Soft Computing (2020) 24:18021–18038
https://doi.org/10.1007/s00500-020-05056-7

METHODOLOGIES AND APPLICATION

Genetic programming for high-dimensional imbalanced classification with a new fitness function and program reuse mechanism

Wenbin Pei[1] · Bing Xue[1] · Lin Shang[2] · Mengjie Zhang[1]

Published online: 22 June 2020

Abstract

Genetic programming (GP) has been successfully applied to classification. However, GP may evolve biased classifiers when encountering the problem of class imbalance. These biased classifiers are often not reliable to be applied to some real-world applications. High dimensionality makes it more difficult for classifiers to effectively separate the majority class and the minority class. The use of GP to handle the joint effect of high dimensionality and class imbalance has not been heavily investigated. In this paper, we propose a GP approach to high-dimensional imbalanced classification, with the goals of increasing the classification performance as well as saving training time. To achieve this goal, a new fitness function is developed to solve the problem of class imbalance, and moreover, a strategy is proposed to reuse previous good GP individuals for improving efficiency. The proposed method is examined on ten high-dimensional imbalanced datasets. Experimental results show that, for high-dimensional imbalanced classification, the proposed method generally outperforms other GP methods and traditional classification algorithms using sampling methods to solve the problem of class imbalance.

Keywords Genetic programming · Fitness function · Class imbalance · High dimensionality

1 Introduction

Genetic programming (GP) (Poli et al. 2008) automatically generates computer programs that are often represented as trees. Classification, a common supervised learning task, refers to a procedure to assign a given instance into its corresponding category or class (Tan et al. 2016). GP has been successfully applied to feature selection and feature construction for addressing the curse of dimensionality issue for many classification algorithms in machine learning (Tran et al. 2016). More importantly, GP can directly construct classifiers (Espejo et al. 2010; Luna et al. 2017).

However, GP may develop the biased classifiers in imbalanced classification if the problem of class imbalance is not well-addressed (Bhowan et al. 2012). Class imbalance is a common issue in some domains, such as fraud detection, medical diagnosis, financial analysis of loan policy or bankruptcy, and text classification (Batista et al. 2004; Chawla et al. 2004). Learning from imbalanced data, not only GP methods, many classification algorithms, e.g. support vector machines (SVMs) and decision trees (DTs), also suffer from a performance bias issue (Fleury et al. 2010; Joshi et al. 2016). These classification algorithms are often biased towards the majority class, ignoring the minority class to some degree. Unfortunately, the minority class is at least as important as the majority class in some applications, e.g. medical diagnosis.

To address the problem of class imbalance, methods could be divided into two groups, at the data level and at the algorithmic level. At the data level, sampling methods are widely used, which often require to rebalance the imbalanced datasets (Batista et al. 2004; He et al. 2008; Liu et al. 2009; Pears et al. 2014; Ramentol et al. 2012; Yen and Lee 2009).

Communicated by V. Loia.

✉ Wenbin Pei
Wenbin.Pei@ecs.vuw.ac.nz

Bing Xue
Bing.Xue@ecs.vuw.ac.nz

Lin Shang
shanglin@nju.edu.cn

Mengjie Zhang
Mengjie.Zhang@ecs.vuw.ac.nz

1 School of Engineering and Computer Science, Victoria University of Wellington, P.O. Box 600, Wellington 6140, New Zealand

2 State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210093, China

However, sampling methods need to change original information. At the algorithmic level, cost-sensitive learning is often used by classification algorithms to internally handle the problem of class imbalance, based on cost information (Freund and Schapire 1997; Fan et al. 1999; Joshi et al. 2001; Zhang et al. 2005; Zhou and Liu 2006). However, cost-sensitive learning often needs domain knowledge to design a cost matrix. One-class learning, kernel modification method, and active learning are also used by classification algorithms to solve the problem of class imbalance at the algorithmic level (Tax and Duin 2004; Hong et al. 2007; Li et al. 2006; Tashk and Faez 2007; Wu and Chang 2005; Ertekin et al. 2007a, b).

In GP, a fitness function is an important component to guide an evolutionary process. Traditionally, many GP methods for classification often employ the overall classification accuracy or error rate as a fitness function. However, this fitness function considers all training instances being equally important. As a result, in imbalanced classification, constructed classifiers are biased towards the majority class. Therefore, some new fitness functions have been developed for addressing the problem of class imbalance (Bhowan et al. 2012, 2010; Patterson and Zhang 2007). Evolutionary multi-objective optimization (EMO) also provides an effective solution for GP to solve the problem of class imbalance. Bhowan et al. (2011b, a, 2013, 2014) developed multi-objective GP (MOGP) methods for imbalanced classification. However, MOGP is often time-consuming to obtain the complete Pareto front.

There are a growing number of imbalanced datasets, where the number of features is far more than the number of instances. This kind of imbalanced datasets may further encounter the curse of dimensionality issue. High dimensionality makes it more challenging to solve the problem of class imbalance, and vice versa. The main difficulty of high-dimensional imbalanced classification is mainly from the following factors:

- Sparsity or dissimilarity of data if high-dimensional imbalanced datasets have a very small number of instances.
  A data space becomes sparse if the features (or dimensions) significantly outnumber instances. The similarity among instances in a high-dimensional space becomes weak, which may increase the difficulty in generating the classifiers with a good generalization ability.
  In high-dimensional imbalanced classification, both the minority class and the majority class do not have a sufficient number of instances. As a consequence, it is prone for classifiers to be overfitting to the training data.
- Overlap or class separability.
  There might be an overlapping area, where the prior probabilities of both classes are almost the same (Haixiang et al. 2017). Therefore, it is often difficult for classifiers to effectively discriminate boundaries of the majority class and the minority class in this area.
- Within-class imbalance (or called small disjuncts) (Galar et al. 2012).
  A single class is composed of several sub-clusters and each cluster does not always contain the same number of instances, which results in within-class imbalance. Within-class imbalance may increase the complexity of a dataset (Stefanowski 2016).
- The influence of noise (Seiffert et al. 2014; Hsieh 2007). The presence of noise may increase a risk of overfitting (i.e. fitting to noise) (Hsieh 2007).

To address the curse of dimensionality issue, feature selection is often used to select a smallest subset of features that are necessary and sufficient to describe the target labels (Tran et al. 2016). However, in classification with high-dimensional data, feature selection is challenging because of a large search space (2^n, n is a number of original features) and complicated feature interactions.

When GP is used to construct classifiers, GP has a built-in capability to automatically select good-quality features that improve the classification performance. Therefore, GP has some benefits for high-dimensional classification. However, GP methods are often time-consuming. Pei et al. (2018) proposed a GP method to enhance the classification performance and reduce training time by using multiple GP processes for high-dimensional classification. However, this work does not specifically consider the problem of class imbalance, and moreover, the later GP processes initialize their population without considering the previous building blocks.

In this paper, we adopt multiple GP processes, and the later GP processes (after the first GP process) is able to reuse previous building blocks from an earlier GP process to further improve the effectiveness and efficiency. Moreover, this paper proposes a new fitness function to solve the problem of class imbalance.

Goals

The overall goal of this paper is to enhance the performance of GP in classification with high-dimensional imbalanced data, in terms of enhancing the accuracies of the majority class and the minority class, as well as saving training time. This goal is composed of the following three sub-goals.

(1) Develop a strategy to reuse previous good GP trees,
(2) Develop a new fitness function to address the problem of class imbalance, and

(3) Investigate whether the proposed method can achieve significantly better or similar performance in classification with high-dimensional imbalanced data, compared with other existing classification algorithms.

This paper is organized as follows. Section 2 introduces some background knowledge related to GP and high-dimensional imbalanced classification. After that, the proposed method is introduced in Sect. 3, and experiment design is introduced in Sect. 4. In Sect. 5, we report the experimental results of the proposed method and baseline methods, and further analyse and discuss results. Conclusions are drawn in Sect. 6.

2 Background

2.1 High-dimensional imbalanced classification

Yin and Gai (2015) tested the joint effect of class imbalance and high dimensionality on C4.5 by investigating two types of feature selection approaches (i.e. wrapper and filter approaches) and data sampling methods (i.e. oversampling and undersampling) on twelve datasets with different dimensions and class imbalance ratios. Yin et al. (2013) provided a feature selection method based on class decomposition, which divides the majority class into several relatively smaller subclasses with a relatively uniform size, based on which feature selection is performed on the decomposed data.

Yang et al. (2009) developed a hybrid system that combines a particle swarm optimization (PSO) algorithm with multiple classifiers and evaluation metrics for evaluation fusion. PSO was used as a sample selection strategy, and samples from the majority class were ranked using multiple objectives or criteria and then combined with the minority class to form a balanced dataset (Yang et al. 2009). Yang et al. (2014) discussed the applications of sample subset optimization (SSO) techniques to address the problem of class imbalance with ensemble learning. To address the problem of class imbalance, the SSO technique was employed as an undersampling technique to identify a subset of the highly discriminative samples in the majority class.

Aydogan et al. (2019) proposed a new cost-sensitive classification method, called the cost-based rough PSO (CBR-PSO), for classification with high-dimensional imbalanced data. CBR-PSO, based on PSO and rough set theory, can simultaneously perform feature selection and classification, and consider the different misclassification costs. Liu et al. (2018) developed a cost-sensitive principal component analysis for feature extraction, and a variant of PSO, called chaos PSO, was applied to the parameter optimization.

2.2 Tree-based genetic programming

In GP, the individuals are often structured in terms of trees, where nodes of a tree are chosen from a function set and a terminal set. A function set is a set of all possible operators or functions, e.g. arithmetic operators or mathematics functions. A terminal set is constituted by the possible arguments for internal nodes. The size of a GP individual is usually limited by a maximum depth which is the longest path from the root to a leaf node. GP (Poli et al. 2008) has the following steps:

(1) Initialization: randomly generate an initial population of individuals (or called trees or programs).
(2) Iteratively perform the following steps until the stopping criterion is satisfied:
  (2.1) *Evaluation*: each individual is evaluated by a predefined fitness function.
  (2.2) *Selection*: good individuals are selected from the population based on their fitness values.
  (2.3) *Evolution*: new individuals are created by the genetic operators (e.g. reproduction, mutation and crossover) with specific probabilities.
(3) Return best individuals.

When GP is used to evolve classifiers, a GP tree is often seen as a classifier, which can be translated into a mathematical expression. The output values of this mathematical expression are used to classify instances. If an output value of a program taking an instance as an input is greater than or equal to a threshold, this instance is classified into the minority class, otherwise it is classified into the majority class.

2.3 GP for imbalanced classification

In GP, the methods to solve the problem of class imbalance have two groups, at the data level and at the algorithmic level.

At the data level, in addition to traditional sampling methods (e.g. undersampling and oversampling), subset selection methods can also be used as the sampling methods to solve the problem of class imbalance. Subset selection methods mainly include random subset selection (RSS), dynamic subset section (DSS) and historical subset selection (HSS) (Gathercole and Ross 1994). These methods could select some instances from the majority class, to ensure its number to be the same or roughly the same as that of the minority class. The selected instances from the majority class are combined with the minority class for evaluations at each generation. The main difference of three methods lies in the methodology to select a subset of instances from a training set during each generation. For RSS, each instance is randomly selected, while DSS tends to choose instances which are often misclassified, or have not been selected for several previous generations (Curry et al. 2007; Gathercole and Ross 1994). HSS selects instances based on the classification difficulty of each instance, determined by how many times they are misclassified by the best GP program in each generation (Gathercole and Ross 1994). Song et al. (2003) designed a two-layer subset selection sampling approach for linear GP, where the first layer is based on RSS and then instances in the second layer are sampled by DSS. Curry et al. (2007) proposed a family of hierarchical DSS algorithms, such as cascaded RSS-DSS and balanced-block DSS, for large datasets.

At the algorithmic level, there are three main methods to solve the problem of class imbalance in GP, including cost-sensitive learning, development of fitness functions, and EMO. Li et al. (2005) showed how cost-sensitive learning can be employed by grammar-guided GP in imbalanced classification. The cost information in a cost matrix is directly embedded in a fitness function. However, the cost matrix is often problem-specific and manually-designed. In many real-world applications, cost information is unknown or unavailable. In addition to cost-sensitive learning, for addressing the problem of class imbalance, new fitness functions have been proposed for GP, without a requirement to provide cost information (Bhowan et al. 2012, 2010; Patterson and Zhang 2007).

By using EMO to develop MOGP for imbalanced classification, the accuracies of the minority class and majority class are often seen as two potentially conflicting objectives. Bhowan et al. (2011b) proposed a new MOGP method, based on non-dominated sorting genetic algorithm II (NSGA-II), employing the negative correlation learning-based (NCL) measure as a diversity measure. However, NCL causes a substantial computational cost. Bhowan et al. (2011a) developed a new evolutionary-based pruning method to find groups of highly cooperative individuals that can improve the accuracy on the minority class. Bhowan et al. (2013) evaluated the effectiveness of two EMO algorithms, i.e. strength Pareto evolutionary algorithm 2 (SPEA2) and NSGA-II, and investigated how the diversity of solutions is encouraged. Bhowan et al. (2014) further designed a two-step MOGP approach. In the first step, a MOGP method, based on SPEA2, was developed to form ensembles, and in the second step, an ensemble selection approach was proposed to reuse GP trees to automatically choose the best classifiers or the combination of classifiers in the ensemble. However, MOGP methods are often time-consuming than single-objective GP methods.

Tran et al. (2016) investigated the performance of GP for feature construction and feature selection in classification with high-dimensional data. Tran et al. (2017) further proposed a cluster-based GP for feature construction, where feature clustering is used to group similar features and the best feature is chosen from each feature cluster to narrow the search space. However, Tran et al. (2016, 2017) employed the balanced accuracy that equally weighs different classes, as a fitness function to address the problem of class imbalance. As mentioned previously, in many cases, the majority class and the minority class are not equally important.

In summary, sampling methods and cost-sensitive learning methods are often used to address the problem of class imbalance. As mentioned previously, sampling methods and cost-sensitive learning have their own disadvantages in addressing the problem of class imbalance. In GP, the use of a fitness function to address the issue of class imbalance is an easy but effective solution, because it does not require to provide the cost information and it is often time-efficient than MOGP methods. Therefore, this paper designs a new fitness function to address the problem of class imbalance, and designs a reuse mechanism to further improve the efficiency.

3 The proposed method

In this section, we will introduce a new GP method, called Genetic Programming with a New Fitness Function and Reuse Mechanism (GPFRM).

When using GP to evolve classifiers, all features in a dataset are fed to the algorithm as terminals. However, for high-dimensional datasets, the search space is large. Based on granular computing, Pei et al. (2018) proposed an approach to hierarchically linking features for GP-based classification to increase the classification performance and save training time. Since GP has the built-in capability to automatically select informative features, the hierarchical feature structure is developed when building a multi-classifier system.

All features are randomly divided into five feature sets (denoted as $F1$, $F2$, $F3$, $F4$ and $F5$) with a roughly same size. The first GP process with a small population and a small number of generations is employed, feeding F1 as terminals, to evolve classifiers. After this evolutionary process, the features selected by good trees are saved as the good features $F1^*$, which is expected to have the similar discrimination ability as its original set $F1$. Those good features $F1^*$ are combined with $F2$ as the terminals in the second GP process. The similar processes continue until the fifth GP process using $F4^*$ combined with $F5$ as the terminals finishes to evolve the fifth classifier.

However, some good GP trees that are evolved after the first evolutionary process, would carry useful information. These trees are worth being reused by the following GP process in initialization to further enhance the effectiveness and efficiency.

3.1 Reusing previous trees

In this subsection, we propose a new reuse mechanism to reuse previously evolved good GP individuals since the sec-

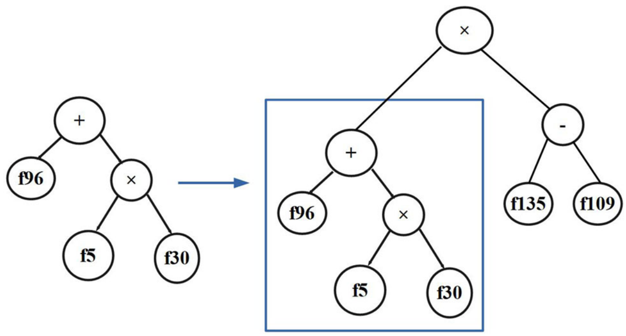

Fig. 1 Reusing trees in initialization

ond evolutionary process. In Fig. 1, we show an example to intuitively explain how these trees are reused. The first tree on the left is a good GP tree that was evolved previously, which is later reused as a terminal by a next GP process, shown as the tree on the right in Fig. 1.

For the first GP process, ramped half-and-half is employed for initialization, feeding the first feature set $F1$ as terminals. After this evolutionary process, the top 1% trees (based on their fitness values) are reused by the next GP process. This process starts at the second GP process, and finishes at the fifth GP process. Note that, a GP process (after the first GP process) only reuses top 1% good trees from an earlier GP process. For example, for the third GP process, it only reuses good trees from the second GP process.

3.2 A new fitness function

As introduced previously, a fitness function can be used to solve the issue of class imbalance in GP. The weighted average classification accuracy (Ave) is a commonly-used fitness function in GP for classification with imbalanced data. However, when Ave is used as a fitness function, it is important and necessary to provide a weight, but determination of this weight is problem-specific. Usually, a weight $W = 0.5$ is often used to equally treat minority class and majority class. However, in some real-world applications, e.g. medical diagnosis, the minority class and majority class are often not equally important.

Area under curve (AUC) is an important measure in imbalanced classification, which can also be used as a fitness function of GP (Bhowan et al. 2012). In general, GP with AUC as a fitness function achieves a good performance but consumes very long training time in fitness evaluations. This is because AUC needs to build the receiver operating characteristic (ROC) curve that requires the re-evaluation of true positive rate and false positive rate many times, based on multiple thresholds. Full AUC (Auc_F) is defined as follows (Bhowan et al. 2012):

$$\mathrm{Auc}_F = \sum_{i=1}^{N-1} \frac{1}{2} * (\mathrm{FPR}_{i+1} - \mathrm{FPR}_i)(\mathrm{TPR}_{i+1} + \mathrm{TPR}_i) \quad (1)$$

where N is the number of thresholds used in the ROC curve (the greater the number is, the better the AUC approximation is), and TPR_i and FPR_i are the true positive rate and false positive rate at the ith threshold.

Wilcoxon–Mann–Whitney ($\mathrm{Auc_w}$) provides a direct estimator for AUC metric, which is defined as (Bhowan et al. 2012):

$$\mathrm{Auc_w} = \frac{\sum_{i \in \mathrm{Min}} \sum_{j \in \mathrm{Maj}} I_{\mathrm{wmw}}(P_i, P_j)}{|\mathrm{Min}| * |\mathrm{Maj}|} \quad (2)$$

where $I_{\mathrm{wmw}}(P_i, P_j) = \begin{cases} 1, & P_i > P_j \text{ and } P_i \geq 0 \\ 0, & \text{otherwise} \end{cases}$

P_i and P_j represent the output values of a program P taking instance i from the minority class (Min) and instance j from the majority class (Maj) as inputs. |Min| and |Maj| indicate a number of instances in the minority class and the majority class, respectively.

GP using $\mathrm{Auc_w}$ as a fitness function is also very time-consuming due to $|\mathrm{Min}| * |\mathrm{Maj}|$ times pairwise comparison in fitness evaluations. In order to save training time, this paper proposes a new fitness function, which is seen as an AUC approximation measure.

The new fitness function is constituted by two components. The first component is to approximate $\mathrm{Auc_w}$, ensuring the outputs of a program taking instances from the minority class being larger than outputs of this program taking instances from the majority class.

The second component is used to encourage the separability of a program (i.e. how far a program is able to separate two classes, based on program outputs). Separability of a program is also important since it could further encourage outputs of a program for two classes separated with each other. Figure 2 explains the importance of the separability of a program. In Fig. 2, if the separability is not considered, program 1 and program 2 have the same fitness value, but in fact, program 2 is preferred to program 1. In this paper, the correlation radio (Fisher 1992) is employed to evaluate the separability of a program. The outputs of the correlation ratio are in the range of [0,1], where 0 indicates the worst separability and 1 indicates the best separability.

The new fitness function is defined as following:

$$\mathrm{Min\_Corr} = \frac{\sum_{i \in \mathrm{Min}} I(P_i, t)}{|\mathrm{Min}|} + \sqrt{\frac{\sum_{c=1}^{K} N_c(\mu_c - \overline{\mu})^2}{\sum_{c=1}^{K} \sum_{i=1}^{N_c} (P_{ci} - \overline{\mu})^2}} \quad (3)$$

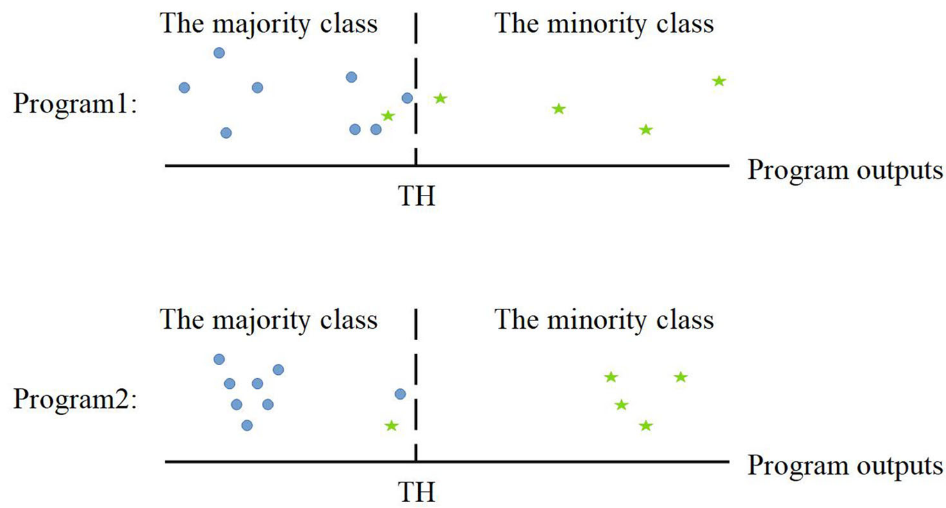

Fig. 2 Importance of separability

where $I(P_i, t) = \begin{cases} 1, & P_i > t \text{ and } P_i \geq 0 \\ 0, & \text{otherwise} \end{cases}$

In $I(P_i, t)$, P_i indicates an output value of a program P taking instance i from the minority class as an input, t is a maximum output value of this program taking instances from the majority class. In the second component, K is the number of classes ($K = 2$ for binary classification), N_c means the number of instances in class c, $\mu_c = \frac{\sum_{i=1}^{N_c} P_{ci}}{N_c}$, $\overline{\mu} = \frac{\sum_{c=1}^{K} N_c \mu_c}{\sum_{c=1}^{K} N_c}$, P_{ci} represents an output value of a program P taking instance i in class c.

In Eq. (3), the first component only needs to do |Min| times pairwise comparisons, rather than $|\text{Min}| * |\text{Maj}|$ times pairwise comparisons like Auc_w. If an output value of a program taking an instance from the minority class is larger than the maximum output value of this program taking instances from the majority class, it makes a correct decision. Therefore, it is expected that GP using the new fitness function is time-efficient than GP using Auc_w.

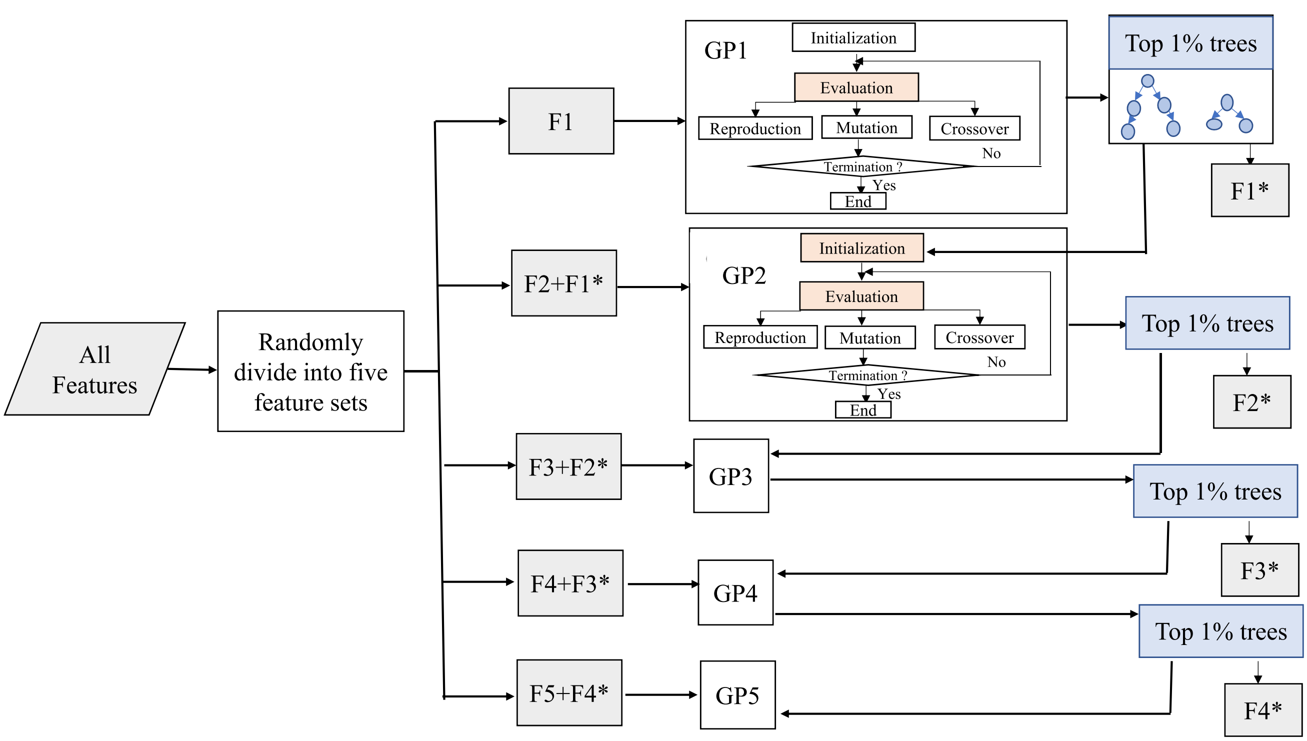

Fig. 3 Overall design

Table 1 Dataset descriptions

Dataset	#Features	#Instances	Majority class (%)	Minority class (%)	IR
Armstrong-2002-v1	1081	72	66	34	2:1
Golub_1990	1868	72	66	34	2:1
Colon	2000	62	65	35	2:1
Leukemia	7129	72	65	35	2:1
DLBCL	5469	77	75	25	3:1
gordon-2002	1626	181	83	17	5:1
Yeoh-2002-v1	2526	248	83	17	5:1
su-2001	1571	174	85	15	6:1
tomlins-2006-v1	2316	104	88	12	8:1
Lung	12,600	156	89	11	8:1

Algorithm 1

Input: The training set and the test set
Output: Accuracies

$q \Leftarrow 1$;
The whole feature set is randomly divided into 5 feature subsets, i.e. F1, F2, F3, F4 and F5;
$F1$ is fed to the GP process;
Obtained a good feature set F_1^* is appended into next feature set F2;
Save the top 1% good GP individuals;
$q \Leftarrow 2$;
while $q \leq 5$ **do**
  Features in a feature set F_q are fed to the next GP process;
  Good GP individuals evolved by an earlier GP process are also fed to the next GP process as terminals;
  Obtained a good feature set F_q^* is appended into next feature set F_{q+1};
  Save top 1% good GP individuals;
  $q \Leftarrow q + 1$;
end while
The best individuals from each GP process are chosen as classifiers that vote for a final decision to classify each instance in a test set;
Calculate accuracies on a test set

3.3 Overall design

Algorithm 1 shows the pseudo-code of the proposed method. The overall design of the training process is shown in Fig. 3.

All features in the whole feature space are randomly divided into five feature subsets (i.e. $F1$, $F2$, $F3$, $F4$ and $F5$), and each subset has roughly the same size. GP1 with a small population and a small number of generations is employed to evolve classifiers. $F1$ is fed to GP1 as terminals, using Ramped half-and-half for initialization. The proposed fitness function is used to evaluate individuals. After this evolutionary process, top 1% good GP trees and features selected by these trees (denoted as $F1^*$) are saved. These good GP trees are reused as a part of the terminal set in the initialization of GP2, and those selected features ($F1^*$) combined with $F2$ are fed to GP2. The similar processes continue until GP5 that uses $F4^*$ combined with $F5$ as the terminals finishes to evolve the fifth classifier. The best individual from each GP process is chosen as five classifiers, and soft voting is used to make a final decision in the test process.

4 Experiment design

4.1 Datasets

Table 1 describes the details of ten high-dimensional imbalanced datasets[1] (Zhu et al. 2007). These datastes have different dimensions and class imbalance ratios (IR). In the experiment, the datasets are divided into the training set (70%) and test set (30%), based on the stratified sampling to ensure the same class imbalance ratio in the training set and the test set. This paper focuses on binary classification tasks.

For many gene expression datasets, their imbalance ratio is not high (i.e. more than 5:1 or more). The main reason is that high-dimensional datasets usually do not have a sufficient number of instances for the majority class as well as the minority class. Accordingly, to examine the performance of the proposed method on the highly-imbalanced high-dimensional datasets, su-2001 (10 classes), tomlins-2006-v1 (5 classes) and Lung (5 classes) are changed into highly-imbalanced binary datasets. For su-2001, class 1 (26 instances) is used as the minority class, while the rest of classes are used together (148 instances) as the majority class (IR = 6:1). For tomlins-2006-v1, class 5 (12 instances) is used as the minority class, while other classes are combined together (92 instances) as the majority class (IR = 8:1). For Lung, the instances of two labels (class 1 used as the majority class and class 2 used as the minority class) are selected to form a highly-imbalanced binary dataset (IR = 8:1).

[1] http://www.gems-system.org; https://schlieplab.org/Static/Supplements/CompCancer/datasets.htm.

Table 2 Baseline fitness functions

Fitness functions
1: $\text{Acc} = \frac{\text{TP}+\text{TN}}{\text{TP}+\text{TN}+\text{FP}+\text{FN}}$
2: $\text{Ave} = W * \frac{\text{TP}}{\text{TP}+\text{FN}} + (1-W) * \frac{\text{TN}}{\text{TN}+\text{FP}}$ $(W=0.5)$
3: $G\_\text{Mean} = \sqrt{\frac{\text{TP}}{\text{TP}+\text{FN}} * \frac{\text{TN}}{\text{TN}+\text{FP}}}$
4: $\text{Amse} = \frac{1}{K}\Sigma_{c=1}^{K}(1 - \frac{\Sigma_{i=1}^{N_c}(\text{sig}(P_{c_i})-T_c)^2}{N_c*2})$, where $\text{sig}(x) = \frac{2}{1+\text{e}^{-x}} - 1$
$\text{Auc}_\text{w} = \frac{\sum_{i\in\text{Min}}\sum_{j\in\text{Maj}} I_{\text{wmw}}(P_i,P_j)}{\lvert\text{Min}\rvert*\lvert\text{Maj}\rvert}$, where $I_{\text{wmw}}(P_i,P_j) = \begin{cases} 1, & P_i > P_j \text{ and } P_i \geq 0 \\ 0, & \text{otherwise} \end{cases}$

In Acc, Ave and $G_$Mean, TP is true positive, FP is false positive, TN is true negative and FN is false negative

In Amse, T_c values are -0.5 and 0.5 for majority class and minority class respectively. K is the number of classes; N_c is the number of instances in the class c; P_{ci} is a program output value for an instance i in class c

4.2 Baseline methods

The proposed method (GPFRM) uses Algorithm 1 to evolve classifiers, and each GP process uses Min_Corr as a fitness function. To examine its effectiveness, the proposed method is compared with GP_{Acc} (i.e. GP using overall classification accuracy Acc as a fitness function), GP_{Ave} (i.e. GP using weighted-average classification accuracy Ave as a fitness function, $W = 0.5$), $\text{GP}_{\text{Min\_Corr}}$ (i.e. GP using Min_Corr as a fitness function), and GP_GrC (Pei et al. 2018).

The population size of GP_{Acc}, GP_{Ave}, and $\text{GP}_{\text{Min\_Corr}}$ is 1024 for 50 generations. Because GPFRM employs five GP processes in total, for each GP process, the population size is 256 for 40 generations, so that the total number of evaluations (i.e. 256 * 40 * 5) is similar to that of other GP methods (i.e. 1024 * 50) for a relatively-fair comparison. GP_GrC is a method based on the multiple GP processes, where Ave ($W = 0.5$) is used as a fitness function. In GP_GrC, the population size is 256 for 33 generations for six GP processes. To ensure a relatively-fair comparison, GP_GrC uses the same function set and maximum tree depth.

To further investigate the proposed fitness function, the proposed fitness function is compared with Acc, Ave, geometric mean ($G_$Mean), average mean squared error mse (Amse), and an AUC measure WMW (Auc_w). Acc and Ave are the most commonly-used fitness functions in GP for classification. $G_$Mean is similar to Ave, but it does not need a weight. Amse utilizes the magnitude of program outputs to the targets (Bhowan et al. 2012). Auc_w provides a direct estimator for AUC metric. The definition of these fitness functions is listed in Table 2.

GPFRM is also compared with other non-GP classification algorithms from machine learning, including 1-nearest neighbours (1NN), random forest (RF), gradient boosting decision tree (GBDT), and Naive Bayes (NB). In addition to standard versions of classification algorithms, sampling methods, including SMOTE (Chawla et al. 2002), adaptive synthetic sampling approach (ADASYN) (He et al. 2008), borderline-SMOTE (Han et al. (2005), SMOTE-Tomek (Batista et al. 2004) and SMOTEENN (Batista et al. 2004), are used individually to solve the problem of class imbalance for 1NN, RF, GBDT and NB.

SMOTE and ADASYN are the most commonly used oversampling methods for imbalanced classification in machine learning. Borderline-SMOTE is an improved version of SMOTE, which has two types, i.e. Borderline-SMOTE1 and Borderline-SMOTE2. SMOTETomek and SMOTEENN (Batista et al. 2004) are hybrid sampling methods for imbalanced learning. In addition, undersampling methods are not used because the number of instances in these datasets is small. For these datasets, the misclassification cost information is not available, so the proposed method is not compared with cost-sensitive learning methods.

4.3 Parameter settings

Table 3 shows the parameter settings of GP methods (with different fitness functions to address the problem of class imbalance) and the proposed GP method. The function set includes four basic arithmetic functions ($+$, $-$, $\times$, and protected division $\div$), and a conditional operator If function. Note that protected division returns zero when dividing by zero. For the If function, it has three arguments. If the first argument is negative, the second argument is returned, otherwise it returns the third argument.

5 Results and discussion

5.1 Results analysis

GP_{Acc}, GP_{Ave}, $\text{GP}_{\text{Min\_Corr}}$, GP_GrC and GPFRM have been independently run 30 times with different 30 random seeds (to ensure a fair comparison, different GP methods must use the same group of random seeds). The results (including accuracy on the majority class, accuracy on the minority class, overall classification accuracy and $G_$Mean) are reported in Table 4. Wilcoxon statistical significance test is conducted to compare $G_$Mean results of the proposed method with a baseline method, with the significance level of 0.05. In Table 4, "+", "=" and "$-$" are used to show that the proposed method is significantly better, similar, and significantly worse than a compared method.

According to Table 4, GPFRM achieves significantly better classification performance ($G_$Mean) than GP_{Acc} and GP_{Ave} in 18 out of the 20 cases. Moreover, GPFRM achieves better classification performance (accuracies of the majority class and the minority class, and G_Mean) than $\mathrm{GP}_{\mathrm{Min\_Corr}}$ on nine datasets (ten datasets in total). Furthermore, by GPFRM, the standard deviations of G_Mean results are smaller than that of $\mathrm{GP}_{\mathrm{Min\_Corr}}$ on these datasets, which may show that the stability of GPFRM is better than $\mathrm{GP}_{\mathrm{Min\_Corr}}$. Averaged training time of GPFRM is much faster than $\mathrm{GP}_{\mathrm{Min\_Corr}}$ on all datasets, only consuming 15.57%–29.34% of training time that is consumed by $\mathrm{GP}_{\mathrm{Min\_Corr}}$. Colon and DLBCL are two difficult datasets, even though the imbalance ratios of the two datasets are not high. By using the proposed method, the classification performance (G_Mean) increases by 12.05% on Colon and 14.28% on DLBCL, compared with $\mathrm{GP}_{\mathrm{Min\_Corr}}$. On Yeoh-2002-v1, the performance of GPFRM is slightly decreased, compared with $\mathrm{GP}_{\mathrm{Min\_Corr}}$. A possible reason is that this dataset has a relatively sufficient number of instances and the ratio of the number of features to the number of instances is not very high. Dividing the whole features and using smaller populations and generations to train each GP process may cause some GP classifiers being underfitting in GPFRM. Therefore, it is likely for several weak GP classifiers against a good GP classifier, thereby making a wrong decision. By comparing GPFRM with GP_GrC, the classification performance (G_Mean) of the proposed method is significantly enhanced (significantly better performance in 9 out of the 10 cases and similar performance in 1 out of the 10 cases). It is also noticed that the performance (G_Mean) of $\mathrm{GP}_{\mathrm{Ave}}$ is also higher than GP_GrC on Yeoh-2002-v1. It seems not necessary to employ multiple GP processes to develop a multi-classifier system if the ratio of a number of features to instances is not high. In addition, on each dataset, the best G_Mean result of the proposed method is often at least similar to other GP methods. More importantly, GPFRM often achieves the narrowest gap between the best result and the averaged result (G_Mean).

5.2 Further investigation on the new fitness function

For further investigation, in this subsection, the new fitness function is investigated and compared with the baseline fitness functions in Table 2. In Table 5, we report the accuracy on the majority class, accuracy on the minority class, G_Mean and AUC results of GP with different fitness functions on the test sets. Wilcoxon statistical significance test is also conducted, with the significance level of 0.05. The results of the significance test are also reported in Table 5, where "+", "=" and "−" are used to show that the new fitness function is significantly better, similar, and significantly worse than a compared fitness function. Training time of GP using different fitness functions are reported in Table 6.

According to Table 5, GP with the proposed fitness function (i.e. $\mathrm{GP}_{\mathrm{Min\_Corr}}$) achieves the best G_Mean on four datasets, and the best AUC on three datasets. GP using the new fitness function achieves significantly better or similar performance in 99 out of the 100 cases. For highly-imbalanced datasets, such as Yeoh-2002-v1 and Lung, $\mathrm{GP}_{\mathrm{Min\_Corr}}$ often achieves significantly better performance (G_Mean and AUC) than GP using other fitness functions only except for $\mathrm{Auc}_{\mathrm{w}}$ (similar performance). For slightly-imbalanced datasets, such as Armstrong-2002-v1, the performance of GP using this fitness functions is also improved. The best AUC and G_Mean results obtained by $\mathrm{GP}_{\mathrm{Min\_Corr}}$ are at least as good as others on all datasets.

Moreover, the gap between the averaged accuracy on the majority class and averaged accuracy on the minority class is obviously narrowed, compared with other baseline fitness functions except for $\mathrm{Auc}_{\mathrm{w}}$. For example, on dataset Yeoh-2002-v1, the gap between the accuracies of the majority class and minority class is 2.02%, which is much narrower than results achieved by GP using Acc, Ave, G_Mean or Amse as a fitness function.

Table 3 Parameter settings

Parameters	Standard GP (with different fitness functions)	Each GP process of the proposed method
Population size	1024	256
Generations	50	40
Initial population	Ramped half-and-half	Ramped half-and-half
Maximum tree depth	10	10
Mutation rate	0.2	0.2
Crossover rate	0.8	0.8
Elitism	1	1
Selection method	Tournament (size = 6)	Tournament (size = 6)
Function set	$+, -, \times, \div, If$	$+, -, \times, \div, If$
Terminal set	All features, a random constant	F_q and F^*_{q-1}, a random constant

Table 4 Results on the test sets (GPMFS Versus other GP methods)

Dataset	Method	Accuracy on the majority class (%)		Accuracy on the minority class (%)		Overall cccuracy (%)		*G*_Mean (%)			Training time (s)
		Best	Mean ± Std	Best	Mean ± Std	Best	Mean ± Std	Best	Mean ± Std	ST	Mean
Armstrong-2002-v1	GP_{Acc}	100	92.89 ± 7.69	100	82.38 ± 13.14	100	89.55 ± 7.34	100	87.17±8.67	+	116.17
	GP_{Ave}	100	91.11 ± 9.32	100	85.71 ± 12.78	100	89.39 ± 5.91	100	87.84±6.52	+	114.86
	GP_{Min_Corr}	100	94.67 ± 4	100	88.57 ± 8.57	100	92.73 ± 5.45	100	91.53±6.38	+	143.6
	GP_GrC	100	92.67 ± 7.95	100	91.43 ± 7.91	100	92.27 ± 5.52	100	91.83 ±5.16	+	41.65
	GPFRM	100	99.11 ± 2.27	100	98.1 ± 4.86	100	98.79 ± 3.09	100	**98.59 ± 3.59**		42.13
Golub_1990	GP_{Acc}	100	93.33 ± 7.13	100	78.75 ± 23.08	100	88.03 ± 9.42	100	84.54 ± 14.10	+	157.57
	GP_{Ave}	100	90.71 ± 11.69	100	82.92 ± 19.23	100	87.88 ± 11.03	100	85.56 ± 13.06	+	158.77
	GP_{Min_Corr}	100	96.43 ± 5.76	100	93.75 ± 10.08	100	95.45 ± 7.33	100	95.05 ± 8.01	=	252.73
	GP_GrC	100	95.71 ±5.71	100	85.42 ± 12.94	100	91.97 ± 5.47	100	90.07 ± 7.17	+	53.36
	GPFRM	100	98.57 ± 2.86	100	97.5 ± 5	100	98.18 ± 3.64	100	**98.03 ±3.94**		58.64
Colon	GP_{Acc}	100	86.67 ± 10	100	52.38 ± 17.82	86.6.	66.07 ± 11.62	88.64	66.07 ± 11.60	+	176.74
	GP_{Ave}	100	87.22 ± 8.53	71.43	57.62 ± 13.54	80.91	70.33 ± 9.83	80.90	70.33 ± 9.83	+	177.08
	GP_{Min_Corr}	91.67	78.06 ± 6.97	85.71	62.38 ± 11.94	89.47	72.28 ± 8.8	88.64	69.69 ± 9.8	+	255.65
	GP_GrC	100	89.72 ± 5.13	85.71	64.29 ± 11.52	89.47	80.35 ± 5.07	88.64	75.57 ± 7.16	+	50.30
	GPFRM	91.67	86.67 ± 4.61	85.71	77.14 ± 7.91	89.47	83.16 ± 5.83	88.64	**81.74 ± 6.38**		50.80
Leukemia	GP_{Acc}	100	85.71 ± 10.59	100	70.42 ± 17.82	95.45	80.15 ± 9.64	96.36	76.79 ± 12.31	+	976.92
	GP_{Ave}	100	89.05 ± 8.79	87.5	71.67 ± 14.04	95.45	82.73 ± 6.58	93.51	79.25 ± 8.45	+	975.7
	GP_{Min_Corr}	100	89.52 ± 5.13	100	81.67 ± 8.98	100	86.67 ± 6.53	100	85.47± 7.16	=	793.88
	GP_GrC	100	88.57 ± 9.69	87.5	75.83 ± 12.47	90.91	83.94 ± 6.61	90.14	81.43 ± 7.37	+	121.7
	GPFRM	92.86	91.43 ± 2.86	87.5	84.58 ± 6.19	90.91	88.94 ± 4.01	90.14	**87.91 ± 4.60**		157.34
DLBCL	GP_{Acc}	100	89.44 ± 6.78	100	59.44 ± 24.6	95.83	81.94 ± 8.83	97.18	69.60 ± 22.63	+	733.43
	GP_{Ave}	100	80.56 ± 10.71	100	57.22 ± 19.09	91.67	74.72 ± 9.12	91.29	66.67 ± 12.40	+	740.03
	GP_{Min_Corr}	100	87.04 ± 7.08	100	61.11 ± 21.23	100	80.56 ± 10.61	100	71.46 ± 19.05	+	670.87
	GP_GrC	100	86.85 ± 7.1	83.33	71.67 ± 9.77	95.83	83.06 ± 5.79	91.29	78.63 ± 6.25	+	116.4
	GPFRM	100	93.52 ± 3.82	100	78.89 ± 12.12	100	89.86 ± 5.76	100	**85.74 ± 8.28**		153.92
gordon-2002	GP_{Acc}	100	98.84 ± 1.46	100	88.52 ± 12.0	100	97.15 ± 2.14	100	93.29±6.56	+	324.87
	GP_{Ave}	100	98.99±1.56	100	88.52 ±12.98	100	97.27±2.73	100	93.32±7.58	+	321.69
	GP_{Min_Corr}	100	98.41±1.58	100	91.85±8.08	100	97.33±2.64	100	95.01±5.02	+	724.74
	GP_GrC	100	99.71±0.74	100	86.67±11.62	100	97.58± 1.84	100	92.73+6.3	+	107.84
	GPFRM	100	99.86±0.54	100	98.89±3.33	100	99.7±0.95	100	**99.36±1.93**		186.7
Yeoh-2002-v1	GP_{Acc}	100	94.68 ± 3.68	76.92	41.28 ± 20.3	92	85.42 ± 4.43	85.56	59.93±17.62	+	779.75
	GP_{Ave}	100	90.43 ± 12.32	100	73.85 ± 20.7	98.67	87.56 ± 10.4	99.19	80.39±13.68	+	773.26

Table 4 continued

Dataset	Method	Accuracy on the majority class (%)		Accuracy on the minority class (%)		Overall cccuracy (%)		G_Mean (%)			Training time (s)
		Best	Mean ± Std	Best	Mean ± Std	Best	Mean ± Std	Best	Mean ± Std	ST	Mean
	GP_{Min_Corr}	100	99.46 ± 1.05	100	97.44 ± 5	100	99.11 ± 1.73	100	**98.42±3.10**	=	703.08
	GP_GrC	96.77	77.04 ± 14.69	100	73.85 ± 14.27	92	76.49 ± 11.81	91.31	74.43±9.98	+	149.46
	GPFRM	100	99.25 ±0.91	100	95.64 ±4.73	100	98.62 ± 1.52	100	97.41 ± 2.83		152.78
su-2001	GP_{Acc}	100	99.7 ± 0.95	100	100 ± 0	100	99.75 ± 0.81	100	99.85 ± 0.48	=	285.91
	GP_{Ave}	100	99.48 ± 1.24	100	99.58 ± 2.24	100	99.5 ± 1.08	100	99.52 ± 1.27	=	288.31
	GP_{Min_Corr}	100	99.85 ± 0.55	100	99.17 ± 3.12	100	99.75 ± 0.94	100	99.5 ± 1.87	=	762.61
	GP_GrC	100	99.93 ± 0.4	100	99.58 ± 2.24	100	99.87 ± 0.47	100	99.75+1.17	=	92.17
	GPFRM	100	100 ± 0	100	100 ± 0	100	100 ± 0	100	**100 ± 0**		185.42
tomlins-2006-v1	GP_{Acc}	100	85.95 ± 9.89	100	54.14 ± 27.45	96.88	81.98 ± 9.84	96.36	62.77 ± 27.61	+	292.52
	GP_{Ave}	100	78.1 ± 13.26	100	82.5 ± 20.56	96.88	78.65 ± 13.26	98.2	78.85 ± 12.68	+	293.08
	GP_{Min_Corr}	100	97.62 ± 2.5	100	83.33 ± 17.48	100	95.83 ± 4.37	100	89.77 ± 10.96	+	650.1
	GP_GrC	96.43	73.81 ± 15.11	100	85.0 ± 17.8	96.88	75.24 ± 13.01	98.2	78.0+11.92	+	69.71
	GPFRM	100	99.4 ± 1.33	100	95.83 ± 9.32	100	98.96 ± 2.33	100	**97.51 ± 5.57**		126.56
Lung	GP_{Acc}	100	95.56 ± 3.41	100	47.33 ± 32.03	100	90.43 ± 4.56	100	60.46 ± 29.43	+	3044.48
	GP_{Ave}	100	95.24 ± 3.79	100	54.67 ± 25.79	95.74	90.92 ± 4.46	97.59	68.41 ± 23.03	+	3048.62
	GP_{Min_Corr}	100	96.75 ± 2.08	100	72.67 ± 17.5	100	94.18 ± 3.72	100	83.28± 11.43	+	2639.07
	GP_GrC	100	97.06 ± 2.66	100	58.67 ± 18.57	100	92.98 ± 3.21	100	74.49 ± 12.24	+	423.19
	GPFRM	100	99.05 ± 1.32	100	92 ± 11.08	100	98.3 ± 2.36	100	**95.31 ± 6.55**		413.20

1. Std means standard deviation
2. Bold values indicate the best result achieved on each dataset

Table 5 Results of GP using different fitness functions on the test sets

Datasets	Fitness functions	Accuracy on the majority class (%)		Accuracy on the minority class (%)		G_Mean (%)		AUC (%)		ST
		Best	Mean ± Std	Best	Mean ± Std	Best	Mean ± Std	Best	Mean ± Std	(G_Mean, AUC)
Armstrong-2002-v1	GP_{Acc}	100	92.89 ± 7.69	100	82.38 ± 13.14	100	87.17 ± 8.67	100	91.02 ± 10.58	(+,+)
	GP_{Ave}	100	91.11 ± 9.32	100	85.71 ± 12.78	100	87.84 ± 6.52	100	94.48 ± 8.4	(+,=)
	GP_{G_Mean}	100	90.22 ± 10.99	100	81.9 ± 13.77	100	85.41 ± 9.31	100	92.13 ± 8.01	(+,+)
	GP_{Amse}	100	75.33 ± 23.61	100	90.48 ± 11.27	100	80.92 ± 15.14	100	90.17 ± 7.65	(+,+)
	GP_{Auc_w}	100	94.22 ± 3.33	100	87.62 ± 7.13	100	90.84 ± 5.30	100	94.46 ± 4.93	(=,=)
	$\mathbf{GP}_{Min\_Corr}$	100	94.67 ± 4	100	88.57 ± 8.57	100	**91.53 ± 6.38**	100	**96.63 ± 4.55**	
Golub_1990	GP_{Acc}	100	93.33 ± 7.13	100	78.75 ± 23.08	100	84.54 ± 14.10	100	90.36 ± 9.16	(+,+)
	GP_{Ave}	100	90.71 ± 11.69	100	82.92 ± 19.23	100	85.56 ± 13.06	100	91.93 ± 10.09	(+,+)
	GP_{G_Mean}	100	91.19 ± 8.59	100	76.67 ± 21.1	100	81.41 ± 17.23	100	88.99 ± 11.89	(+,+)
	GP_{Amse}	100	81.43 ± 13.12	100	75.83 ± 18.52	100	77.64±12.42	100	82.78 ± 11.62	(+,+)
	GP_{Auc_w}	100	97.62 ± 4.64	100	95.83 ± 8.12	100	**96.70 ± 6.43**	100	**98.42 ± 3.38**	(=,=)
	$\mathbf{GP}_{Min\_Corr}$	100	96.43 ± 5.76	100	93.75 ± 10.08	100	95.05 ± 8.01	100	97.23 ± 7.42	
Colon	GP_{Acc}	100	86.67 ± 10	100	52.38 ± 17.82	88.60	66.07 ± 11.60	90.48	70.99 ± 11.84	(=,+)
	GP_{Ave}	100	87.22 ± 8.53	71.43	57.62 ± 13.54	80.90	70.33 ± 9.83	91.67	75.52 ± 10.11	(=,=)
	GP_{G_Mean}	100	90.56 ± 7.37	85.71	46.67 ± 20.18	88.64	61.90 ± 20.45	92.86	71.51 ± 12.95	(=,=)
	GP_{Amse}	100	86.94 ± 11.11	85.71	54.29 ± 14.94	84.52	67.73 ± 10.34	95.24	74.8 ± 10.76	(=,=)
	GP_{Auc_w}	91.67	80.83 ± 5.75	85.71	67.14 ± 9.86	88.64	**73.62 ± 8.06**	91.67	**78.97 ± 7.3**	(=,=)
	$\mathbf{GP}_{Min\_Corr}$	91.67	78.06 ± 6.97	85.71	62.38 ± 11.94	88.64	69.69 ± 9.8	95.24	74.76 ± 12.4	
Leukemia	GP_{Acc}	100	85.71 ± 10.59	100	70.42 ± 17.82	96.36	76.79 ± 12.31	99.11	88.04 ± 9.23	(+,=)
	GP_{Ave}	100	89.05 ± 8.79	87.5	71.67 ± 14.04	93.51	79.25 ± 8.45	98.21	88.79 ± 7.74	(+,=)
	GP_{G_Mean}	100	87.38 ± 8.39	100	64.17 ± 22.76	96.36	73.44 ± 15.62	100	81.79 ± 15.38	(+,+)
	GP_{Amse}	92.86	83.33 ± 9.64	100	72.08 ± 20.07	96.36	76.47 ± 12.01	100	81.73 ± 11.84	(+,+)
	GP_{Auc_w}	100	87.62 ± 6.88	100	78.33 ± 12.05	100	82.78 ± 9.66	100	86.28 ± 9.68	(=,+)
	$\mathbf{GP}_{Min\_Corr}$	100	89.52 ± 5.13	100	81.67 ± 8.98	100	**85.47 ± 7.16**	100	**90.09 ± 6.81**	
DLBCL	GP_{Acc}	100	89.44 ± 6.78	100	59.44 ± 24.6	97.18	69.60 ± 22.63	99.07	82.03 ± 11.93	(=,=)
	GP_{Ave}	100	80.56 ± 10.71	100	57.22 ± 19.09	91.29	66.67 ± 12.40	98.15	75.4 ± 15.67	(+,+)
	GP_{G_Mean}	100	89.07 ± 8.18	100	36.67 ± 21.69	88.19	54.56 ± 15.87	100	77.01 ± 15.75	(+,+)
	GP_{Amse}	100	80.56 ± 16.09	100	60 ± 24.57	97.18	66.76 ± 13.37	100	77.19 ± 13.18	(+,+)
	GP_{Auc_w}	100	87.96 ± 4.32	100	63.89 ± 12.97	100	**74.74 ± 9.5**	100	**85.54 ± 10.83**	(=,−)
	$\mathbf{GP}_{Min\_Corr}$	100	87.04 ± 7.08	100	61.11 ± 21.23	100	71.46 ± 19.05	100	81.14 ± 15.28	

Table 5 continued

Datasets	Fitness functions	Accuracy on the majority class (%)		Accuracy on the minority class (%)		G_Mean (%)		AUC (%)		ST
		Best	Mean ± Std	Best	Mean ± Std	Best	Mean ± Std	Best	Mean ± Std	(G_Mean, AUC)
gordon-2002	GP_{Acc}	100	98.84 ± 1.46	100	88.52 ± 12.0	100	93.29 ± 6.56	100	96.64 ± 5.62	(=,=)
	GP_{Ave}	100	98.99 ± 1.56	100	88.52 ± 12.98	100	93.32 ± 7.58	100	98.26 ± 2.81	(=,=)
	$GP_{G\text{\_Mean}}$	100	98.77 ± 2.22	100	88.89 ± 10.73	100	93.5 ± 5.98	100	98.38 ± 3.01	(=,=)
	GP_{Amse}	100	96.96 ± 3.05	100	81.85 ± 16.85	100	88.58 ± 10.3	100	96.49 ± 4.46	(+,=)
	GP_{Auc_w}	100	98.99 ± 1.34	100	94.81 ± 6.87	100	**96.83 ± 4.21**	100	**99.23 ± 2.06**	(=,=)
	$\mathbf{GP}_{\text{Min\_Corr}}$	100	98.41 ± 1.58	100	91.85 ± 8.08	100	95.01 ± 5.02	100	98.48 ± 2.35	
Yeoh-2002-v1	GP_{Acc}	100	94.68 ± 3.68	76.92	41.28 ± 20.3	85.56	59.93 ± 17.62	93.18	70.02 ± 12.43	(+,+)
	GP_{Ave}	100	90.43 ± 12.32	100	73.85 ± 20.7	99.19	80.39 ± 13.68	100	83.97 ± 11.91	(+,+)
	$GP_{G\text{\_Mean}}$	100	96.13 ± 2.11	84.62	39.49 ± 21.55	89.94	58.75 ± 18.46	95.78	66.33 ± 16.09	(+,+)
	GP_{Amse}	100	38.6 ± 29.26	100	68.46 ± 22.5	80.94	43.26 ± 16.98	92.06	63.79 ± 12.06	(+,+)
	GP_{Auc_w}	100	99.14 ± 1.43	100	95.9 ± 6.8	100	97.46 ± 4.23	100	**98.95 ± 2.32**	(=,=)
	$\mathbf{GP}_{\text{Min\_Corr}}$	100	99.46 ± 1.05	100	97.44 ± 5	100	**98.42 ± 3.10**	100	98.8 ± 3.35	
su-2001	GP_{Acc}	100	99.7 ± 0.95	100	100 ± 0	100	**99.85 ± 0.48**	100	99.98 ± 0.1	(=,=)
	GP_{Ave}	100	99.48 ± 1.24	100	99.58 ± 2.24	100	99.52 ± 1.27	100	99.88 ± 0.47	(=,=)
	$GP_{G\text{\_Mean}}$	100	99.41 ± 1.14	100	99.58 ± 2.24	100	99.49 ± 1.24	100	**99.99 ± 0.05**	(=,=)
	GP_{Amse}	100	97.78 ± 3.67	100	97.5 ± 5.0	100	97.59 ± 3.37	100	99.11 ± 1.49	(=,=)
	GP_{Auc_w}	100	99.93 ± 0.4	100	99.58 ± 2.24	100	99.75 ± 1.35	100	99.99 ± 0.05	(=,=)
	$\mathbf{GP}_{\text{Min\_Corr}}$	100	99.85 ± 0.55	100	99.17 ± 3.12	100	99.5 ± 1.87	100	99.57 ± 2.24	
tomlins-2006-v1	GP_{Acc}	100	85.95 ± 9.89	100	54.14 ± 27.45	96.36	62.77 ± 27.61	100	74.93 ± 19.97	(+,+)
	GP_{Ave}	100	78.1 ± 13.26	100	82.5 ± 20.56	98.2	78.85 ± 12.68	100	88.87 ± 13.49	(+,+)
	$GP_{G\text{\_Mean}}$	100	74.88 ± 13.75	100	80.83 ± 20.09	96.36	76.76 ± 12.7	100	84.54 ± 14.55	(+,+)
	GP_{Amse}	96.43	74.52 ± 10.38	100	92.5 ± 14.65	94.49	82.51 ± 9.09	100	92.96 ± 8.04	(+,=)
	GP_{Auc_w}	100	95.95 ± 3.02	100	71.67 ± 21.15	100	82.07 ± 14.33	100	91.0 ± 9.75	(+,=)
	$\mathbf{GP}_{\text{Min\_Corr}}$	100	97.62 ± 2.5	100	83.33 ± 17.48	100	**89.77 ± 10.96**	100	**93.65 ± 9.97**	
Lung	GP_{Acc}	100	95.56 ± 3.41	100	47.33 ± 32.03	100	60.46 ± 29.43	100	78.92 ± 16.25	(+,+)
	GP_{Ave}	100	95.24 ± 3.79	100	54.67 ± 25.79	97.59	68.41 ± 23.03	100	82.08 ± 14.11	(+,+)
	$GP_{G\text{\_Mean}}$	100	96.59 ± 2.8	100	44 ± 28.47	98.80	57.19 ± 31.40	98.57	76.22 ± 18.77	(+,+)
	GP_{Amse}	100	91.59 ± 5.95	100	56.67 ± 25.86	100	69.95 ± 17.61	100	78.65 ± 18.75	(+,+)
	GP_{Auc_w}	100	97.54 ± 1.89	100	79.33 ± 15.9	100	**87.60 ± 17.63**	100	**92.57 ± 10.14**	(=,=)
	$\mathbf{GP}_{\text{Min\_Corr}}$	100	96.75 ± 2.08	100	72.67 ± 17.5	100	83.28 ± 11.43	100	90.1 ± 11.29	

Bold values indicate the best result achieved on each dataset

Table 6 Average training time of GP using different fitness functions (s)

Methods	Datasets									
	Armstrong-2002-v1	Golub_1990	Colon	Leukemia	DLBCL	gordon-2002	Yeoh-2002-v1	Su-2001	tomlins-2006-v1	Lung
GP_{Acc}	116.17	157.57	176.704	976.92	733.43	324.87	779.75	285.91	292.52	3044.48
$GP_{WAveAcc}$	114.86	158.77	177.08	975.7	740.03	321.69	773.26	288.31	293.08	3048.62
GP_{G_Mean}	114.88	158.31	174.21	979.29	731.45	323.48	767.4	289.37	296.92	3038.89
GP_{Amse}	146.24	226.35	203.33	793.34	638.3	734.51	717.74	636.12	655.62	2503.15
GP_{Auc_w}	1917.99	3089.78	2348.72	10,396.7	7845.03	21,978.57	24,174.23	18,100.27	7865	45,375.33
GP_{Min_Corr}	143.6	252.73	225.65	793.88	670.87	724.74	703.08	762.61	750.1	2639.07

Table 7 Comparison with other traditional classification methods using SMOTE or ADASYN (G_Mean%)

	1NN	1NN w.t. SMOTE	1NN w.t. ADASYN	RF	RF w.t. SMOTE	RF w.t. ADASYN	GBDT	GBDT w.t. SMOTE	GBDT w.t. ADASYN	NB	NB w.t. SMOTE	NB w.t. ADASYN	**GPFRM** Best	Mean
Armstrong	92.58+	96.61=	92.58+	88.66+	90.33+	90.33+	89.44+	89.44+	89.44+	92.58+	92.58+	92.58+	100	**98.59**
Golub	82.92+	82.92+	86.60+	81.40+	86.63+	86.74+	92.58+	92.58+	92.58+	61.24+	81.23+	81.23+	100	**98.03**
Colon	48.80+	65.47+	69.01+	54.22+	60.80+	60.81+	50.18+	56.75+	66.28+	40.82+	43.64+	43.64+	88.64	**81.74**
Leukemia	70.71=	76.76+	86.60=	74.03+	75.71+	74.03+	86.60=	86.60=	86.60=	**100**−	**100**−	**100**−	90.14	87.91
DLBCL	78.17+	78.17+	78.17+	61.83+	73.20+	71.18+	66.83+	73.20+	71.18+	76.98+	88.71 =	**91.29**−	100	85.74
gordon	94.49+	100 =	97.80=	89.00+	94.08+	93.55+	89.00+	90.30+	88.19+	88.19+	88.19+	88.19+	100	**99.36**
Yeoh	78.45=	87.01+	90.70 +	75.04+	75.04+	77.05+	96.08=	96.08=	96.08=	87.70+	85.56+	85.56+	100	**97.41**
su-2001	**100**=	**100**=	**100**=	98.77=	99.62=	99.01=	**100**=	**100**=	**100**=	**100**=	**100**=	**100**=	100	**100**
tomlins	**98.20** =	**98.20**=	**98.20**=	81.38 +	80.35+	82.25+	73.88+	80.01+	79.19+	86.60+	70.71+	70.71+	100	97.51
Lung	95.11=	95.11=	95.11=	77.71+	84.46+	80.90+	**100**−	**100**−	**100**−	76.53+	77.46+	77.46+	100	95.31
Summary	5+, 5=, 0−	5+, 5=, 0−	5+, 5=, 0−	9+, 1=, 0−	9+, 1=, 0−	9+, 1=, 0−	6+, 3=, 1−	6+, 3=, 1−	6+, 3=, 1−	8+, 1=, 1−	7+, 2=, 1−	7+, 1=, 2−		

Bold values indicate the best result achieved on each dataset

When datasets are imbalanced, Auc_w is a good fitness function, but GP using it as a fitness function needs to consume very long training time. $\text{GP}_{\text{Min\_Corr}}$ often achieves at least similar performance as $\text{GP}_{\text{Auc}_\text{w}}$, but its training time is much shorter than $\text{GP}_{\text{Auc}_\text{w}}$, according to Table 6. However, based on Table 6, GP using this new fitness function often consumes longer training time than GP using other fitness functions, e.g. Ave.

5.3 Comparison with other non-GP classification algorithms with sampling methods

Table 7 shows results of standard classification algorithms with and without standard oversampling methods (i.e. SMOTE and ADASYN). For further investigation, an improved version of SMOTE, i.e. borderline-SMOTE (including borderline-SMOTE1 and borderline SMOTE2), is used to solve the issue of class imbalance for standard classification algorithms, and results are reported in Table 8. The results of standard classification algorithms with hybrid sampling methods (i.e. SMOTETomek and SMOTEENN) are reported in Table 9. The proposed method (GPFRM) is compared with these classification methods (without and with sampling methods) by using Wilcoxon statistical significance test, with the significance level of 0.05 ("+", "=" and "−" are used to show that GPFRM is significantly better, similar, and significantly worse than a compared method).

In general, based on Tables 7, 8 and 9, GPFRM achieves significantly better or similar performance in 266 out of the 280 cases (significantly better performance in 186 cases and similar performance in 80 cases).

According to Table 7, GPFRM achieves significantly better performance than other classification algorithms in 81 out of the 120 cases, and achieves similar performance in 32 out of the 120 cases. On datasets with $IR \geq 5$, such as Yeoh-2002-v1 and tomlins-2006-v1, GPFRM achieves at least similar performance than other classification algorithms in 57 out of the 60 cases (significantly better performance in 35 cases, and similar performance in 22 cases, respectively).

Similarly, based on Table 8, GPFRM achieves at least similar performance in 77 out of the 80 cases, and by comparing with classification algorithms with hybird sampling methods, GPFRM achieves at least similar performance in 76 out of the 80 cases based on Table 9. In addition, for the best results achieved by GPFRM, they are often better than other classification algorithms.

Overall, by comparing with other GP methods and non-GP classification methods, the proposed GP method (GPFRM) achieves at least similar performance compared with other methods.

Table 8 Comparison with other traditional classification methods using borderline-SMOTE ($G\_Mean\%$)

	1NN w.t. B-SMOTE1	1NN w.t. B-SMOTE2	RF w.t. B-SMOTE1	RF w.t. B-SMOTE2	GBDT w.t. B-SMOTE1	GBDT w.t. B-SMOTE2	NB w.t. B-SMOTE1	NB w.t. B-SMOTE2	GPFRM Best	GPFRM Mean
Armstrong	96.61 =	93.09+	89.42+	92.14+	89.44+	89.44+	84.52+	**100**=	100	98.59
Golub	93.54+	93.54+	97.92=	81.20+	92.58+	92.58+	76.18+	80.18+	100	**98.03**
Colon	65.47+	69.01+	61.02+	61.10+	68.77+	61.05+	43.64+	43.64+	88.64	**81.74**
Leukemia	86.60=	86.60=	76.81+	75.65+	86.60=	83.45+	**100**−	93.54−	90.14	87.91
DLBCL	78.17+	66.67+	71.69+	77.16+	68.71+	88.72=	79.35=	**88.71**=	100	85.74
gordon	98.91=	95.55+	93.70+	92.49+	97.70=	91.29+	88.19+	88.19+	100	**99.36**
Yeoh	89.80+	69.56+	64.86+	73.77+	96.08=	96.08=	80.47+	90.49+	100	**97.41**
su-2001	**100**=	**100**=	98.87=	98.56=	**100**=	**100**=	**100**=	**100**=	100	**100**
tomlins	**98.20**=	90.63+	82.27+	91.37+	79.02+	97.99=	86.60+	70.71+	100	97.51
Lung	93.86=	91.28=	77.06+	83.02+	**100**−	91.29=	77.46+	77.46+	100	95.31
Summary	4+, 6=, 0−	7+, 3=, 0−	8+, 2=, 0−	9+, 1=, 0−	5+, 4=, 1−	5+, 5=, 0−	7+, 2=, 1−	6+, 3=, 1−		

B-SMOTE1 means borderline-SMOTE1

B-SMOTE2 means borderline-SMOTE2.

Bold values indicate the best result achieved on each dataset

Table 9 Comparison with other traditional classification methods using SMOTETomek and SMOTEENN (G_Mean %)

	1NN w.t. SMOTETomek	1NN w.t. SMOTEENN	RF w.t. SMOTETomek	RF w.t. SMOTEENN	GBDT w.t. SMOTETomek	GBDT w.t. SMOTEENN	NB w.t. SMOTETomek	NB w.t. SMOTEENN	GPFRM Best	GPFRM Mean
Armstrong	81.65+	96.61=	88.56+	90.68+	90.07+	90.07+	89.44+	92.58+	100	**98.59**
Golub	93.54+	93.54+	82.44+	84.92+	92.58+	92.58+	61.24+	80.18+	100	**98.03**
Colon	59.76+	69.01+	60.18+	60.53+	77.26+	67.92+	69.01+	43.64+	88.64	**81.74**
Leukemia	70.08+	90.14=	65.18+	73.36+	83.45+	86.61=	86.61=	**93.54**−	90.14	87.91
DLBCL	84.98=	84.98=	78.65+	71.51+	73.32+	68.72+	86.07=	**91.29**−	100	85.74
gordon	**100**=	**100**=	92.58+	94.63+	81.64+	91.35+	88.19+	88.19+	100	99.36
Yeoh	89.80+	89.80+	72.14+	68.63+	96.08=	96.08=	92.12+	80.47+	100	**97.41**
su-2001	**100**=	**100**=	98.62=	99.38=	**100**=	**100**=	**100**=	**100**=	100	**100**
tomlins	**98.20**=	**98.20**=	84.26+	82.74+	75.31+	78.93+	86.60+	86.60+	100	97.51
Lung	93.86=	95.11=	79.40+	82.84+	**100**−	**100**−	77.46+	77.46+	100	95.31
Summary	5+, 5=, 0−	3+, 7=, 0−	9+, 1=, 0−	9+, 1=, 0−	7+, 2=, 1−	6+, 3=, 1−	7+, 3=, 0−	7+, 1=, 2−		

Bold values indicate the best result achieved on each dataset

6 Conclusions and future work

This paper proposed a new method (GPFRM) to evolve classifiers by employing multiple GP processes for classification with high-dimensional imbalanced data. This paper suggests not only reusing good features previously selected but also reusing good trees in the initialization of the later GP process. Moreover, this paper proposed a new fitness function for AUC approximation to solve the problem of class imbalance in classification with high-dimensional imbalanced data.

In the experiment, ten high-dimensional imbalanced datasets are used to examine the classification performance of GPFRM. Experimental results show that the proposed method significantly reduces training time, and more importantly, the classification performance is increased in most cases, compared to those GP methods using all features as terminals at the same time. For the proposed fitness function, GP using it achieves a competitive performance in most cases, and it is not as time-consuming as GP_{Auc_w}. One piece of future work is to investigate a more flexible method for feature grouping in classification with high-dimensional imbalanced data, instead of randomly splitting features.

Acknowledgements This work was supported in part by the Marsden Fund of New Zealand government under contracts VUW1509 (Mengjie Zhang) and VUW1615 (Bing Xue and Mengjie Zhang), the Science for Technological Innovation Challenge (SfTI) fund under grant E3603/2903 (Mengjie Zhang and Bing Xue), the University Research Fund at Victoria University of Wellington (Grant Number 216378/3764 and 223805/3986, Bing Xue and Mengjie Zhang), MBIE Data Science SSIF Fund under the contract RTVU1914 (Mengjie Zhang and Bing Xue), and National Natural Science Foundation of China (NSFC), under grant 61876169 (Bing Xue) and grant 61672276 (Lin Shang). Wenbin Pei was supported by China Scholarship Council/Victoria University Scholarship.

Compliance with ethical standards

Conflict of interest All authors declare that they have no conflict of interest.

Ethical approval This article does not contain any studies with human participants or animals performed by any of the authors.

References

Aydogan EK, Ozmen M, Delice Y (2019) CBR-PSO: cost-based rough particle swarm optimization approach for high-dimensional imbalanced problems. Neural Comput Appl 31(10):6345–6363

Batista GE, Prati RC, Monard MC (2004) A study of the behavior of several methods for balancing machine learning training data. ACM SIGKDD Explor Newsl 31(10):6345–6363

Bhowan U, Zhang M, Johnston M (2010) Genetic programming for classification with unbalanced data. In: European conference on genetic programming. Springer, p 1–13

Bhowan U, Johnston M, Zhang M (2011a) Ensemble learning and pruning in multi-objective genetic programming for classification with unbalanced data. In: Australasian joint conference on artificial intelligence. Springer, pp 192–202

Bhowan U, Johnston M, Zhang M (2011b) Evolving ensembles in multi-objective genetic programming for classification with unbalanced data. In: Proceedings of the 13th annual conference on genetic and evolutionary computation. ACM, pp 1331–1338

Bhowan U, Johnston M, Zhang M (2012) Developing new fitness functions in genetic programming for classification with unbalanced data. IEEE Trans Syst Man Cybern Part B (Cybern) 42(2):406–421

Bhowan U, Johnston M, Zhang M, Yao X (2013) Evolving diverse ensembles using genetic programming for classification with unbalanced data. IEEE Trans Evol Comput 17(3):368–386

Bhowan U, Johnston M, Zhang M, Yao X (2014) Reusing genetic programming for ensemble selection in classification of unbalanced data. IEEE Trans Evol Comput 18(6):893–908

Blagus R, Lusa L (2013) Improved shrunken centroid classifiers for high-dimensional class-imbalanced data. BMC Bioinform 14(1):64

Chawla NV, Bowyer KW, Hall LO, Kegelmeyer WP (2002) Smote: synthetic minority over-sampling technique. J Artif Intell Res 16:321–357

Chawla NV, Japkowicz N, Kotcz A (2004) Special issue on learning from imbalanced data sets. ACM Sigkdd Explor Newsl 6(1):1–6

Curry R, Lichodzijewski P, Heywood MI (2007) Scaling genetic programming to large datasets using hierarchical dynamic subset selection. IEEE Trans Syst Man Cybern Part B (Cybern) 37(4):1065–1073

Ertekin S, Huang J, Bottou L, Giles L (2007a) Learning on the border: active learning in imbalanced data classification. In: Proceedings of the sixteenth ACM conference on information and knowledge management. ACM, pp 127–136

Ertekin S, Huang J, Giles CL (2007b) Active learning for class imbalance problem. In: Proceedings of the 30th annual international ACM SIGIR conference on research and development in information retrieval, vol 7, pp 823–824

Espejo PG, Ventura S, Herrera F (2010) A survey on the application of genetic programming to classification. IEEE Trans Syst Man Cybern Part C (Appl Rev) 40(2):121–144

Fan W, Stolfo SJ, Zhang J, Chan PK (1999) Adacost: misclassification cost-sensitive boosting. In: Proceedings of the sixteenth international conference on machine learning, vol 99, pp 97–105

Fisher RA (1992) Statistical methods for research workers. In: Kotz S et al. (eds) Breakthroughs in statistics. Springer, pp 66–70

Fleury A, Vacher M, Noury N (2010) SVM-based multimodal classification of activities of daily living in health smart homes: sensors, algorithms, and first experimental results. IEEE Trans Inf Technol Biomed 14(2):274–283

Freund Y, Schapire RE (1997) A decision-theoretic generalization of on-line learning and an application to boosting. J Comput Syst Sci 55(1):119–139

Galar M, Fernandez A, Barrenechea E, Bustince H, Herrera F (2012) A review on ensembles for the class imbalance problem: bagging-, boosting-, and hybrid-based approaches. IEEE Trans Syst Man Cybern Part C (Appl Rev) 42(4):463–484

Gathercole C, Ross P (1994) Dynamic training subset selection for supervised learning in genetic programming. In: International conference on parallel problem solving from nature. Springer, pp 312–321

Haixiang G, Yijing L, Shang J, Mingyun G, Yuanyue H, Bing G (2017) Learning from class-imbalanced data: review of methods and applications. Expert Syst Appl 73:220–239

Han H, Wang W-Y, Mao B-H (2005) Borderline-SMOTE: a new over-sampling method in imbalanced data sets learning. In: International conference on intelligent computing. Springer, pp 878–887

He H, Bai Y, Garcia EA, Li S (2008) Adasyn: adaptive synthetic sampling approach for imbalanced learning. In: IEEE international joint conference on neural networks. IEEE, pp 1322–1328

Hong X, Chen S, Harris CJ (2007) A kernel-based two-class classifier for imbalanced data sets. IEEE Trans Neural Netw 18(1):28–41

Hsieh WW (2007) Nonlinear principal component analysis of noisy data. Neural Netw 20(4):434–443

Joshi MV, Kumar V, Agarwal RC (2001) Evaluating boosting algorithms to classify rare classes: comparison and improvements. In: Proceedings 2001 IEEE international conference on data mining. IEEE, pp 257–264

Joshi A, Dangra J, Rawat M (2016) A decision tree based classification technique for accurate heart disease classification and prediction. Int J Technol Res Manag 3:1–4

Li J, Li X, Yao X (2005) Cost-sensitive classification with genetic programming. In: The 2005 IEEE congress on evolutionary computation, vol 3. IEEE, pp 2114–2121

Li P, Chan KL, Fang W (2006) Hybrid kernel machine ensemble for imbalanced data sets. In: 18th international conference on pattern recognition (ICPR'06), vol 1. IEEE, pp 1108–1111

Liu XY, Wu J, Zhou ZH (2009) Exploratory undersampling for class-imbalance learning. IEEE Trans Syst Man Cybern Part B (Cybern) 39(2):539–550

Liu J, Chen XX, Fang L, Li JX, Yang T, Zhan Q, Tong K, Fang Z (2018) Mortality prediction based on imbalanced high-dimensional ICU big data. Comput Ind 98:218–225

Luna JM, Pechenizkiy M, del Jesus MJ, Ventura S (2017) Mining context-aware association rules using grammar-based genetic programming. IEEE Trans Cybern 48:3030–3044

Patterson G, Zhang M (2007) Fitness functions in genetic programming for classification with unbalanced data. In: Australasian joint conference on artificial intelligence. Springer, pp 769–775

Pears R, Finlay J, Connor AM (2014) Synthetic minority over-sampling technique (SMOTE) for predicting software build outcomes. arXiv:1407.2330

Pei W, Xue B, Shang L, Zhang M (2018) Genetic programming based on granular computing for classification with high-dimensional data. In: Australasian joint conference on artificial intelligence. Springer, pp 643–655

Poli R, Langdon WB, McPhee NF (2008) A field guide to genetic programming. http://www.gp-field-guide.org.uk

Ramentol E, Caballero Y, Bello R, Herrera F (2012) SMOTE-RSB*: a hybrid preprocessing approach based on oversampling and undersampling for high imbalanced data-sets using SMOTE and rough sets theory. Knowl Inf Syst 33(2):245–265

Seiffert C, Khoshgoftaar TM, Van Hulse J, Folleco A (2014) An empirical study of the classification performance of learners on imbalanced and noisy software quality data. Inf Sci 259:571–595

Song D, Heywood MI, Zincir-Heywood AN (2003) A linear genetic programming approach to intrusion detection. In: Genetic and evolutionary computation conference. Springer, pp 2325–2336

Stefanowski J (2016) Dealing with data difficulty factors while learning from imbalanced data. In: Challenges in computational statistics and data mining. Springer, pp 333–363

Tan P-N, Steinbach M, Kumar V (2016) Introduction to data mining. Pearson Education India

Tashk ARB, Faez K (2007) Boosted bayesian kernel classifier method for face detection. In: Proceedings of the third international conference on natural computation. IEEE, pp 533–537

Tax DM, Duin RP (2004) Support vector data description. Mach Learn 54(1):45–66

Tran B, Xue B, Zhang M (2016) Genetic programming for feature construction and selection in classification on high-dimensional data. Memet Comput 8(1):3–15

Tran B, Xue B, Zhang M (2017) Using feature clustering for GP-based feature construction on high-dimensional data. In: European conference on genetic programming. Springer, pp 210–226

Wu G, Chang EY (2005) KBA: kernel boundary alignment considering imbalanced data distribution. IEEE Trans Knowl Data Eng 6:786–795

Yang P, Xu L, Zhou BB, Zhang Z, Zomaya AY (2009) A particle swarm based hybrid system for imbalanced medical data sampling. In: BMC genomics, vol 10. BioMed Central, p S34

Yang P, Yoo PD, Fernando J, Zhou BB, Zhang Z, Zomaya AY (2014) Sample subset optimization techniques for imbalanced and ensemble learning problems in bioinformatics applications. IEEE Trans Cybern 44(3):445–455

Yen SJ, Lee YS (2009) Cluster-based under-sampling approaches for imbalanced data distributions. Expert Syst Appl 36(3):5718–5727

Yin H, Gai K (2015) An empirical study on preprocessing high-dimensional class-imbalanced data for classification. In: IEEE 7th international symposium on cyberspace safety and security (CSS), IEEE 12th international conference on embedded software and systems (ICESS), IEEE 17th international conference on high performance computing and communications (HPCC). IEEE, pp 1314–1319

Yin L, Ge Y, Xiao K, Wang X, Quan X (2013) Feature selection for high-dimensional imbalanced data. Neurocomputing 105:3–11

Zhang S, Qin Z, Ling CX, Sheng S (2005) “Missing is useful”: missing values in cost-sensitive decision trees. IEEE Trans Knowl Data Eng 17(12):1689–1693

Zhou ZH, Liu XY (2006) Training cost-sensitive neural networks with methods addressing the class imbalance problem. IEEE Trans Knowl Data Eng 18(1):63–77

Zhu Z, Ong YS, Dash M (2007) Markov blanket-embedded genetic algorithm for gene selection. Pattern Recognit 40(11):3236–3248