On the potential of the nature-inspired algorithms for pure binary classification

Iztok Fister Jr.[1], Iztok Fister[1], Dušan Fister[1], Grega Vrbančič[1], Vili Podgorelec[1] *

University of Maribor, Faculty of Electrical Engineering and Computer Science,
Koroška cesta 46, Slovenia

Abstract. With the advent of big data, interest for new data mining methods has increased dramatically. The main drawback of traditional data mining methods is the lack of comprehensibility. In this paper, the firefly algorithm was employed for standalone binary classification, where each solution is represented by two classification rules that are easy understandable by users. Implicitly, the feature selection is also performed by the algorithm. The results of experiments, conducted on three well-known datasets publicly available on web, were comparable with the results of the traditional methods in terms of accuracy and, therefore, the huge potential was exhibited by the proposed method.

Keywords: firefly algorithm, data mining, binary classification

1 Introduction

Data Mining is the most complex part of the Knowledge Discovery from Data (KDD) process that is comprised of: Data selection and creation, preprocessing (i.e., data cleaning and transformation), data mining, and evaluation [9]. Typically, the data preprocessing captures the feature extraction and feature selection. The aim of the former is to select the subset of attributes in a dataset, while the latter to find the best subset of features from a set of features. Classification and clustering are two of the most widely studied tasks of data mining, where the classification is referred to a prediction of the class labels on the basis of test observations during the process of learning [16].

Mainly, the traditional classification methods based on Decision Trees [14], Bayesian networks [6], Neural Networks [7], and Support Vector Machines [8]. Although these methods are able to find the local optimal classification models in some situations, the majority of them are not very comprehensible, and thus are hard to handle by the ordinal users. Usually, they are also too time consuming. Fortunately, searching for the best classification model out of all the possible candidates can be defined as an optimization problem appropriate for solving with stochastic nature-inspired population-based algorithms, where the quality

* Corresponding Author: `iztok.fister1@um.si`

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50426-7_2

of solutions is evaluated according to classification accuracy and comprehensibility. The majority of these algorithms represented the classification model in terms of "If-then" rules, and are, therefore, close to human comprehension.

Stochastic nature-inspired population-based algorithms have frequently been applied to data mining in the last three decade. For instance, Srikanth et al. in [18] proposed the Genetic Algorithm (GA) for clustering and classification. In the Swarm Intelligence (SI) domain, Particle Swarm Optimization (PSO) and Ant Colony Optimization (ACO) attracted the most scientists to use them for solving the problems in data mining. For instance, Sousa et al. [17] compared the implemented PSO algorithms for data mining with the GA, while Ant-Miner, developed by Parpinelly et al. [11] using ACO, was proposed for discovering classification rules. The more complete surveys of using EAs in data mining can be found in [4, 5, 15], while the review of papers describing SI-based algorithms in data mining was presented in [10]. Recently, a Gravitational Search (GS) has achieved excellent results in discovering classification models, as reported by Peng et al. [13].

This paper tries to answer the question if the stochastic nature-inspired population-based algorithms can be competitive tools for pure binary classification compared with the classical data mining methods. Here, the pure means that the algorithms perform classification by standalone, i.e., without any interaction with the traditional methods. In line with this, the Firefly Algorithm (FA) [20] for binary classification was proposed, that is capable of discovering the classification models and evaluating their quality according to a classification accuracy. In our opinion, the main advantage of the FA against the PSO algorithm lays in the principle of FA working, because particles in this algorithm are not dependent only on the global best solution as in PSO, but also on the more attractive particles in the neighborhood. On the other hand, the model in the proposed FA consists of two classification rules, i.e., one for True Negative (TN) and the other for True Positive (TP) classification results. Moreover, each classification rule is represented by omitting some features. This means that the feature selection task is included into the classification implicitly.

The proposed FA was applied to three well-known datasets for binary classification that are publicly available on the web. The obtained results showed its big potential in binary classification that could also be applied for general classification.

The main goals of this paper are as follows:

- developing the new classification method based on real-coded FA,
- encoding two classification rules simultaneously, and decoding by the new genotype-phenotype mapping,
- performing the feature selection implicitly by the classification,
- evaluating the proposed method on some datasets for binary classification.

In the remainder, the structure of the paper is as follows: Section 2 introduces fundamentals of the FA. In Section 3, the proposed classification method is described in detail. The experiments and results are presented in Section 4, while the paper is concluded with Section 5, in which directions for the future work are also outlined.

2 Fundamentals of the Firefly Algorithm

The inspiration for the Firefly Algorithm (FA) was fireflies with flashing lights that can be admired on clear summer nights. The light is a result of complex chemical reactions proceeding in a firefly‘s body and has two main purposes for the survival of these small lightning bugs: (1) To attract mating partners, and (2) To protect against predators. As follows from a theory of physics, the intensity of the firefly‘s light decreases with increasing the distance r from the light source, on the one hand, and the light is absorbed by the air as the distance from the source increases, on the other.

Both physical laws of nature are modeled in the FA developed by Yang at 2010 [20], as follows: The FA belongs to a class of Swarm Intelligence (SI) based algorithms, and therefore operates with a population of particles representing solutions of the problem in question. Thus, each solution is represented as a real-valued vector, in other words:

$$\mathbf{x}_i^{(t)} = \{x_{i,1}^{(t)}, \ldots, x_{i,D}^{(t)}\}, \quad \text{for } i = 1, \ldots, N, \tag{1}$$

where N denotes the population size, D a dimension of the problem to be solved, and t is a generation number. Here, the elements are initialized according to the following equation:

$$x_{i.j}^{(0)} = U(0,1) \cdot (Ub_j - Lb_j) + Lb_j, \quad \text{for } i = 1, \ldots, N \wedge j = 1, \ldots, D, \tag{2}$$

where $U(0,1)$ denotes the random number drawn from uniform distribution in interval $[0,1]$, and Ub_j and Lb_j are the upper and lower bounds of the j-th element of the vector.

The physical laws of a firefly flashing are considered in the FA by introducing the light intensity relation, as follows:

$$I(r) = I_0 \cdot \exp^{-\gamma r^2}, \tag{3}$$

where I_0 denotes the light intensity at the source, and γ is a light absorption coefficient. Similar to the light intensity, the attraction between two fireflies, where the brighter is more capable of attracting a potential mating partner, is calculated according to the following equation:

$$\beta(r) = \beta_0 \cdot \exp^{-\gamma r^2}, \tag{4}$$

where β_0 is the attractiion at $r = 0$.

The distance $r_{i,j}^{(t)}$ between two fireflies $\mathbf{x}_i^{(t)}$ and $\mathbf{x}_j^{(t)}$ is expressed as a Euclidian distance, as follows:

$$r_{i,j}^{(t)} = \|\mathbf{x}_i^{(t)} - \mathbf{x}_j^{(t)}\| = \sqrt{\sum_{k=1}^{D} x_{i,k}^{(t)} - x_{j,k}^{(t)}}. \tag{5}$$

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50426-7_2

The variation operators are implemented as a move of a definite virtual firefly i towards the more attractive firefly j according to the following equation:

$$\mathbf{x}_i^{(t+1)} = \mathbf{x}_i^{(t)} + \beta_0 \cdot \exp^{-\gamma {r_{i,j}^{(t)}}^2} \left(\mathbf{x}_j^{(t)} - \mathbf{x}_i^{(t)}\right) + \alpha \cdot \epsilon_i, \tag{6}$$

which consists of three terms: The current position $\mathbf{x}_i^{(t)}$ of the i-th firefly, the social component determining the move of the i-th firefly towards the more attractive j-th firefly, and a randomization component determining the random move of the same firefly in the search space.

Typically, the step size scaling factor α is proportional to the characteristics of the problem, while the randomization factor ϵ_i is a random number drawn from Gaussian distribution with mean zero and Standard Deviation one, denoted as $N(0, 1)$. Contrarily, the uniform distribution $U(0, 1)$ from interval $[0, 1]$ was used in our study instead of the normal distribution.

The quality of the solution, expressed by the fitness function, is, in the FA, proportional to the light intensity as $I(\mathbf{x}_i) \propto f(\mathbf{x}_i)$. The pseudo-code of the FA is illustrated in the Algorithm 1, from which it can be seen that this consists of the following components: (1) Representation of a solution, (2) Initialization (line 1), (3) Termination condition (line 4), (4) Move operator (line 8), (5) Evaluation function (lines 2 and 11), and (6) Ranking and finding the best solution.

Algorithm 1 Pseudo code of the basic Firefly algorithm

Input: Population of fireflies $\mathbf{x} = (\mathbf{x}_1, \ldots, \mathbf{x}_N)$, objective function $f(\mathbf{x}_i)$.
Output: The best solution $\mathbf{x}_{best}$ and its value $f_{min} = \min(f(\mathbf{x}_{best}))$.

```
 1: generate_initial_population $\mathbf{x}^{(0)} = (\mathbf{x}_1^{(0)}, \ldots, \mathbf{x}_N^{(0)})$;
 2: $f(\mathbf{x}_i^{(0)})$ = evaluate_new_solution_and_update_light_intensity;
 3: $t = 0$;
 4: while $t < MAX\_GEN$ do
 5:     for $i = 1$ to $N$ do
 6:         for $j = 1$ to $N$ do
 7:             if $I_j > I_j$ then
 8:                 move_firefly_$i$_towards_$j$_using_Gaussian_distribution;
 9:             end if
10:         end for
11:         $f(\mathbf{x}_i^{(t)})$ = evaluate_new_solution_and_update_light_intensity;
12:     end for
13:     rank_fireflies_and_find_the_best;
14:     $t = t + 1$
15: end while
```

3 Proposed method

The task of the proposed stochastic nature-inspired population-based algorithm is to search for the model appropriate for binary classification of an arbitrary dataset. The model consists of two rules containing features of the dataset for predicting the True Negative and True Positive values. Thus, the learning is divided into a training phase, in which 80 % of dataset instances are included, and a test phase, where we operate with the remaining 20 % of the instances in the same dataset. The search for a model is defined as an optimization, where the fitness function is defined as a classification accuracy metric that is expressed mathematically as:

$$Acc = \frac{TN + TP}{TN + FN + TP + FP}, \tag{7}$$

where TN= True Negative, TP= True Positive, FN= False Negative, and FP= FalsePositive . Indeed, the accuracy measures the performance of a statistical measure.

The solutions $\mathbf{x}_i$ for $i = 1, \ldots, N$ in the proposed algorithm are represented as real-valued vectors with elements $x_{i,j} \in [0, 1]$ for $j = 1, \ldots, L$, representing features, where L is the length of a solution, to which the binary vector $\mathbf{b}_i = \{b_{i,j}\}$ is attached with elements $b_{i,k} \in \{0, 1\}$ for $k = 1, \ldots, M$, and M is the number of features. These are obtained in the preprocessing phase, where the dataset in question is analyzed in detail. The features can be either categorical (i.e., $b_{i,k} = 0$) or numerical (i.e., $b_{i,k} = 1$). The former consists of attributes drawn from a discrete set of feasible values, while the latter of continuous intervals limited by their lower and upper bounds. Each feature has its own predecessor *control*, determining its presence or absence in the specific rule.

In summary, the length of the solution L is calculated as:

$$L = 2 \cdot num\_of\_category\_attr + 3 \cdot num\_of\_numeric\_attr + 1, \tag{8}$$

where *num_of_category_attr* denotes the number of categorical features, *num_of_numeric_attr* is the number of numerical features, and one is reserved for *threshold* that determines if the definite feature belongs to the rule or not. Obviously, the feature belongs to the rule when the relation *control* ≥ *threshold* is satisfied.

In order to transform the representation of solutions into their problem context, the genotype-phenotype mapping is needed. The genotype-phenotype mapping determines how the genotype $\mathbf{x}_i$ of length L, calculated according to Eq. (8), is mapped into the corresponding phenotype $\mathbf{y}_i$ for $k = 1, \ldots, M$, where the variable M denotes the number of features in a dataset.

There are two ways in which to perform the genotype-phenotype mapping, depending on the type of feature: Actually, the categorical variables demand two, and the numerical even three elements for this mapping. In general, the mapping is expressed mathematically as (Fig. 1):

$$y_{i,k} = \begin{cases} -1, & \text{if } x_{i,0}^{(k)} < x_{i,L}, \\ \left\lfloor |Attr_k| \cdot x_{i,1}^{(k)} \right\rfloor, & \text{if } b_{i,k} = 0, \\ \left[\left\lfloor |D_k| \cdot x_{i,1}^{(k)} \right\rfloor, \left\lfloor |D_k| \cdot x_{i,2}^{(k)} \right\rfloor \right], & \text{if } b_{i,k} = 1, \end{cases} \tag{9}$$

for $k = 1, \ldots, M$, where $|Attr_k|$ denotes the size of the k-th attribute set $Attr_k = \{a_1, \ldots, a_{n_k}\}$, and D_k is a domain of feasible values of attributes, expressed as

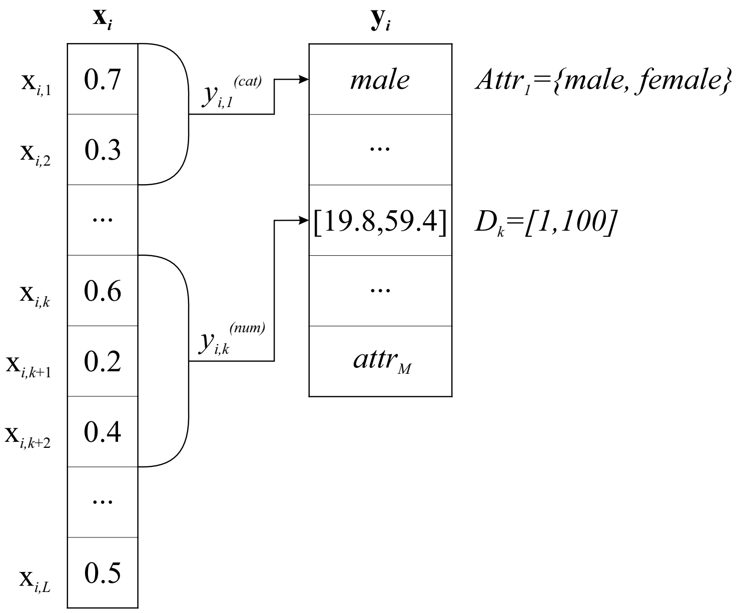

Fig. 1: Genotype-phenotype mapping.

$(Max_k - Min_k)$, where Max_k and Min_k represent the maximum and minimum values of the numerical feature found in the dataset.

In summary, the phenotype value $y_{i,k}$ can obtain three values after the genotype-phenotype mapping: (1) -1, If the feature is not present in the rule, (2) The attribute of the feature set, if the feature is categorical, and (3) The interval of the feasible values, if the feature is numerical.

4 Experiments and results

The aim of conducting experiments was twofold: (1) To evaluate the performance of the proposed method on some well-known binary datasets, and (2) To compare the obtained results with the results of some classical classification methods. In line with this, the results of the FA for binary classification were compared with the results obtained by: (1) Random Forest (RF) [3], (2) Multi-Layer Perceptron (MLP) [21], and (3) Bagging [2]. All algorithms in the experiments were applied to three well-known datasets for binary classification taken from the UCI Machine Learning Repository [1], whose characteristics are depicted in Table 1.

As can be seen from the Table, the binary datasets in question are not extremely big, because the number of features are less than 10, while the number of instances does not exceed the value of 1,000.

The parameter setting of the FA during the tests is presented in Table 2, from which it can be seen that the maximum number of fitness function evaluations

Table 1: Datasets used in our experiments.

Dataset name	No. of features	No. of instances
Pima Indians Diabetes data set	8	768
Haberman survival dataset	3	306
Breast cancer	9	683

Table 2: Parameter settings of firefly algorithm.

Parameter	Abreviation	Value
Maximum number of generations	$nFES$	5,000
Population size	N	90
Step size scaling factor	α	0.5
Attractiveness at $r = 0$	β_0	0.2
Light absorption factor	γ	1.0

amount to $5,000$. Settings of the other algorithm parameters were taken from the existing literature. This means that no special optimization of parameter settings was performed in the study. Moreover, the proposed FA did not include any domain-specific knowledge about imposed classification problems incorporated in the sense of adaptation or hybridization.

Indeed, the proposed FA for binary classification was implemented in the Python programming language using the external NiaPy library [19]. The implementations of the remaining three methods were taken from the scikit-learn Python package [12], where default parameter settings were adopted. Let us emphasize that 25 independent runs were conducted for each method in question, where the achieved classification accuracy was collected after each run. As a result, the quality of the methods was evaluated according to the five aforementioned standard statistical measures: Minimum, maximum, average, median, and Standard Deviation values.

The detailed results of the comparative analysis according to classification accuracy are illustrated in Table 3, where five statistical measures are analyzed according to the used algorithms and the observed datasets. Thus, the best results are presented in bold case. As can be seen in the Table 3, the best results were achieved by the RF and MLP classification methods, where the RF gained the better accuracy by classifying the Pima dataset, the MLP was better at the Haberman dataset, while, at the Breast dataset, both mentioned methods obtained the same classification accuracy.

According to Table 4, where the percent of deviation of the results of the definite method from the best results designated by '‡' in the Table, are calculated, the RF and MLP classification methods exhibit the best percent in general, because they outperformed all the others even three times. The bagging achieved the best mean results by classification of the Pima dataset. Although the FA for binary classification did not achieve the best accuracy in any instance of dataset, its best, as well as mean results, were no worse than 10 % of the best results,

Table 3: Detailed results of the binary classification according to accuracy.

Dataset	Algorithm	*Min*	*Max*	*Mean*	*Median*	*Std*
Pima	FA	0.5844	0.7662	0.6849	0.6818	0.0519
	RF	0.6688	**0.7987**	0.7387	0.7402	0.0342
	MLP	0.5779	0.7467	0.6800	0.6818	0.0437
	Bagging	0.6948	0.7922	**0.7407**	0.7402	0.0273
Haberman	FA	0.6451	0.7903	0.7264	0.7419	0.0368
	RF	0.5645	0.8064	0.7070	0.6935	0.0565
	MLP	0.6612	**0.8709**	**0.7658**	0.7741	0.0587
	Bagging	0.5645	0.7903	0.6741	0.6774	0.0558
Breast	FA	0.8029	0.8978	0.8616	0.8686	0.0293
	RF	0.9270	**0.9854**	**0.9623**	0.9562	0.0162
	MLP	0.9343	**0.9854**	0.9620	0.9635	0.0133
	Bagging	0.9270	0.9781	0.9570	0.9562	0.0146

Table 4: Summary results of the binary classification according to accuracy.

Dataset	Measure	FA	RF	MLP	Bagging
Pima	*Max* [%]	95.93	‡	93.49	99.19
	Mean [%]	92.47	99.73	91.81	‡
Haberman	*Max* [%]	90.75	92.59	‡	90.75
	Mean [%]	94.86	92.32	‡	88.03
Breast	*Max* [%]	91.11	‡	‡	99.26
	Mean [%]	89.49	‡	99.92	99.40

except by classification of the Breast dataset, where the accuracy was close to this border value (precisely 89.49 %).

Finally, an example of classification rules generated by the proposed FA in classifying the Pima Diabetes dataset is illustrated in Table 5 that is divided into two parts: (1) Feature, and (2) Classification rules. The former consists of three fields: Sequence number, feature name and type. The latter is divided into two rules, i.e., for True Negative TN, and for True Positive TP classifications.

As can be seen from the Table, there are eight features in the dataset. Interestingly, the dataset supports only numerical attributes. These attributes are, therefore, represented as continuous domains of values. Thus, the first rule determines the combination attributes that are classified as True Negative predictions, while the second rule as True Positive predictions.

From this Table, it can be concluded that this representation of rules is really comprehensive, and, in that way, is easily understandable by the user.

Table 5: An example of classification rule in classifying Pima Diabetes dataset generated by the proposed FA for binary classification.

Feature			Classification rules	
Num.	Name	Class	TN	TP
1	Number of times pregnant	Numeric	[0.79,16.04]	[13.69,16.28]
2	Plasma glucose concentration	Numeric	[25.92,148.08]	n/a
3	Diastolic blood pressure	Numeric	[6.18,84.45]	[53.71,81.74]
4	Triceps skin fold thickness	Numeric	[8.33,52.15]	[15.39,27.88]
5	2-hour serum insulin	Numeric	[435.02,730.53]	[759.30,840.51]
6	Body mass index	Numeric	[36.43,37.96]	[31.75,58.41]
7	Diabetes pedigree function	Numeric	n/a	n/a
8	Age	Numeric	[68.45,75.98]	34.29,41.01]

5 Discussion

The huge progress in big data has caused the rapid development of new data mining methods that need to satisfy two requests: (1) To process enormous volumes of data, and (2) To ensure enough processing time for their analysis. The classical data mining methods suffer from a lack of comprehensibility that disallows users to use them as effectively as possible. Mainly, the stochastic nature-inspired population-based algorithms are well-known general tools suitable for solving the hardest optimization problems. Recently, this family of algorithms has also been applied for solving the problems from data mining, where they can search for the best model in the model search space.

In this preliminary study, the FA was proposed for the binary classification task, with the following advantages: The FA search process searches for new solutions, not only on basis of the best global solution, but moves each particle in the search space with regard to its neighborhood consisting of the more attractive particles. Furthermore, the original FA operates with real-valued vectors, which represent the solutions of the problem in question. The genotype-phenotype mapping must be performed in order to transform the representation in the genotype space into the solution in the problem context. In our case, the mapping decodes two classification rules from each solution, where the first is dedicated for classification of TN predictions, while the second for classification of TP. Moreover, the algorithm is capable of performing the feature selection implicitly, because only the more important features must be presented in the solution. Finally, the features can be either categorical or numerical. Both types are represented as real values and, therefore, no discretization is necessary.

The proposed FA for binary classification was applied to three well-known datasets publicly available on the web. The obtained results were compared with three classical classification methods: RF, MLP, and boosting. Although the FA did not improve the results achieved by the classical methods, they showed that this has a big potential for improving its results in the future, especially due to the fact that the algorithm was used as is, i.e., no features were implemented to improve it.

In line with this, the improvement of the FA in the sense of adaptation and hybridization should be a reasonable direction for the future. However, testing the behavior of the algorithm on general classification problems could also be challenging.

Acknowledgment

Iztok Fister Jr., Grega Vrbančič and Vili Podgorelec acknowledge the financial support from the Slovenian Research Agency (Research Core Funding No. P2-0057). Dušan Fister acknowledges the financial support from the Slovenian Research Agency (Research Core Funding No. P5-0027). Iztok Fister acknowledges the financial support from the Slovenian Research Agency (Research Core Funding No. P2-0041).

References

1. Arthur Asuncion and David Newman. Uci machine learning repository, 2007.
2. Leo Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996.
3. Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.
4. Erick Cantú-Paz. On the use of evolutionary algorithms in data mining. In *Data Mining: A Heuristic Approach*, pages 22–46. IGI Global, 2002.
5. Alex A Freitas. *A Survey of Evolutionary Algorithms for Data Mining and Knowledge Discovery*, pages 819–845. Springer Berlin Heidelberg, Berlin, Heidelberg, 2003.
6. Nir Friedman, Dan Geiger, and Moises Goldszmidt. Bayesian network classifiers. *Mach. Learn.*, 29(2-3):131–163, November 1997.
7. Simon Haykin. *Neural Networks: A Comprehensive Foundation.* Prentice Hall PTR, Upper Saddle River, NJ, USA, 2nd edition, 1998.
8. Marti A. Hearst. Support vector machines. *IEEE Intelligent Systems*, 13(4):18–28, July 1998.
9. Oded Maimon. *Introduction to Soft Computing for Knowledge Discovery and Data Mining*, pages 1–13. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.
10. David Martens, Bart Baesens, and Tom Fawcett. Editorial survey: swarm intelligence for data mining. *Machine Learning*, 82(1):1–42, jan 2011.
11. R. S. Parpinelli, H. S. Lopes, and A. A. Freitas. Data mining with an ant colony optimization algorithm. *Trans. Evol. Comp*, 6(4):321–332, August 2002.

12. Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12(Oct):2825–2830, 2011.
13. Lizhi Peng, Bo Yang, Yuehui Chen, and Ajith Abraham. Data gravitation based classification. *Inf. Sci.*, 179(6):809–819, March 2009.
14. J. Ross Quinlan. Induction of decision trees. *Machine learning*, 1(1):81–106, 1986.
15. Sashi Rekha. A survey of evolutionary algorithms and its use in data mining application. *Journal of Pure and Applied Mathematics*, 119(12):13593–13600, 2018.
16. Saroj, Jyoti Vashishtha, Pooja Goyal, and Jyoti Ahuja. A Novel Fitness Computation Framework for Nature Inspired Classification Algorithms. *Procedia Computer Science*, 132:208–217, 2018.
17. Tiago Sousa, Arlindo Silva, and Ana Neves. Particle swarm based data mining algorithms for classification tasks. *Parallel Comput.*, 30(5-6):767–783, May 2004.
18. Radhakrishnan Srikanth, Roy George, N Warsi, Dev Prabhu, Frederick E Petry, and Bill P Buckles. A variable-length genetic algorithm for clustering and classification. *Pattern Recognition Letters*, 16(8):789–800, 1995.
19. Grega Vrbančič, Lucija Brezočnik, Uroš Mlakar, Dušan Fister, and Iztok Fister Jr. NiaPy: Python microframework for building nature-inspired algorithms. *Journal of Open Source Software*, 3, 2018.
20. Xin-She Yang. Firefly algorithm, stochastic test functions and design optimisation. *International Journal of Bio-Inspired Computation*, 2(2):78–84, 2010.
21. Jacek M Zurada. *Introduction to artificial neural systems*, volume 8. West publishing company St. Paul, 1992.

ICCS Camera Ready Version 2020
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-50426-7_2