A novel binary classification approach based on geometric semantic genetic

I. Bakurov[a], M. Castelli[a], A. Scotto di Freca[b], L. Vanneschi[a], F. Fontanella[b]

[a] *NOVA Information Management School (NOVA IMS), Universidade Nova de Lisboa, Campus de Campolide, Lisboa 1070-312, Portugal*

[b] *Dipartimento di Ingegneria Elettrica e dell'Informazione, Universitá di Cassino e del Lazio meridionale, Via G. Di Biasio 43, 03043 Cassino (FR) { Italy*

This is the accepted author *manuscript of the following article published by Elsevier:* Bakurov, I., Castelli, M., Fontanella, F., Scotto Di Freca, A., & Vanneschi, L. (2022). A novel binary classification approach based on geometric semantic genetic programming. Swarm and Evolutionary Computation, 69(March), 1-12. [101028]. https://doi.org/10.1016/j.swevo.2021.101028

Funding:

This work was supported by national funds through the FCT (Fundação para a Ciência e a Tecnologia) by the projects GADgET (DSAIPA/DS/0022/2018), BINDER (PTDC/CCIINF/29168/2017), and AICE (DSAIPA/DS/0113/2019). Mauro Castelli acknowledges the financial support from the Slovenian Research Agency (research core funding no. P5-0410).

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License**.**

A novel binary classification approach based on geometric semantic genetic programming

I. Bakurov[a], M. Castelli[a], A. Scotto di Freca[b], L. Vanneschi[a], F. Fontanella[b,*]

[a] *NOVA Information Management School (NOVA IMS)*
Universidade Nova de Lisboa, Campus de Campolide, Lisboa 1070-312 – Portugal
[b] *Dipartimento di Ingegneria Elettrica e dell'Informazione*
Università di Cassino e del Lazio meridionale
Via G. Di Biasio 43, 03043 Cassino (FR) – Italy

Abstract

Geometric semantic genetic programming (GSGP) is a recent variant of genetic programming. GSGP allows the landscape of any supervised regression problem to be transformed into a unimodal error surface, thus it has been applied only to this kind of problem. In a previous paper, we presented a novel variant of GSGP for binary classification problems that, taking inspiration from perceptron neural networks, uses a logistic-based activation function to constrain the output value of a GSGP tree in the interval $[0, 1]$. This simple approach allowed us to use the standard RMSE function to evaluate the train classification error on binary classification problems and, consequently, to preserve the intrinsic properties of the geometric semantic operators. The results encouraged us to investigate this approach further. To this aim, in this paper, we present the results from 18 test problems, which we compared

*Corresponding author
Email addresses: ibakurov@isegi.unl.pt (I. Bakurov), mcastelli@novaims.unl.pt (M. Castelli), a.scotto@unicas.it (A. Scotto di Freca), vanneschi@novaims.unl.pt (L. Vanneschi), fontanella@unicas.it (F. Fontanella)

with those achieved by eleven well-known and widely classification schemes. We also studied how the parameter settings affect the classification performance and the use of the F-score function to deal with imbalanced data. The results confirmed the effectiveness of the proposed approach.
Keywords: Binary classification, Geometric Semantic Genetic Programming

1. Introduction

Improving the effectiveness of genetic programming (GP) [1] for tackling classification tasks has been a steady research trend, at least in the last two decades. A relatively complete survey [2] reports numerous contributions published before 2010, proposing the use of GP for evolving classification algorithms, classification rules, and classifier expressions by incorporating several different techniques to represent solutions and ways of assessing fitness, including single- and multi-objective approaches. The limitations of the various proposed methods, discussed in [3], paved the way for the further research that took place actively in the last decade. A recent survey can be found in [4]. Among the existing approaches, particularly successful were some attempts of integrating GP with other classifier systems, such as k-Nearest Neighbors, with and without data discretization [5]; with clustering, often applied on feature spaces remapped by GP expressions [6, 7], and with ensemble techniques [8, 9, 10]. All in all, the recently proposed GP methods all share some sophistication and algorithmic complexity, which testifies to a certain difficulty of GP in tackling classification tasks, clearly recognized, for instance, in [6]. This highly contrasts with the recent successes that were ob-

tained by GP on regression problems [11]. In particular, geometric semantic genetic programming (GSGP) [12, 13], a recent development of GP, stood out for numerous successful real-life applications [14]. Despite its many successful applications in regression problems, researchers involved with GSGP still have not found a way to adapt it successfully to classification problems.

In a previous paper, we proposed an extension of GSGP for tackling binary classification tasks [15]. Contrary to some of the most recently proposed approaches, the idea was extremely simple, which made it much easier to understand and implement compared to many others. The proposed approach was inspired by the functioning of the perceptron artificial neural network [16]: the model output of GSGP (which is a floating-point number, traditionally used for regression) is given as input to an activation function, that limits its values in a given interval, making the interpretation of the output as a class label easier. The solid theory behind perceptron, as well as its numerous successes in real-world applications, strengthened by recent developments [17], encouraged us to pursue this promising idea. We tested the proposed approach on 10 data sets and compared its results with those achieved by a support vector machine (SVM). The results confirmed the effectiveness of the GSGP extension.

In this paper, we propose an extension of that work. In particular, we test our approach on 18 data sets and compare the results with those achieved by seven classification schemes. We also introduce the use of the F-score function to measure methods' performance on problems with imbalanced data.

The work is organized as follows. In Section 3, we briefly introduce GSGP. In Section 4, we explain how GSGP is extended to tackle binary classification

tasks; analogously to perceptron, the method introduces a new parameter, whose effect on classification performance is thoroughly discussed. Section 5 presents our experimental study, first introducing the 18 chosen test problems (publicly available binary classification data sets, characterized by different feature space dimensions and degrees of imbalance), then discussing the used parameter settings, and lastly presenting and discussing the obtained results. The discussion of the experimental results is subdivided into a part dedicated to the tuning of the new parameter and a part in which the proposed system is compared to state-of-the-art machine learning methods, such as Support Vector Machines, Decision Trees, Artificial Neural Networks, Bayesian Classifiers, and Logistic Regression. Finally, Section 6 concludes the paper and proposes ideas for future research.

2. Related work

As already mentioned in the Introduction, many attempts have been made to solve the limitations of standard GP-based approaches for classification problems, including the integration with other classification systems or the use of ensemble techniques.

In [18], Z-Flores et al. proposed a GP-based system for binary classification. The work combined GP with a local search strategy to improve the performance of a standard GP classifier. The work of Z-Flores et al. is inherently different with respect to the one we propose in this paper. In particular, Z-Flores et al. relied on standard syntax-based genetic operators for genetic programming, and they did not consider semantic awareness in the evolutionary process. Moreover, in their method, they weighted each

node of a GP tree with a real-value parameter, and they employed a non-linear Gauss-Newton method (the Trust Region algorithm [19]) as a local search method. On the other hand, the method proposed in this paper exploits semantic awareness to guide the search process, does not assign any parameter to the node of the tree, and finally, employs the traditional GP evolutionary process, with no local search method. For these reasons, the work of Z-Flores et al., although dealing with binary classification, belongs to the family of memetic GP, and it is conceptually different with respect to the approach we are proposing. The only common intuition we shared with the work of Z-Flores et al. is the use of the sigmoid function for constraining the output of the GP tree in [0;1]. Nonetheless, also in this case, there is an important difference: Z-Flores et al. used the sigmoid to provide gradient information to the local search method. However, in this work, we use the sigmoid function to obtain continuous values in [0;1] for using the RMSE as a fitness function. The values can subsequently be transformed into either 0 or 1 by applying a threshold, similarly to what happens in the traditional classification approach with GP.

Another noteworthy contribution is the one based on the multi-dimensional multi-class genetic programming (M2GP) approach [20]. The main idea behind M2GP is to exploit the search ability of GP to find a suitable transformation that allows the original data to be grouped into unique clusters, one for each class. Once the training samples have been transformed according to the mapping encoded by an individual, class centroids are computed, and each sample is labeled with the class of its closest centroid. Then the percentage of samples correctly classified (i.e., the accuracy) is used as the

fitness value for that individual. Unfortunately, the search ability of M2GP is limited by the fact that the dimensionality of the transformed space must be set a priori because the operators implemented cannot change the number of dimensions of the transformation encoded by a given individual. To tackle this drawback, in [6] the authors presented an enhanced version of the algorithm, named M3GP. They implemented the ability to evolve individuals with a variable number of dimensions by using operators that can add or remove the dimensions of the individuals to be modified. In [7] the authors presented a further enhancement of the M2GP, based on a novel encoding program that simplifies the construction of multidimensional representations. This encoding allowed for achieving better performance thanks to the implementation of an advanced parent selection technique that led to more accurate classifiers. Finally, the extension presented in [21] uses a stack-based program representation, which permits a further simplification of the construction of multidimensional solutions. This extension also incorporates a multiobjective parent selection and survival technique, as well as an archiving strategy that maintains a set of optimal solutions, taking into account their complexity and accuracy. Moreover, the selection of the final model from this archive allowed a reduction of overfitting. The combination of these techniques allowed this further extension to outperform the previous MGP versions on a wide set of test problems. More recently, La Cava and Moore [22] enriched the MGP approach by adding two semantic crossover operators to choose where useful building blocks are placed during crossover. The experimental results, from a large set of benchmark regression problems, confirmed the effectiveness of the proposed operators.

The main difference between M2GP, M3GP (and the subsequent improvements), and the approach proposed here stands in the fact that the approach we propose is much simpler. It is, in fact, able to work directly on the original feature space, without any remapping required. Furthermore, M2GP, M3GP, and their descendants force the user to make choices that may not be straightforward and that generally have an important impact on the system's performance. The first of these choices, and arguably the most crucial one, concerns the metric to quantify distances in the mapped feature space. Significant performance differences are reported in the literature according to the different metrics employed [20, 6]. On the other hand, the algorithm we propose does not calculate distance, and so this choice can be avoided. Secondly, even though M3GP and the following improvements relieve the user from choosing the dimension of the mapped space, which was fixed in M2GP, an upper bound to this dimension still needs to be chosen, and the initialization algorithm uses this information. On the other hand, the user of the proposed approach is completely free from setting this parameter because no mapping is required. Last but not least, M2GP, M3GP, and successors have been specifically designed to tackle multi-class classification problems and spend a significant computational effort to deal with multi-class labels. However, binary classification is the target of this work. This fact implies that the MGP-based approaches and our method employ different orders of magnitude of the computational effort (our approach being much cheaper from this viewpoint). This makes the two approaches hardly comparable. For this reason, we do not include any experimental comparison with any of the MGP-based approaches here.

GSGP has proven its effectiveness in many real-life applications, which spans from the prediction of concrete strength [23] to the support of medical decisions for treating rare diseases [24]. In particular, in [23], the authors use GSGP to predict a value that measures the strength of high-performance concrete, given some parameters representing the materials (cement, water, etc.) used to prepare the mixture to be evaluated. Whereas, in [24], medical decisions are supported by using GSGP to develop predictive models to forecast the effect of a specialized therapy; this forecast is performed by predicting the value of patients' motor functioning, taking as input six multidimensional factors, and is integrated in a user-friendly web application. GSGP has also been used to solve complex problems in the field of pharmacokinetics. In [25], it was used to address two well-known problems in the field, namely the prediction of human oral bioavailability and protein-plasma binding levels of medical drugs. Also in this case, the problems faced required the prediction of a real-valued index. As regards medical applications, GSGP has also been applied for the prediction of the unified Parkinson's disease rating scale (UPDRS) index [26], a score that provides an efficient and flexible way of measuring and monitoring PD-related disability and impairment. Here the UPDRS score was predicted using 18 features. More recently, the GSGP has been enhanced by using local search operators [27, 28]. In [27], the authors applied the enhanced GSGP approach to two problems in the biomedical field: computerized tomography (CT) scan and 3D Protein Structure. In the first problem, given a CT image represented by features describing bone structures and air inclusions, GSGP was used to predict the target variable that is the relative location of the image on the axial axis. In the second

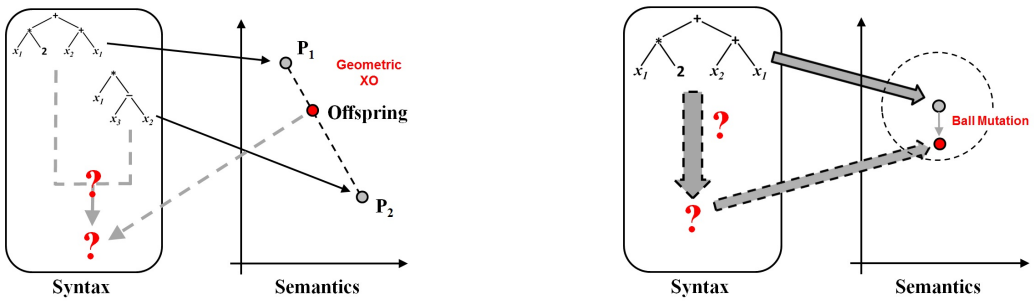
problem, GSGP was used to predict a target variable related to the size of the residues, taking as input a protein's tertiary structure data. On the other hand, in [28] GSGP, has been used to predict the performance of a complex organizational structure, which can be a regional innovation system. The proposed approach takes input indicators related to the regional knowledge base and competitiveness indexes and provides four indicators as output.

3. Geometric semantic genetic programming

Let $\mathbf{X} = \{\overrightarrow{x_1}, \overrightarrow{x_2}, ..., \overrightarrow{x_n}\}$ be the set of input data (training instances, observations or fitness cases) of a symbolic regression problem, and $\overrightarrow{t} = [t_1, t_2, ..., t_n]$ the vector of the respective expected output or target values (in other words, for each $i = 1, 2, ..., n$, t_i is the expected output corresponding to input $\overrightarrow{x_i}$). A GP individual (or program) P can be seen as a function that, for each input vector $\overrightarrow{x_i}$ returns the scalar value $P(\overrightarrow{x_i})$. Following [12], we call *semantics* of P the vector $\overrightarrow{s_P} = [P(\overrightarrow{x_1}), P(\overrightarrow{x_2}), ..., P(\overrightarrow{x_n})]$. This vector can be represented as a point in an n-dimensional space, which we call *semantic space*. Note that the target vector $\overrightarrow{t}$ is a point in the semantic space.

As explained above, GSGP is a variant of GP, in which the traditional crossover and mutation are replaced by new operators called geometric semantic operators (GSOs). The objective of GSOs is to define modifications on the syntax of GP individuals that have a precise and known effect on their semantics. In particular, as schematically shown in Figure 1, GSOs are as follows:

- Geometric semantic crossover. This operator generates only one offspring, whose semantics stand in line to join the semantics of the two



(a) (b)

Figure 1: Geometric semantic crossover (plot [a]); respectively geometric semantic mutation (plot [b]) performs a transformation on the syntax of the individual that corresponds to geometric crossover (respectively geometric mutation) on the semantic space. In this figure, the unrealistic case of a bidimensional semantic space is considered, for simplicity.

parents in the semantic space.

- Geometric semantic mutation. With this operator, by mutating an individual i, we obtain another individual j, such that the semantics of j stands inside a ball of a given predetermined radius, which is centered in the semantics of i.

One of the reasons why GSOs became so popular in the GP community is related to the fact that GSOs induce a unimodal error surface (on the training data) for any supervised learning problem, for which fitness is calculated using an error measure between outputs and targets. In practice, the use of GSOs guarantees that the error surface on the training data does not have a locally optimal solution but rather a single global optimum. This property holds, for instance, for any regression or classification problem, independently of how big and how complex data are (see reference [13] for a

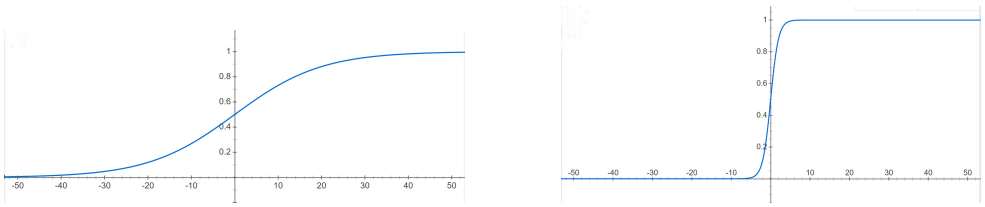
detailed explanation). The definitions of the GSOs follow, as given in [12], respectively.

Geometric Semantic Crossover (GSC). Given two parent functions $T_1, T_2 : \mathbb{R}^n \to \mathbb{R}$, the geometric semantic crossover returns the real function $T_{XO} = (T_1 \cdot T_R) + ((1 - T_R) \cdot T_2)$, where T_R is a random real function whose output values range in the interval $[0, 1]$.

Geometric Semantic Mutation (GSM). Given a parent function $T : \mathbb{R}^n \to \mathbb{R}$, the geometric semantic mutation with mutation step ms returns the real function $T_M = T + ms \cdot (T_{R1} - T_{R2})$, where T_{R1} and T_{R2} are random real functions.

Even though this is not in the original definition of GSM, later contributions [13] have clearly shown that limiting the codomain of T_{R1} and T_{R2} in a predefined interval (for instance $[0, 1]$, as done for T_R in GSC) helps to improve the generalization ability of GSGP. As in several previous works [13, 29], here we constrain the outputs of T_R, T_{R1} and T_{R2} by wrapping them in a logistic function.

Only the definitions of the GSOs for symbolic regression problems are given here because they are the only ones used in this work. For the definition of GSOs for other domains, the reader is referred to [12]. Figure 1 shows a graphical representation of the mapping between the syntactic and semantic space given by the GSOs. Using these operators, the semantics of the offspring is completely defined by the semantics of the parents: the semantics of an offspring produced by GSC will lie on the segment between the semantics of both parents (geometric crossover), but GSM defines a mutation such that the semantics of the offspring lies within the ball of radius



(a) $\alpha = 0.1$ (b) $\alpha = 1.0$

Figure 2: Graphical representations of the logistic function (Equation [1]) with two different values of the α parameter.

ms that surrounds the semantics of the parent (geometric mutation).

As reported in [12, 13], the GSO's property of inducing a unimodal error surface has a price. The price, in this case, is that GSOs always generate larger offspring than the parents, and this entails a rapid growth of the size of the individuals in the population. To counteract this problem, in [29] an implementation of GSOs was proposed that makes GSGP not only usable in practice but also significantly faster than standard GP. This is possible through a smart representation of GP individuals, which allows us to not store their genotypes during the evolution. This is the implementation used here. Even though this implementation is efficient, it does not solve the problem of the size of the final model's size: the genotype of the final solution returned by GSGP can be reconstructed, but it is so large that it is practically impossible to understand. This turns GSGP into a "black-box" system, as many other popular machine learning systems are, including deep neural networks.

4. Binary classification with a regression-based GSGP approach

As mentioned in the Introduction, in the work presented here, we use symbolic regression GSOs to implement a system to solve binary classification problems. Our approach is inspired by the functioning of the multi-layer perceptron (with sigmoidal activation function) neural network [16]. To this aim, we transformed the two class labels of the classification problem into the numeric values 0 and 1. This transformation allows binary classification problems to be considered as regression ones, containing only 0 and 1 as possible target values. The output produced by a tree encoded by an individual is given as an input to an activation function. As for the fitness function, we use the root mean square error (RMSE) between the output of the activation function and the binary target values, computed over a training set [1].

As customary in perceptrons, we use the logistic activation function:

$$L(x) = \frac{1}{1 + e^{-\alpha x}} \tag{1}$$

This function has two asymptotes, respectively in 0 and 1. The effect of this function is twofold. On the one hand it makes it possible to constrain the final output of the individual in the range $[0, 1]$, limiting the error between the output of each individual tree and the binary targets (0 or 1). On the other hand, it should favor a better generalization ability of the proposed system by increasing the separation between the samples belonging to different classes. In practice, along the evolutionary process, with the aim of minimizing the error on the training set, our system generates GP individuals

[1] Note that in the following we will use the term 'RMSE' and 'error' interchangeably.

that return larger and larger positive (negative) values for samples labelled as 1(0). At the same time, individuals returning small values for most of the training samples are discarded, as those small outputs are far from the binary target values (0 or 1) and increase the error (fitness) more than large output values. It is worth noting that although the logistic function performs a transformation of the semantic space, it does not affect or alter the ability of GS operators to explore such space.

The extent of the data separation can be tuned by modifying the value of the parameter α of the logistic function (see equation [1]). To have a visual intuition of this, Figure 2 shows two different logistic functions, one with $\alpha = 0.1$ (plot (a)), and the other with $\alpha = 1.0$ (plot (b)).

Looking at the two plots, we can see that to obtain the same approximated output value equal to 1 (respectively 0), when $\alpha = 0.1$ the output has to be larger (respectively smaller), compared to the case $\alpha = 1.0$. As a consequence, given a set of training data for the same value of the error, samples belonging to different classes are better separated for smaller values of α. On the other hand, using small values of α has a drawback: given a prefixed value of the training error ϵ, finding individuals with an error equal to ϵ will be computationally harder for smaller values of α. Thus, finding a value of α providing a good trade-off between generalization ability and computational cost is a challenging task and requires a preliminary tuning phase. The logistic function is used only in the training phase. Indeed, once a model minimizing the fitness (RMSE) on the training data has been found, unseen data are classified according to the following simple rule: observations with positive outputs are predicted as belonging to the same class of

the training samples labeled as 1, whereas the remaining ones are categorized in the opposite class.

5. Experimental study

To assess the effectiveness of the proposed approach, we performed two sets of comparative experiments. In the first set, we investigated the classification performances of the proposed approach as a function of α, which determines the steepness of the logistic function applied on the trees' output (see equation 1 and Figure 2). In the second set, instead, we compared the results achieved by our GSGP-based binary classification system with those of eleven well-known and widely used classification schemes, including four state-of-the-art ensemble methods. The comparison was based on the classification accuracy and the F-score measure. The latter was particularly relevant given the fact that some of the data sets exhibit class-imbalance. [2]

In the following, we detail how classification accuracy and F-score measures were computed. Given a GP individual I representing the function $f_I : R^n \rightarrow R$, where R^n is the feature space, a given data instance $\mathbf{x}$ is labeled as belonging to class 1 if $f_I(\mathbf{x}) > 0$ and to class 0 otherwise. Based on these hard-labels, the accuracy and the F-score are computed. Note that, in practice, the logistic function is only necessary in the training phase, where the fitness function is given by the RMSE. The results reported in the follow-

[2]Note that with the term "class-imbalance" refers to those classification problems in which the classes' proportion is uneven. To mention the percentage of instances belonging to the less represented class, we will use the term "degree of imbalance" (the lower it is, the more imbalanced is a given problem).

ing were computed by averaging those achieved by the best individual found at the end of each run. As for the parameters, we performed some preliminary trials to set them. These parameters were used for all the experiments described below and are reported in Table 3.

5.1. Test problems

The proposed method was tested on 18 real-world data sets, publicly available on the UCI [30] and openML [31] repositories. The considered classification problems present different characteristics in terms of the number of attributes and instances. Moreover, as we were looking for a supervised learning procedure, and in particular for a dichotomizer, the data sets have two target classes. Table 1 reports, for each data set, the number of input features (variables), the number of instances (observations), and the degree of imbalance (i.e., the percentage of instances belonging to the less represented class). To avoid the evolutionary process being biased by the wide variation of the features' ranges, data was normalized to zero mean and unit variance.

To allow a robust statistically-supported comparison of parameters and classification schemes, 30 independent runs were performed for each of the considered problem. At each run, the data was randomly shuffled and split into equally sized training and a test sets. The results reported in the following have been averaged on the errors, and the Accuracy and the F-score were obtained on the 30 independent test partitions.

Table 1: Benchmark problems.

Data set	**#Features**	**#Instances**	**Degree of imbalance**
Blood	4	749	0.24
Clima	20	540	0.09
Eeg	14	14980	0.45
Fertility	9	100	0.12
Gina	970	3468	0.49
Hill	100	1212	0.50
Ilpd	10	579	0.28
Kc	21	2109	0.15
Liver	6	345	0.42
Musk	166	476	0.43
Ozone	72	2534	0.06
Pc1	21	1109	0.07
Pc3	37	1563	0.10
Qsar	41	1055	0.34
Retinopathy	19	1151	0.47
Scene	299	2407	0.18
Spam	57	3037	0.39
Spect	22	267	0.21

5.2. Other classification schemes

To test the effectiveness of our approach, we compared the proposed method with eleven classification schemes chosen from literature. Out of these, seven regard single classification schemes, from now on referred to as *base classifiers*: decision trees (C4.5 algorithm, DT in the following) [32], artificial neural networks (NN) [33], support vector machines (SVM) [34], bayesian network (K2 algorithm, BN in the following) [35, 36], naive bayes (NB) [37], logistic regression (LG) [38] and the standard genetic programming [1]. We used the implementation provided by the WEKA tool [39]. We chose these classifiers because: (i) they are widely used and standard implementations are available for each; (ii) they represent different paradigms of classification algorithms; and (iii) they represent effective classification schemes. The results reported have been achieved after a hyper-parameter optimization step. We used a grid search procedure: once a set of values is defined for each of the parameter to be tuned, this procedure exhaustively tests all parameter combinations. The set of values tested for each of the hyper-parameter tuned is shown in Table 2.

We extended our study by comparing the proposed classification method with four state-of-the-art parallel ensembles, such as Random Forest [40] (RF), Extremely Randomized Trees [41] (E-RF), Rotation Forest [42] (R-RF), and Oblique Random Forest [43, 44] (O-RF). In order to make the forests' comparison as fair as possible, we defined an assortment of common parameter, whereas for the number of trees (N_t) and the maximum number of sampled features at each split (N_f) we performed the grid-search procedure mentioned above on the following sets of values:

- $S_{N_t} = \{50, 100, 200, 500\}$;
- $S_{N_f} = \{\sqrt{m}, \log_2(N_t)\}$;

where m is the number of input features.

5.3. Study of GSGP's parameters

As mentioned in Section 4, our approach introduces a new parameter, called α, which determines the steepness of the logistic function that processes individuals' semantics and shrinks their output to $[0, 1]$ interval. In practice, this parameter affects how the GP trees separate samples belonging to different classes, with small α values requiring greater GP trees outputs to get closer to the asymptotic values (i.e., 0 or 1, representing the class labels; see Figure 2), and then ensuring, at least in principle, a better class separation than greater α values. Moreover, we considered the parameter R that represent the limits of the range of the random constants $[-R, +R]$, thus affecting the input values given as input to the GP trees. Since tuning the parameters of a metaheuristic algorithm is a crucial aspect [45], we performed the experiments detailed below to try to answer the following questions:

i. How much does the setting influence the system performance, especially alpha, which is a new parameter introduced by our method?
ii. How much do the two parameters influence each other?
iii. Is it possible to suggest default values for these two parameters?

To this aim, we performed a factorial experiment. We considered the following sets of values for α and R:

- $S_\alpha = \{0.0001,\ 0.001, 0.01, 0.05, 0.1, 0.2, 0.5, 1.0\}$;

Table 2: Values of the classifier hyper-parameters tested for the base classifiers.

Classifier	**Parameter**	**values**
BN	batch size search	20, 50, 100, 200
	algorithm max.	K2, TabuSearch
	n. parents	1, 2, 3, 4
NB	batch size	20, 50, 100, 200
DT	confidence factor	0.25, 0.50
	min. n. instances per leaf	2, 10, 20
	maximum depth	4, 8, 16
LG	Optimizer	BFGS
	Ridge	$1e{-}8$
	maximum iterations	50,100, 200
	C	1, 10, 100
NN	learning rate	0.001, 0.01, 0.1, 0.3
	momentum	0.2, 0.5, 0.8
	hidden Neurons	10, 100, (#features + #classes)/2
	epochs	500, 1000
SVM	kernel	RBF, sigmoid
	C	1, 10, 100
	γ	0.01, 0.1, 0.5
GP	functions set	$+$, $-$, $*$, and protected /
	terminal set	input variables, constants under $U \sim (-2, 2)$
	population size	500
	# generations	2000
	initialization	ramped-half-and-half (max. depth=6)
	selection	tournament (pool size=4)
	crossover	random trees' swap
	mutation	random sub-tree replacement
	P $(crossover)$	0, 0.3, 0.7, 1
	P $(mutation)$	1, 0.5, 0.3, 0
	elitism	best individual always survives

Table 3: Parameters' settings for the proposed GSGP-based classification scheme.

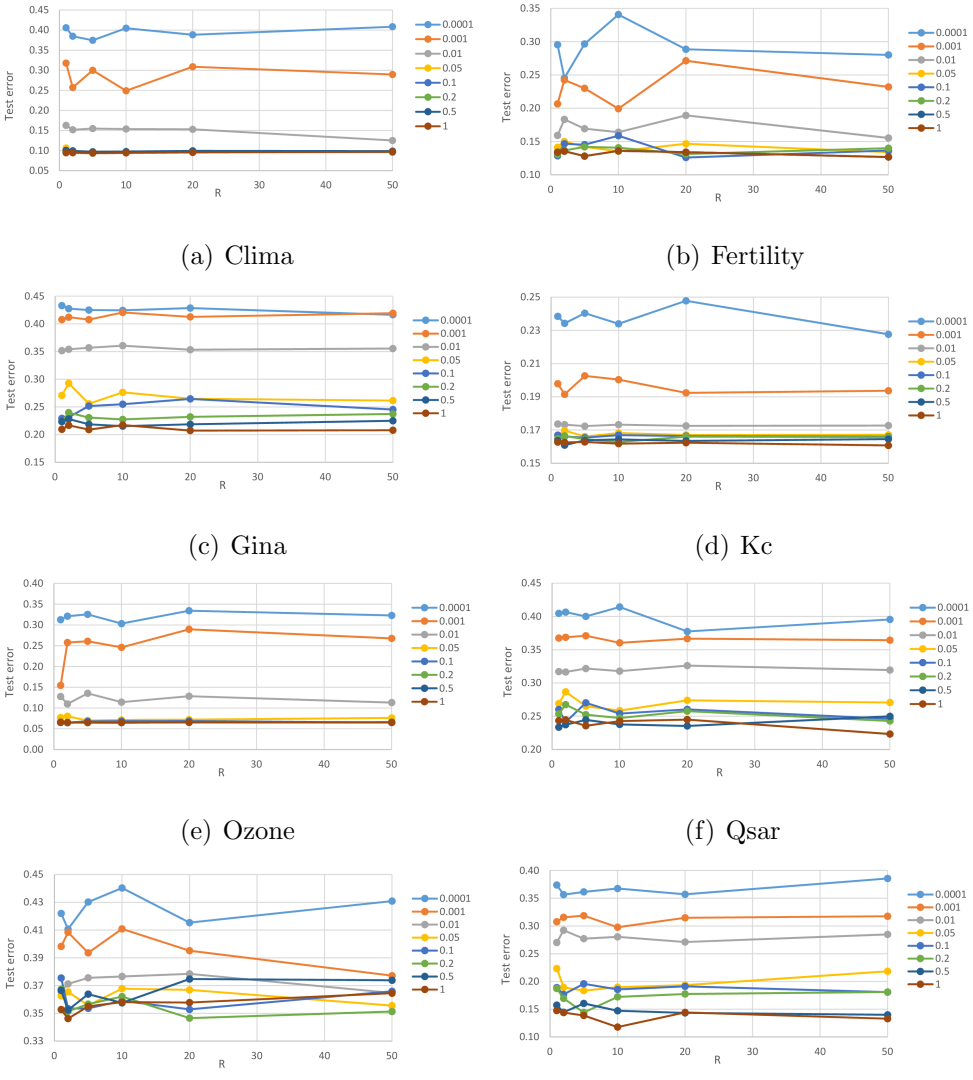
Parameters	**Values**
functions set	$+, -, *$, and protected /
terminal set	input variables, constants under $U \sim (-2,\ 2)$
population size	500
# generations	2000
initialization	ramped-half-and-half, max. depth=6
selection	tournament (pool size=4)
crossover	random trees' swap
mutation	random sub-tree replacement
$P(crossover)$	0.0
$P(mutation)$	1.0
elitism	best individual always survives

- $S_R = \{1, 2, 5, 10, 20, 50\}$.

For the remaining parameters, we used those shown in Table 3.

To try to answer the first two questions listed above, we plotted the test error as a function of R for the different α values tested. For the sake of conciseness, we report the plots for eight of the 16 data sets used. The plots are shown in Figure 3. From Table 4, it can be noted that, for all the considered data sets, the lowest α values achieve the worst performance, especially with the values 0.0001 and 0.001. These results prove that α values that are too low do not improve the "data separation" effect mentioned in Section 4. Most probably, this is because these values produce curves that are too flat (similar to straight lines with a low slope). In practice, these α values produce similar output values for a large range of input values returned by the GP individuals, thus reducing the selective pressure of the evolutionary process.

From the plots, it can be noted that, for a given value of α, the value of R does not affect the performance significantly. To test whether the performance achieved with the different values of R exhibit differences that are statistically significant, we performed the one-way ANOVA statistical test. This test checks the null hypothesis samples from different groups are drawn from populations with the same mean against the alternative hypothesis the population means are not all the same. In our experiment, for each value of $\alpha \in S_\alpha$, we considered as a group of samples the ones obtained by using a fixed value of $R \in S_R$. In practice, for each value of α, we tested whether the means of the test error achieved with the different R values exhibited statistically significant differences. The obtained p-values are shown in Ta-



(a) Clima

(b) Fertility

(c) Gina

(d) Kc

(e) Ozone

(f) Qsar

(g) Retinopathy

(h) Spam

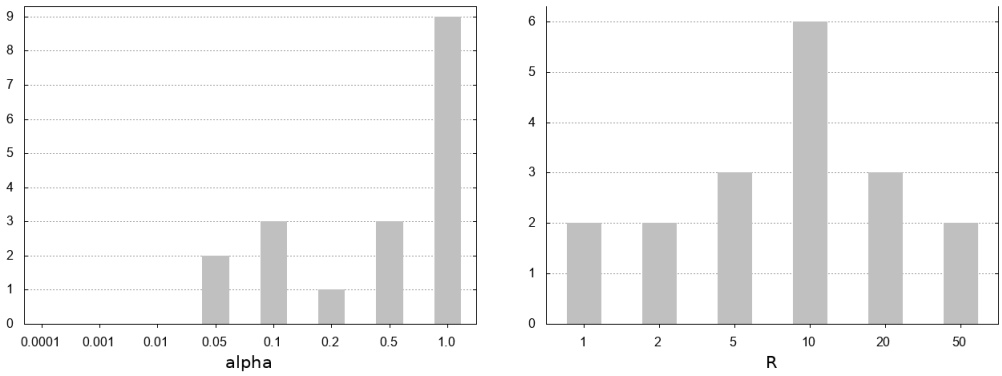
Figure 3: Test error as a function of the limits of the range of the random constants $[-R, +R]$ for the different values of α tested.

ble 4. From Table 4, it can be noted that, in most cases, different R values achieve results that are not statistically different. For some data sets, these differences are statistically significant. However, these differences occurs only for $\alpha = 0.0001$ and $\alpha = 0.001$ which as discussed above, represent the least performing values and then can be discarded from the set of possible values for α. Therefore, these results confirm that, for our system, the choice of R is not critical.

As concerns the third question listed above (i.e., trying to suggest good default values for α and R), we plotted two histograms, for α and R respectively (see Figure 4). For each $\alpha \in S_\alpha$ or $R \in S_R$ value, the histograms show the number of data sets on which that value achieved the best result among all the values tested. From the plots, it can be seen that, as expected, the values of $\alpha \leq 0.01$ never achieved the best performance, whereas the value 1.0 achieved the best performance on nine of the 18 data sets considered in our experiments. Regarding the other values ($0.05 \leq \alpha \leq 0.5$), they equally "share" the best performance, confirming the trends shown by the plots in Figure 3, where it can be seen that, in most cases, values of $\alpha \geq 0.05$ achieved comparable results. As for the parameter R, we can see that there is a "peak" around the value 10, which achieved the best performance on six of the 18 data sets considered, while the adjacent values of 5 and 20 achieved the best results on three data sets. However, the other tested values achieved the best performance on two data sets, confirming the trends shown in Figure 3 and Table 4: the choice of R is not critical for our system because the different values yield comparable results. Therefore, according to the histograms, we have chosen $\alpha = 1.0$ and $R = 10$ as default values. For each data set we

Table 4: P-values of the Anova test. Values less than the significance level chosen (0.05) are in bold.

	0.0001	0.001	0.01	0.1	0.2	0.5	1
Blood	0.57	0.53	0.21	0.74	0.95	0.55	0.96
Clima	0.91	0.03	0.83	0.90	0.13	0.49	0.30
Eeg	0.24	0.59	0.11	0.16	0.71	0.33	0.59
Fertility	**0.01**	**0.02**	0.62	0.14	0.76	0.35	0.90
Gina	0.69	0.78	0.99	0.16	0.78	0.66	0.50
Hill	0.26	0.85	0.51	0.84	0.44	0.58	0.35
Ilpd	0.63	0.12	0.67	0.60	0.62	0.57	0.84
Kc	0.79	0.84	0.85	0.73	0.09	0.28	0.83
Liver	0.59	0.86	0.24	0.07	0.81	0.02	0.55
Musk	0.07	0.88	0.78	0.79	0.43	0.13	0.80
Ozone	0.93	**8e−4**	0.86	0.04	0.44	0.38	0.42
Pc1	0.99	0.08	0.53	0.28	0.31	0.77	0.85
Pc3	0.76	0.57	0.08	0.12	0.98	0.38	0.25
Qsar	**0.03**	0.95	0.95	0.40	0.63	0.87	0.65
Retinopathy	**0.02**	**0.03**	0.78	0.08	0.20	0.16	0.60
Scene	0.79	0.15	0.34	0.46	0.19	0.64	0.34
Spam	0.18	0.61	0.63	0.92	0.31	0.79	0.36
Spect	0.12	0.78	0.96	0.13	0.12	0.06	0.57



(a) alpha (b) R

Figure 4: These histograms show the number of datasets in which that value of α (left) or R (right) achieved the best result amongst all the values tested.

verified whether any differences could be noticed between the performance achieved by using these values and the best obtained value. The results of this comparison are shown in Table 5. From Table 5, it can be seen that, for 13 out of the 18 data sets used for testing our approach, the differences between the accuracies achieved by using the default values and the best ones are not statistically significant. Regarding the remaining data sets, we can note that the differences are not so remarkable, because the relative difference are always less than 5%. Then, we can conclude that the values $\alpha = 1.0$ and $R = 10$ are good candidates as default values for these parameters.

5.4. Comparison results

In this section, we describe the comparison results between the proposed method and the eleven classifiers mentioned above. We compared the results in terms of accuracy and F-score (see Section 5.4.1), both calculated on the test data. The F-score was necessary to viably assess and compare classifiers'

Table 5: Comparison of the results achieved by using the default values tested ($\alpha = 1.0, R = 10$) and the best results. Note that the symbol p (second column) represents the p-value of Wilcoxon test. Values less than the significance level chosen (0.05) are in bold. Note that for the data sets Hill, Musk, Scene, and Spam the parameters that achieved the best results coincide with the default ones.

		Best			$\alpha = 1.0, R = 10$	
Dataset		(α, R)	error		error	
	p		avg	std	avg	std
Blood	**0.001**	(0.05,1)	0.226	0.004	0.232	0.01
Clima	0.504	(1,5)	0.093	0.004	0.094	0.01
Eeg	**0.019**	(0.5,10)	0.382	0.02	0.396	0.02
Fertility	0.821	(0.1,20)	0.126	0.01	0.127	0.02
Gina	0.368	(1,20)	0.207	0.02	0.217	0.04
Hill	1.0	(1,10)	0.365	0.05	0.365	0.05
Ilpd	**3e−4**	(0.05,5)	0.256	0.001	0.269	0.02
Kc	0.589	(1,50)	0.161	0.01	0.162	0.01
Liver	0.063	(0.5,1)	0.353	0.03	0.378	0.05
Musk	1.0	(1,10)	0.242	0.04	0.242	0.04
Ozone	0.389	(0.5,2)	0.065	0.001	0.065	0.001
Pc1	**0.046**	(0.1,10)	0.076	0.002	0.078	0.01
Pc3	**0.004**	(0.1,20)	0.107	0.003	0.111	0.01
Qsar	0.086	(1,50)	0.223	0.05	0.243	0.05
Retinopathy	0.124	(1,2)	0.346	0.04	0.358	0.04
Scene	1.0	(1.0,10)	0.069	0.03	0.069	0.03
Spam	1.0	(1.0,10)	0.118	0.04	0.118	0.04
Spect	0.118	(0.2,5)	0.183	0.02	0.195	0.03

performance in the context of class imbalance.

To statistically validate the comparison results, we performed two non-parametric statistical tests: the Wilcoxon rank-sum test [46] and the Friedman test [47]. The former compares the performance of pairs of classifiers on a single data set, whereas the latter performs a statistical validation of the performance differences for a set of classifiers across multiple data sets. Note that, for both tests, we rejected the null hypothesis when the p-value was less than 0.05.

Table 6 exhibits classifiers' accuracies on the 18 test problems. The first column reports the average accuracy of the proposed GSGP-based classifier, while the following columns regard the seven base classifiers and the four ensembles considered in our study. Whenever GSGP's performance was found to be statistically superior, worse or equivalent to any other classifier (after the Wilcoxon rank-sum test), the symbols '+', '-' and '≈' were used, respectively.

From Table 6 we can be observe that, in terms of win/tie/loss (see last row), GSGP outperformed almost all the base classifiers except the NN classifier: the proposed GSGP-based classification scheme and the NN classi- fier achieved the best performance on eight problems each. What regards GSGP's comparison with the ensemble methods, it can be verified that the latter achieved better accuracy in a significantly larger number of problems. Given the class-imbalance that several problems exhibit, our analysis will be continued in Section 5.4.1 and shade more light on the relative effectiveness of the proposed approach.

Figure 5 shows the average ranking of the compared systems in terms

Table 6: Accuracies on the test set (expressed in percentages) of the proposed GSGP classification approach and seven other base classification schemes, averaged on the test partitions of data. The symbols '+'/'-' denote the GSGP's statistically- sustained win/loss, whereas the symbol '≈' denotes a tie (i.e. the performance differences are not statistically significant according to the Wilcoxon test; p-value > 0.05).

Dataset	**GSGP**	**GP**	**SVM**	**DT**	**NN**	**BN**	**NB**	**LG**	**RF**	**E-RF**	**R-RF**	**O-RF**
Blood	77.4	76.3 +	77.1 ≈	77.4 ≈	87.1 -	75.9 +	76.5 +	77.5 ≈	77.8 ≈	76.5 +	75.6 +	78.9 -
Clima	90.7	87.2 +	92.7 -	89.2 +	87.5 +	93.5 -	89.1 +	90.6 ≈	90.7 ≈	90.7 ≈	90.8 ≈	90.7 ≈
Eeg	61.8	63.7 -	73.0 -	86.4 -	60.2 +	76.1 -	56.3 +	80.1 -	91.5 -	93.4 -	95.7 -	80.9 -
Fertility	87.4	85.4 +	87.9 ≈	85.4 +	83.8 +	88.4 ≈	86.9 ≈	84.1 +	88.1 ≈	88.1 ≈	87.7 ≈	88.1 ≈
Gina	79.3	77.3 ≈	55.2 +	85.6 -	84.5 -	89.8 -	76.9 +	70.7 +	92.5 -	93.2 -	92.4 -	89.8 -
Hill	63.5	90.7 -	57.4 +	59.6 +	59.6 +	54.1 +	53.1 +	85.7 -	57.0 +	59.3 +	60.5 +	100.0 -
Ilpd	74.4	71.2 +	72.3 +	64.8 +	71.8 +	70.6 +	57.4 +	72.2 +	73.1 +	74.5 ≈	72.9 +	77.6 -
Kc	83.9	83.6 ≈	86.6 -	81.1 +	86.8 -	84.0 ≈	81.2 +	78.9 +	86.0 -	84.9 -	84.2 ≈	84.2 ≈
Liver Musk	64.7	64.6 ≈	62.1 +	84.6 -	70.3 -	59.6 +	54.6 +	85.7 -	72.3 -	71.5 -	74.5 -	75.8 -
Ozone	75.8	60.7 +	71.1 +	78.2 -	91.2 -	81.6 -	72.9 +	77.1 ≈	86.9 -	88.9 -	90.1 -	88.1 -
Pc1	93.5	92.5 +	94.9 -	90.9 +	93.7 ≈	92.4 +	90.2 +	91.2 +	94.9 -	95.1 -	95.4 -	93.8 ≈
Pc3	92.4	91.5 +	94.9 -	89.4 -	94.3 -	92.9 -	89.5 +	89.8 +	95.5 -	96.2 -	94.9 -	94.6 -
Qsar	89.3	88.2 +	91.2 -	86.6 +	87.8 +	87.2 +	79.5 +	85.8 +	92.4 -	89.7 ≈	89.5 ≈	92.0 -
Retinophty	77.7	82.8 -	74.7 +	88.8 -	86.3 -	82.2 -	77.3 ≈	89.0 -	89.5 -	89.9 -	89.7 -	89.7 -
Scene Spam	65.3	69.9 -	62.9 +	63.2 +	71.1 -	63.1 +	60.0 +	84.8 -	86.5 -	86.2 -	88.5 -	86.2 -
Spect	93.1	96.8 -	90.9 +	82.3 +	98.0 +	91.9 +	83.5 +	85.9 +	90.2 +	92.9 ≈	95.3 -	89.3 +
win/tie/loss	88.2	92.3 -	84.8 +	91.7 -	89.0 ≈	92.8 -	82.5 +	92.7 -	95.7 -	96.2 -	94.7 -	93.5 -
	81.7	78.6 +	82.0 ≈	80.0 +	79.5 +	78.9 +	76.2 +	80.2 +	85.3 -	86.5 -	81.4 ≈	80.2 +
	–	9/3/6	9/3/6	10/1/7	8/2/8	9/2/7	16/2/0	9/3/6	3/3/12	2/5/11	3/5/10	2/4/12

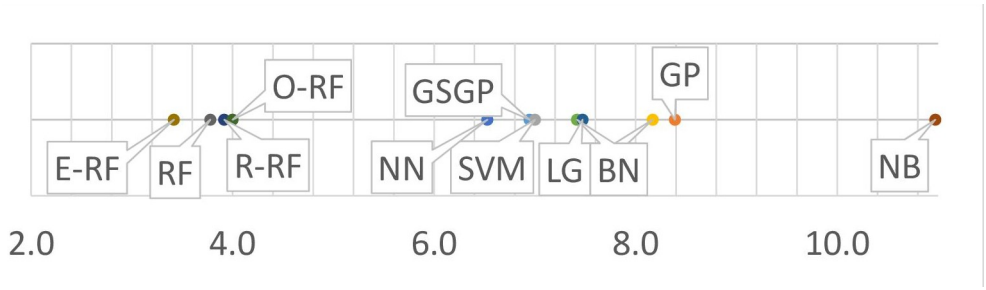


Figure 5: Average ranking for the test accuracy, computed over the 18 data sets considered, achieved by GSGP and the eleven classifiers used for the comparison. The Friedman test returned a p-value equal to 1.92E-12 .

of classification accuracy, computed over the 18 data sets. According to the Friedman test, the results are statistically significant with a p-value of $1.92E{-}12$. The figure shows that the GSGP outperformed all the classifiers, except the ensemble methods However, it is worth pointing out that the ensemble methods used for this comparison combine the responses from at least 50 learned models (see Subsection 5.2), whereas GSGP is based on the learning of a single tree. Finally, as concerns the NN classifier even if it achieved on average a better ranking than GSGP, in the direct comparison it achieved a similar performance (win/tie/loss: 8/2/8, see last row of Table 6).

5.4.1. Imbalanced learning

In order to assess the ability of our system to deal with imbalanced data, we performed two experiments. In the first, we used the F-score measure to compare our results with those achieved by the other methods. In the second, we compared our results with those achieved by using two oversampling methods, namely K-means Smote and Adasyn.

F-Score measure evaluation. The results of the F-score measure are split into two tables, each containing problems with different degrees of imbalance. Table 7 shows the results achieved on the problems with a degree of imbalance less than about 0.2, whereas Table 8 shows those achieved on the remaining problems. The tables shows the F-score, averaged over the test partitions, first for the proposed GSGP-based classification scheme (column **GSGP**), then for the other classifiers considered in our study (these are present in the successive columns of the table). The F-score function was computed as follows:

$$\begin{aligned} F\text{-}score &= 2 \cdot \frac{precision \cdot recall}{precision + recall} \\ precision &= \frac{N_{tp}}{N_{tp} + N_{fp}} \\ recall &= \frac{N_{tp}}{N_{tp} + N_{fn}} \end{aligned} \tag{2}$$

where N_{tp} (N_{fp}) is the number of true (false) positives and N_{tn} (N_{fn}) is the number of true (false) negatives.

From Table 7, we can observe that, in terms of win/tie/loss (see table's last row), the proposed GSGP-based system outperformed all the classifiers used in the comparison (including the ensembles) except the RF classifier: our GSGP-based approach and the RF classifier achieved the best perfor- mance on three problems each. This result demonstrates the effectiveness of our approach, particularly, when solving binary classification problems with class-imbalance.

Looking at Table 8 (degree of imbalance $\gtrapprox$ 0.20), we can see that GSGP outperformed six out of seven base classifiers. The only base classifier that

Table 7: F-score values of the proposed GSGP classification approach and eleven other classification schemes, averaged on the test partitions of data sets with a degree of imbalance $\lessapprox$ 0.20. The symbols '+'/'-' denote GSGP's statistically-sustained win/loss, whereas the symbol '$\approx$' denotes a tie, i.e., the performance differences are not statistically significant according to the Wilcoxon test (p-value$>$ 0.05).

Dataset	**GSGP**	**GP**	**SVM**	**DT**	**NN**	**BN**	**NB**	**LG**	**RF**	**E-RF**	**R-RF**	**O-RF**
Clima (0.09)	0.94	$\approx$	-	$\approx$	+	+	$\approx$	-				
Fertility (0.12)	0.40	+	$\approx$	+	+	$\approx$	-	+	$\approx$	+	$\approx$	+
Kc (0.15)	0.47	+	+	+	+	+	+	+	+	+	-	$\approx$
Ozone (0.06)	0.36	$\approx$	+	+	+	+	$\approx$	$\approx$	$\approx$	+	-	-
Pc1 (0.07) Pc3	0.27	-	-	+	+	+	+	+	$\approx$	**0.33**	$\approx$	+
(0.10) Scene	0.31	+	+	-	-	-	$\approx$	$\approx$	$\approx$	+	+	$\approx$
(0.18) Spect	0.78	+	+	+	$\approx$	-	$\approx$	$\approx$	5/2/1	5/2/1	4/3/1	4/2/2
(0.21)	0.88	5/3/0	4/2/2	4/2/2	3/2/3	4/1/3	3/3/2	5/1/2				
win/tie/loss	-											

Table 8: F-score values of the proposed GSGP classification approach and eleven other classification schemes, averaged on the test partitions of data sets with a degree of imbalance $\gtrapprox$ 0.20. The symbols '+'/'-' denote GSGP's statistically-sustained win/loss, whereas the symbol '$\approx$' denotes a tie, i.e., the performance differences are not statistically significant according to the Wilcoxon test (p-value$>$ 0.05).

Dataset GSGP		**GP**	**SVM**	**DT**	**NN**	**BN**	**NB**	**LG**	**RF**	**E-RF**	**R-RF**	**O-RF**
Blood (0.24)	0.46	0.30	+ 0.15	+ 0.40	+ 0.48	- 0.13	+ 0.27	+ 0.22	+ 0.38	+ 0.39	+ 0.38	+ 0.35	+ 0.51	- 0.73	- 0.85	- 0.37
Eeg (0.45)	0.40	+ 0.69	- 0.29	+ 0.69	- 0.90	- 0.92	- 0.91	- 0.80	- 0.74	$\approx$ 0.54	+ 0.84	- 0.84	- 0.89	- 0.75	+ 0.71	+ 0.92	-
Gina (0.49)	0.77	0.93	- 0.92	- 0.90	- 0.92	- 0.60	+ 0.42	+ 0.50	+ 0.37	+ 0.54	+ 0.87	- 0.59	+ 0.60	+ 0.61	+ 1.0	- 0.32	+
Hill (0.50)	0.75	0.30	+ 0.34	+ 0.37	+ 0.34	- 0.57	- 0.35	+ 0.37	+ 0.43	+ 0.36	+ 0.40	+ 0.72	+ 0.77	$\approx$ 0.83	- 0.75	+		
Ilpd (0.28)	0.51	0.69	+ 0.52	+ 0.85	- 0.77	$\approx$ 0.75	+ 0.77	$\approx$ 0.80	- 0.52	+ 0.66	+ 0.61	+ 0.72	$\approx$ 0.58	+ 0.36	+ 0.74	-		
Liver(0.58)	0.77	0.80	- 0.85	- 0.86	- 0.84	- 0.74	$\approx$ 0.68	+ 0.81	- 0.78	- 0.74	$\approx$ 0.74	$\approx$ 0.82	- 0.83	- 0.83	- 0.83	- 0.77	- 0.90	
Musk (0.43)	0.71	- 0.70	+ 0.61	+ 0.71	+ 0.58	+ 0.40	+ 0.83	- 0.80	- 0.78	- 0.83	- 0.80	- 0.90	- 0.59	+ 0.88	$\approx$ 0.87	$\approx$ 0.90		
Qsar (0.34)	0.74	- 0.81	+ 0.90	- 0.92	- 0.92	- 0.91	- 0.90	- 4/2/4	8/2/1	5/1/4	5/2/3	5/1/4	8/1/1	3/0/7	3/1/6	4/0/6		
Retinophaty (0.53)	0.73	3/1/6	2/0/8															
Spam (0.39)	0.87																	
win/tie/loss	-																	

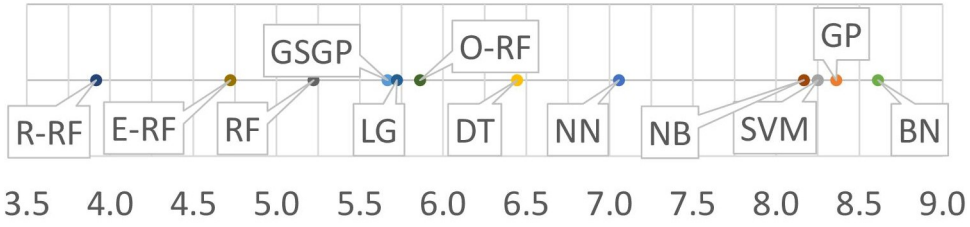


Figure 6: Average ranking for the test F-score, computed over the 18 considered data sets, achieved by GSGP and the eleven classifiers used for the comparison. The Friedman test returned a p-value equal to $p = 4.63E - 06$

better solved more problems was LG. What regards the ensembles, as is the case of the accuracy (see Table 6), they achieved better accuracy in a significantly larger number of problems. This result was expected because, as mentioned above, ensemble methods combine responses from multiple models and they typically outperform classification or regression methods [48] based on the learning of a single model. However, unlike it was expected, the same cannot be observed when the assessment involves a higher class-imbalance (see Table 7). Such a paradoxical situation has an explanation. It happens that ensembles' base-learners (i.e., the classifiers that compose the ensemble), are trained on a bootstrap sample of the training data sample, which means that, in a given sample, some observations may be repeated, while others not appear at all. From the literature, when the bootstrap's sample size equals the size of the sample from which it was taken, when the sample size is *large*, one can expect about 63.2% unique data instances (the rest being duplicates) [49]. In this sense, when working with an imbalanced dataset, it is likely that a bootstrap sample contains few or even none of the

minority class, resulting in a base-learner that poorly predicts the minority class [50]. To avoid the aforementioned bias by sample, several approaches were developed by the scientific community. For example, instances' classes representation can be taken into account when extracting a bootstrap sample by using stratified random sampling (aka undersampling). Alternatively, the loss function can be changed to penalize the base-learners when mistaking the prediction of data instances from the minority class(es). These and other techniques can be found in [51].

Figure 6 shows the average ranking of the compared systems in terms of F-score, computed over the 18 data sets. According to the Friedman test, the results are statistically significant with a p-value of $4.63E-06$. The figure shows that the GSGP outperforms all the classifiers, except the ensemble methods, confirming its effectiveness when compared with classification schemes consisting of single models.

In general terms, one can conclude that the results shown in Table 7 and Table 8 as well as those shown in Figure 6 confirm the effectiveness of the proposed classification approach, especially in dealing with imbalanced class distribution.

Oversampling methods. To further assess the ability of our approach to deal with imbalanced problems, we compared our results (achieved without using any oversampling method but only using the F-score measure as fitness function) with those achieved by two oversampling methods, namely K-means Smote [52] and Adasyn [53]. However, for the Spect data set we used the Smoten approach. We made this choice because Spect data set consists of binary features only.

Table 9: F-score values of the proposed GSGP classification approach and eleven other classification schemes, averaged on the test partitions of data sets with a degree of imbalance $\lessapprox 0.20$. The eleven other classification schemes have been trained on oversampled dataset. The symbols '+'/'-' denote GSGP's statistically-sustained win/loss, whereas the symbol '$\approx$' denotes a tie, i.e., the performance differences are not statistically significant according to the Wilcoxon test (p-value> 0.05).

Dataset	**Method**	**GSGP**	**GP SVM DT NN BN NB LG RF E-RF R-RF O-RF** 0.97 - 0.97 - 0.96 - 0.91 + 0.92 + 0.95
Clima	Adasync	0.94	≈ 0.97 - 0.98 - 0.97 - 0.97 - 0.98 - 0.97 - 0.98 - 0.95 ≈ 0.92 + 0.93 + 0.96 - 0.98 - 0.97 - 0.97 - 0.97 -
	KMeans-Smotenc		0.99 - 0.31 + 0.48 - 0.45 - 0.36 + 0.47 - 0.48 - 0.33 + 0.40 ≈ 0.40 ≈ 0.44 - 0.40 ≈ 0.35 + 0.50 - 0.46 -
Fertility	Adasync	0.40	0.35 + 0.51 - 0.49 - 0.37 + 0.42 - 0.41 ≈ 0.45 - 0.42 - 0.33 + 0.39 + 0.43 + 0.38 + 0.39 + 0.47 ≈ 0.41
	KMeans-Smotenc		+ 0.41 + 0.40 + 0.43 + 0.35 + 0.36 + 0.41 + 0.45 + 0.41 + 0.41 + 0.49 - 0.41 + 0.42 + 0.43 + 0.42
Kc	Adasync	0.47	+ 0.37 + 0.31 + 0.37 ≈ 0.36 ≈ 0.40 - 0.43 - 0.32 + 0.43 - 0.31 + 0.29 + 0.36 ≈ 0.36 ≈ 0.29 + 0.37 -
	KMeans-Smotenc		0.38 - 0.42 - 0.41 - 0.32 + 0.45 - 0.29 + 0.29 + 0.38 - 0.35 ≈ 0.31 - 0.28 ≈ 0.32 - 0.28 ≈ 0.27 ≈ 0.38 -
Ozone	Adasync	0.36	0.30 - 0.49 - 0.47 - 0.52 - 0.50 - 0.33 - 0.28 ≈ 0.35 - 0.30 - 0.30 - 0.38 - 0.28 ≈ 0.53 - 0.51 - 0.52 - 0.50 -
	KMeans-Smotenc		0.25 + 0.27 + 0.29 + 0.30 + 0.27 + 0.33 - 0.37 - 0.29 + 0.25 + 0.29 + 0.24 + 0.25 + 0.25 + 0.30 ≈
Pc1	Adasync	0.27	0.31 ≈ 0.27 + 0.33 - 0.39 - 0.30 + 0.25 + 0.28 + 0.29 + 0.96 - 0.76 + 0.96 - 0.98 - 0.82 - 0.60 + 0.90 -
	KMeans-Smotenc		0.70 + 0.81 - 0.88 - 0.60 + 0.95 - 0.77 ≈ 0.96 - 0.98 - 0.83 - 0.65 + 0.92 - 0.65 + 0.76 ≈ 0.85 - 0.58 +
Pc3	Adasync	0.31	0.91 - 0.94 - 0.92 - 0.93 - 0.44 + 0.89 ≈ 0.92 - 0.95 - 0.94 - 0.95 - 0.96 -
	KMeans-Smotenc		
Scene	Adasync	0.78	
	KMeans-Smotenc		
Spect	Smoten	0.88	
win or tie/loss		–	8/7 9/6 6/9 9/6 8/7 7/8 5/10 9/7 9/6 5/10 9/6

Table 9 shows the F-score results achieved on the problems with a degree of imbalance less than about 0.2, averaged over the test partitions. Note that in this case we categorized the comparison results in two categories: win or tie vs loss. This because the tie result can be considered as favourable for our approach because it was achieved without the costly procedure of data oversampling, but just using the F-score measure as fitness function.

From the table we can observe that, in terms of "win or tie vs loss", the proposed GSGP-based system outperformed seven out of the eleven classifiers used in the comparison. This can be considered a good result since, as mentioned above, it was achieved without using any specifically devised strategy to deal with imbalanced data, but only using the F-score measure as fitness function.

6. Conclusions and future work

GSGP is a recent enhancement of GP based on the use of semantic operators that allow the fitness landscape of any supervised learning problem to be transformed into a new one characterized by the absence of locally suboptimal solutions, for which the fitness of a tree is computed on the training set by summing the distance (error) between the target value of each sample and the corresponding output values calculated by the tree. These semantic operators have proven their effectiveness in many real-world regression problems.

In this paper, we presented an extension of GSGP to solve binary classification problems. The idea behind the extension, inspired by the functioning of the perceptron artificial neural network, is extremely simple: the value

computed by a GSGP tree (a floating-point number, typically used for regression) is given as input to a sigmoid, to limit the output value in the interval $[0, 1]$, where the values 0 and 1 are the two class labels. In the experiments presented here we first investigated how the new parameter introduced (i.e., the alpha of the sigmoid) affects the classification performance. Then, we compared the results of our approach with those achieved by eleven state-of-the-art classification algorithms, including four ensemble-based methods. Finally, we used the F-score function as fitness to deal with imbalanced data.

The results of the first set of experiments showed that:

- too small values of α (≤ 0.001) achieve bad performances, which is due to the fact that these values produce too flat curves, returning similar output values for very large ranges of input values; this similarity has the effect of reducing the selective pressure of the evolutionary process;
- the value of the GP-specific parameter R (it represents the limits of the range of the random constants) does not affect the performance significantly, confirming that its choice is not critical for the performance of our system;
- the values $\alpha = 1.0$ and $R = 10$ are good candidates as default values for these parameters.

Regarding the comparison with other state-of-the-art methods, we performed two nonparametric statistical tests, namely the Wilcoxon rank-sum test and the Friedman test. The former confirmed that GSGP outperforms six of the seven base classifiers used for the comparison in terms of win/tie/loss on the single data sets, whereas, as expected, ensemble methods outperformed

GSGP. The latter (computed over the 18 data sets used) confirmed that GSGP is the top ranked among the classifiers we compared, except the ensemble methods. Finally, on imbalanced data (represented by six of the 18 data sets used) GSGP, using the F-score as the fitness function, outperforms all the classifiers used for the comparison, confirming the effectiveness of the proposed approach in dealing with imbalanced data.

Future work will focus on investigating two aspects. First, to extend the proposed approach to multi-class problems, we will investigate the use of combination rules such as the one-versus-all and the one-versus-one, already successfully used for binary classifiers, such as SVM. Second, we will use ensemble and combining techniques [54, 55] to improve further the classification performance achieved.

References

[1] J. R. Koza, Genetic Programming: On the Programming of Computers by Means of Natural Selection, MIT Press, Cambridge, MA, USA, 1992.

[2] P. G. Espejo, S. Ventura, F. Herrera, A survey on the application of genetic programming to classification, IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews) 40 (2) (2010) 121–144.

[3] H. Jabeen, A. R. Baig, Review of classification using genetic programming, International Journal of Engineering Science and Technology 2 (2) (2010) 94–103.

[4] K. Nag, N. R. Pal, Genetic Programming for Classification and Feature Selection, Springer International Publishing, Cham, 2019, pp. 119–141.

[5] N. Al-Madi, S. A. Ludwig, Improving genetic programming classification for binary and multiclass datasets, in: 2013 IEEE Symposium on Computational Intelligence and Data Mining (CIDM), 2013, pp. 166–173.

[6] L. Muñoz, S. Silva, L. Trujillo, M3gp – multiclass classification with gp, in: P. Machado, M. I. Heywood, J. McDermott, M. Castelli, P. García-Sánchez, P. Burelli, S. Risi, K. Sim (Eds.), Genetic Programming, Springer International Publishing, Cham, 2015, pp. 78–91.

[7] W. La Cava, S. Silva, L. Vanneschi, L. Spector, J. Moore, Genetic programming representations for multi-dimensional feature learning in biomedical classification, in: G. Squillero, K. Sim (Eds.), Applications of Evolutionary Computation, Springer International Publishing, Cham, 2017, pp. 158–173.

[8] C. De Stefano, F. Fontanella, G. Folino, A. Di Freca, A bayesian approach for combining ensembles of gp classifiers, Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 6713 LNCS (2011) 26–35. doi:10.1007/978-3-642-21557-5_5.

[9] C. De Stefano, G. Folino, F. Fontanella, A. Scotto Di Freca, Using bayesian networks for selecting classifiers in gp ensembles, Information Sciences 258 (2014) 200–216. doi:10.1016/j.ins.2013.09.049.

[10] Y. Bi, B. Xue, M. Zhang, An automated ensemble learning framework using genetic programming for image classification, in: Proceedings of the Genetic and Evolutionary Computation Conference, GECCO '19, Association for Computing Machinery, New York, NY, USA, 2019, p. 365–373. doi:10.1145/3321707.3321750.

[11] I. De Falco, A. Della Cioppa, T. Koutny, M. Krcma, U. Scafuri, E. Tarantino, Genetic programming-based induction of a glucose-dynamics model for telemedicine, Journal of Network and Computer Applications 119 (2018) 1 – 13. doi:https://doi.org/10.1016/j.jnca.2018.06.007.

[12] A. Moraglio, K. Krawiec, C. Johnson, Geometric semantic genetic programming, in: C. Coello, V. Cutello, K. Deb, S. Forrest, G. Nicosia, M. Pavone (Eds.), Parallel Problem Solving from Nature - PPSN XII, Vol. 7491 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2012, pp. 21–31.

[13] L. Vanneschi, An introduction to geometric semantic genetic programming, in: O. Schutze, L. Trujillo, P. Legrand, Y. Maldonato (Eds.), Results of the Numerical and Evolutionary Optimization Workshop (NEO 2015), Springer, 2017, pp. 3–42.

[14] L. Vanneschi, S. Silva, M. Castelli, L. Manzoni, Geometric Semantic Genetic Programming for Real Life Applications, Springer New York, New York, NY, 2014, pp. 191–209.

[15] I. Bakurov, M. Castelli, F. Fontanella, L. Vanneschi, A regression-like

classification system for geometric semantic genetic programming, in: IJCCI 2019 - Proceedings of the 11th International Joint Conference on Computational Intelligence, 2019, pp. 40–48.

[16] L. Vanneschi, M. Castelli, Multilayer perceptrons, in: S. Ranganathan, M. Gribskov, K. Nakai, C. Schönbach (Eds.), Encyclopedia of Bioinformatics and Computational Biology, Academic Press, Oxford, 2019, pp. 612 – 620. doi:https://doi.org/10.1016/B978-0-12-809633-8.20339-7.

[17] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, The MIT Press, 2016.

[18] E. Z-Flores, L. Trujillo, O. Schütze, P. Legrand, A local search approach to genetic programming for binary classification, in: Proceedings of the 2015 annual conference on genetic and evolutionary computation, 2015, pp. 1151–1158.

[19] D. C. Sorensen, Newton's method with a model trust region modification, SIAM Journal on Numerical Analysis 19 (2) (1982) 409–426.

[20] V. Ingalalli, S. Silva, M. Castelli, L. Vanneschi, A multi-dimensional genetic programming approach for multi-class classification problems, in: M. Nicolau, K. Krawiec, M. I. Heywood, M. Castelli, P. García-Sánchez, J. J. Merelo, V. M. Rivas Santos, K. Sim (Eds.), Genetic Programming, Springer Berlin Heidelberg, Berlin, Heidelberg, 2014, pp. 48–60.

[21] W. G. L. Cava, S. Silva, K. Danai, L. Spector, L. Vanneschi, J. H. Moore, Multidimensional genetic programming for

multiclass classification, Swarm Evol. Comput. 44 (2019) 260–272. doi:10.1016/j.swevo.2018.03.015.

[22] W. G. L. Cava, J. H. Moore, Semantic variation operators for multidimensional genetic programming, CoRR abs/1904.08577 (2019).

[23] M. Castelli, L. Vanneschi, S. Silva, Prediction of high performance concrete strength using genetic programming with geometric semantic genetic operators, Expert Systems with Applications 40 (17) (2013) 6856 – 6862. doi:https://doi.org/10.1016/j.eswa.2013.06.037.

[24] I. Bakurov, M. Castelli, L. Vanneschi, M. J. Freitas, Supporting medical decisions for treating rare diseases through genetic programming, in: P. Kaufmann, P. A. Castillo (Eds.), Applications of Evolutionary Computation, Springer International Publishing, Cham, 2019, pp. 187–203.

[25] L. Vanneschi, M. Castelli, L. Manzoni, S. Silva, A new implementation of geometric semantic gp and its application to problems in pharmacokinetics, in: K. Krawiec, A. Moraglio, T. Hu, A. Ş. Etaner-Uyar, B. Hu (Eds.), Genetic Programming, Springer Berlin Heidelberg, Berlin, Heidelberg, 2013, pp. 205–216.

[26] M. Castelli, L. Vanneschi, S. Silva, Prediction of the unified parkinson's disease rating scale assessment using a genetic programming system with geometric semantic genetic operators, Expert Systems with Applications 41 (10) (2014) 4608 – 4616. doi:https://doi.org/10.1016/j.eswa.2014.01.018.

[27] L. Vanneschi, M. Castelli, I. Gonçalves, L. Manzoni, S. Silva, Geometric semantic genetic programming for biomedical applications: A state of the art upgrade, in: 2017 IEEE Congress on Evolutionary Computation (CEC), 2017, pp. 177–184.

[28] P. Hajek, R. Henriques, M. Castelli, L. Vanneschi, Forecasting performance of regional innovation systems using semantic-based genetic programming with local search optimizer, Computers & Operations Research 106 (2019) 179 – 190. doi:https://doi.org/10.1016/j.cor.2018.02.001.

[29] M. Castelli, S. Silva, L. Vanneschi, A c++ framework for geometric semantic genetic programming, Genetic Programming and Evolvable Machines 16 (1) (2015) 73–81.

[30] D. Dua, E. Karra Taniskidou, UCI machine learning repository (2017). URL `http://archive.ics.uci.edu/ml`

[31] J. Vanschoren, J. N. van Rijn, B. Bischl, L. Torgo, Openml: Networked science in machine learning, SIGKDD Explorations 15 (2) (2013) 49–60. doi:10.1145/2641190.2641198.

[32] J. R. Quinlan, C4.5: Programs for Machine Learning, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993.

[33] T. Hastie, R. Tibshirani, J. Friedman, The elements of statistical learning: data mining, inference and prediction, 2nd Edition, Springer, 2009.

[34] C. Cortes, V. Vapnik, Support-vector networks, Machine Learning 20 (3) (1995) 273–297.

[35] J. Pearl, Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference, Morgan Kaufmann, 1988.

[36] G. Cooper, E. Herskovits, A bayesian method for the induction of probabilistic networks from data, Machine Learning 9 (4) (1992) 309–347.

[37] H. Zhang, The optimality of naive bayes, in: V. Barr, Z. Markov (Eds.), Proceedings of the Seventeenth International Florida Artificial Intelligence Research Society Conference (FLAIRS 2004), AAAI Press, 2004.

[38] S. le Cessie, J. van Houwelingen, Ridge estimators in logistic regression, Applied Statistics 41 (1) (1992) 191–201.

[39] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, I. H. Witten, The weka data mining software: an update, SIGKDD Explorations 11 (1) (2009) 10–18.

[40] L. Breiman, Random forests, Machine Learning 45 (1) (2001) 5–32. doi:10.1023/A:1010933404324.

[41] P. Geurts, D. Ernst, L. Wehenkel, Extremely randomized trees, Mach. Learn. 63 (1) (2006) 3–42. doi:10.1007/s10994-006-6226-1.

[42] J. Rodriguez, L. Kuncheva, C. Alonso, Rotation forest: A new classifier ensemble method, IEEE Transactions on Pattern Analysis and Machine Intelligence 28 (10) (2006) 1619–1630. doi:10.1109/TPAMI.2006.211.

[43] L. Zhang, P. N. Suganthan, Benchmarking ensemble classifiers with novel co-trained kernel ridge regression and random vector functional

link ensembles [research frontier], IEEE Computational Intelligence Magazine 12 (4) (2017) 61–72. doi:10.1109/MCI.2017.2742867.

[44] R. Katuwal, P. Suganthan, L. Zhang, Heterogeneous oblique random forest, Pattern Recognition 99 (2020) 107078. doi:https://doi.org/10.1016/j.patcog.2019.107078.

[45] E. Osaba, E. Villar-Rodriguez, J. Del Ser, A. J. Nebro, D. Molina, A. LaTorre, P. N. Suganthan, C. A. Coello Coello, F. Herrera, A tutorial on the design, experimentation and application of metaheuristic algorithms to real-world optimization problems, Swarm and Evolutionary Computation 64 (2021) 100888. doi:https://doi.org/10.1016/j.swevo.2021.100888.

[46] W. Haynes, Wilcoxon Rank Sum Test, Springer New York, New York, NY, 2013, pp. 2354–2355.

[47] J. Demšar, Statistical comparisons of classifiers over multiple data sets, Journal of Machine Learning Research 7 (2006) 1–30.

[48] L. Rokach, Pattern Classification Using Ensemble Methods, World Scientific Publishing Co., Inc., USA, 2010.

[49] B. Efron, R. Tibshirani, Improvements on cross-validation: The .632+ bootstrap method, Journal of the American Statistical Association 92 (438) (1997) 548–560.

[50] C. Chen, L. Breiman, Using random forest to learn imbalanced data, Report (01 2004).

[51] A. Fernández1, S. García, M. Galar, R. C. Prati, B. Krawczyk, F. Herrera, Learning from Imbalanced Data Sets, Springer, Cham, 2018.

[52] G. Douzas, F. Bacao, F. Last, Improving imbalanced learning through a heuristic oversampling method based on k-means and smote, Information Sciences 465 (2018) 1–20.

[53] H. He, Y. Bai, E. A. Garcia, S. Li, Adasyn: Adaptive synthetic sampling approach for imbalanced learning, in: 2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence), 2008, pp. 1322–1328.

[54] C. De Stefano, F. Fontanella, A. Scotto Di Freca, A novel naive bayes voting strategy for combining classifiers, in: Proceedings - International Workshop on Frontiers in Handwriting Recognition, IWFHR, 2012, pp. 467–472. doi:10.1109/ICFHR.2012.166.

[55] L. Cordella, C. De Stefano, F. Fontanella, A. Scotto Di Freca, A weighted majority vote strategy using bayesian networks, Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 8157 LNCS (PART 2) (2013) 219–228. doi:10.1007/978-3-642-41184-7_23.