nonzero-sum games of nonlinear constrained-input systems. Int J Mach Learn Cybern 7:967–980
26. Yang J, Jiang B, Lv Z, Raymond Choo KK (2020) A task scheduling algorithm considering game theory designed for energy management in cloud computing. Future Gen Comput Syst 105:985–992
27. Peng X, Xu D (2013) A local information-based feature-selection algorithm for data regression. Pattern Recogn 46:2519–2530
28. Wang L, Zhu J, Zou H (2006) The doubly regularized support vector machine. Stat Sin 16(2):589–615
29. Berrendero JR, Cuevas A, Torrecilla JL (2016) The mRMR variable selection method: a comparative study for functional data. J Stat Comput Simul 86(5):891–907
30. Kira K, Rendell LA (1992) The feature selection problem: traditional methods and a new algorithm. In: Proceedings of ninth national conference on AI, pp 129–134
31. Fukunaga K (1990) Introduction to statistical pattern recognition, 2nd edn. Academic Press, New York
32. Kwak N, Lee JW (2010) Feature extraction based on subspace methods for regression problems. Neurocomputing 73(10–12):1740–1751
33. Kwak N, Kim C (2006) Dimensionality reduction based on ICA for regression problems. In: Proceedings of the international conference on artificial neural networks, pp 1–10
34. Robnik Sikonja M, Kononenko I (1997) An adaptation of relief for attribute estimation in regression. In: Proceedings of the fourteenth ICML, pp 296–304
35. Arauzo-Azofra A, Manuel Benitez J, Castro JL (2004) A feature set measure based on relief. In: Proceedings of the fifth international conference on recent advances in soft computing, pp 104–109
36. Radovic M, Ghalwash M, Filipovic N, Obradovic Z (2017) Minimum redundancy maximum relevance feature selection approach for temporal gene expression data. BMC Bioinform 18:1
37. Rao H, Shi X, Rodrigue AK, Feng J, Yuan X, Elhoseny M, Yuan X, Gu L (2019) Feature selection based on artificial bee colony and gradient boosting decision tree. Appl Soft Comput 74:634–642
38. Zhang L, Mistry K, Peng Lim C, Neoh SC (2018) Feature selection using firefly optimization for classification and regression models. Decis Support Syst 106:64–85
39. Ghimatgar H, Kazemi K, Helfroush MS, Aarabi A (2018) An improved feature selection algorithm based on graph clustering and ant colony optimization. Knowl Based Syst 159:270–285
40. Ding W, Lin CT, Prasad M (2018) Hierarchical co-evolutionary clustering tree-based rough feature game equilibrium selection and its application in neonatal cerebral cortex MRI. Expert Syst Appl 101:243–257
41. Liu G, Xiao Z, Hua Tan G, Li K, Chronopoulos AT (2020) Game theory-based optimization of distributed idle computing resources in cloud environments. Theor Comput Sci 806:468–488

42. Cheng FY (1999) Multiobjective optimum design of structures with genetic algorithm and game theory: application to life-cycle cost design. Computational mechanics in structural engineering. Elsevier, Amsterdam, pp 1–6
43. Périaux J, Chen HQ, Mantel B, Sefrioui M, Sui HT (2001) Combining game theory and genetic algorithms with application to DDM-nozzle optimization problems. Finite Elem Anal Des 37(5):417–429
44. Kwak SK, Kim JH (2017) Statistical data preparation: management of missing values and outliers. Korean J Anesthesiol 70(4):407–411
45. Gibert K, Marrè MS, Izquierdo J (2016) A survey on pre-processing techniques: relevant issues in the context of environmental data mining. AI Commun 29:627–663
46. Leavline EJ, Singh D (2016) Model-based outlier detection system with statistical preprocessing. J Mod Appl Stat Methods 15(1):789–801
47. Kang H (2013) The prevention and handling of the missing data. Korean J Anesthesiol 64(5):402–406
48. Moradi P, Gholampour M (2016) A hybrid particle swarm optimization for feature subset selection by integrating a novel local search strategy. Appl Soft Comput 43:117–130
49. Li D (2014) Cooperative quantum-behaved particle swarm optimization with dynamic varying search areas and Lévy flight disturbance. Sci World J
50. UCI Dataset (2019) https://archive.ics.uci.edu/ml
51. Tian D, Zhao X, Shi Z (2019) Chaotic particle swarm optimization with sigmoid-based acceleration coefficients for numerical function optimization. Swarm and evolutionary computation, 51. Elsevier, Amsterdam
52. Mittal N, Singh U, Sohi BS (2016) Modified grey wolf optimizer for global engineering optimization. Applied computational intelligence and soft computing. Springer, New York
53. Mateos-García D, García-Gutiérrez J, Riquelme-Santos JC (2016) An evolutionary voting for k-nearest neighbours. Expert Syst Appl 43:9–14
54. Weston J, Mukherjee S, Chapelle O, Pontil M, Poggio T, Vapnik V (2001) Feature selection for SVMs. In: Advances in neural information processing systems, pp 668–674
55. https://github.com/javidi7/Feature-selection
56. Yu X, Zhou Y, Liu XF (2019) A novel hybrid genetic algorithm for the location routing problem with tight capacity constraints. Appl Soft Comput J 85:105760
57. Mistry K, Zhang L, Neoh SC, Lim CP, Fielding B (2017) A micro-GA embedded PSO feature selection approach to intelligent facial emotion recognition. IEEE Trans Cybern 47(6):1496–1509
58. Wilcoxon F (1945) Individual comparisons by ranking methods. Biometr Bull 1(6):80–83
59. Gore S, Govindaraju V (2016) Feature selection using cooperative game theory and relief algorithm. In: 8th International conference on knowledge, information, and creativity support systems, pp 401–412
60. Duda RO, Hart PE, Stork DG (2012) Pattern classification. Wiley, New York
61. Hall MA, Smith LA (1999) Feature selection for machine learning: comparing a correlation-based filter approach to the wrapper. In: Proceedings of the twelfth international FLAIRS conference