GEML: A Grammatical Evolution, Machine Learning Approach to Multi-class Classification

Jeannie M. Fitzgerald(✉), R. Muhammad Atif Azad, and Conor Ryan

Biocomputing and Developmental Systems Group,
University of Limerick, Limerick, Ireland
{jeannie.fitzgerald,atif.azad,conor.ryan}@ul.ie

Abstract. In this paper, we propose a hybrid approach to solving multi-class problems which combines evolutionary computation with elements of traditional machine learning. The method, *Grammatical Evolution Machine Learning* (GEML) adapts machine learning concepts from decision tree learning and clustering methods and integrates these into a Grammatical Evolution framework. We investigate the effectiveness of GEML on several supervised, semi-supervised and unsupervised multi-class problems and demonstrate its competitive performance when compared with several well known machine learning algorithms. The GEML framework evolves human readable solutions which provide an explanation of the logic behind its classification decisions, offering a significant advantage over existing paradigms for unsupervised and semi-supervised learning. In addition we also examine the possibility of improving the performance of the algorithm through the application of several ensemble techniques.

Keywords: Multi-class classification · Grammatical evolution · Evolutionary computation · Machine learning

1 Introduction

Evolutionary algorithms (EAs) are algorithms which are inspired by biological evolution and which are constructed to emulate aspects of evolution, such as genetic mutation and recombination and the notion of natural selection. Genetic Programming (GP) [29] is an evolutionary algorithm which has been successful on a wide range of problems from various diverse domains [19], achieving many *human competitive* results [4]. However, a significant proportion of previous work has concentrated on supervised learning tasks and, aside from some notable exceptions, studies on unsupervised and semi-supervised learning have been left to the wider machine learning (ML) community.

Two of the most important problems types which benefit from the application of ML techniques are regression and classification, and GP has proven itself as an effective learner on each of these: achieving particularly competitive results on symbolic regression and binary classification tasks. Although many studies have

J.J. Merelo et al. (eds.), *Computational Intelligence*,

been undertaken, multi-class classification (MCC) remains a problem which is considered challenging for traditional tree based GP [11].

While we are concerned with multi-class classification generally, an important motivation for the current investigation is the requirement for an algorithm which can be applied to multi-class grouping/categorisation tasks involving both labelled and unlabelled inputs from the *medical domain*, where the unsupervised algorithm must be able to supply *human interpretable* justification for categorisation decisions.

Clustering is a natural choice for this type of task, but standard clustering algorithms generally fail to satisfy the requirement of providing the reasoning behind cluster allocations in a human readable form. In the medical domain, it is usually important that the learner has the capability to provide human understandable explanations of its decisions so that human experts can have confidence in the system. In this respect, decision trees (DTs) have the attractive property that the induced DT itself provides an easily comprehensible explanation of all decisions. Unfortunately, traditional DTs rely on ground truth information to make decisions and use of this information is not permissible in an unsupervised context. For these reasons, although each of these methods have attractive properties, we conclude that neither DTs nor clustering approaches are, *in their normal mode of use,* appropriate for unsupervised categorisation tasks which require an explanation from the learner.

Although there is some important existing work in the area of unsupervised classification in the medical domain, including for example [6,22,32], the subject remains relatively unexplored. This paper takes up a triple challenge: it investigates MCC in a supervised, semi-supervised as well as in an unsupervised context.

We hypothesise that it may be possible to combine the desirable qualities of both algorithms by taking the underlying concepts and wrapping them in an evolutionary framework – specifically a grammatical evolution (GE) [40] framework. This approach is appealing due to its symbiotic nature: a GE grammar is used to generate human readable decision tree like solutions and the evolutionary process is applied to the task of optimising the resulting cluster assignments – thus emulating both the decision making behaviour of DTs and the iterative operation of traditional clustering approaches. Not only does GE produce human readable solutions, but it has been shown [3] that the paradigm seems to be able to avoid bloated, over-complex ones.

While we can hypothesise that this hybrid approach might be a good idea, objective evaluation of algorithm performance is required before concluding that the resulting models are likely to be of any practical use. One approach to accomplishing this is to compare results of the hybrid method with other unsupervised algorithms using some common metric of cluster performance. However, it could be argued that without ground truth information any method of comparison is flawed. Another possible approach for evaluating the effectiveness of the proposed method would be to apply it to data about which something is already

known, where that knowledge is not used in the learning process for the purpose of evaluating and comparing performance afterwards.

Considering our original objective, we were also interested in learning about potential performance differences may expected between our unsupervised system and, supervised and semi-supervised approaches using the same data. Thus, we choose to construct this study such that it would be possible to compare the performance and behaviour of the hybrid unsupervised learner with another state of the art unsupervised algorithm as well as with supervised and semi-supervised learners on the same data. Rather than using our original medical dataset at this point, we chose to carry out this first study using controllable synthetic data as outlined in Sect. 4.2, with the intention of optimising the GEML system based on lessons learned, if results of these preliminary experiments prove encouraging. Once optimised, the system can be applied to the more challenging medical datasets in the future.

In this work, we investigate the hypothesis that combining ML concepts with GE can facilitate the development of a new hybrid algorithm with three important properties: the ability to learn multi-class problems in both supervised and unsupervised environments, and the capability of producing human readable results. However, due to the way in which we have designed the experiments – so that meaningful evaluation of the proposed algorithm would be possible, the resulting system delivers much more than initially planned – functioning in supervised, unsupervised and semi-supervised domains.

In summary, we investigate a hybrid GE system which incorporates ideas from two well known ML techniques: decision tree learning which is often applied to supervised tasks, and clustering methods which are commonly used for unsupervised learning tasks. The proposed system which we call GEML is applied to supervised, unsupervised and semi-supervised MCC problems. Its performance is compared with several state of the art algorithms and is shown to outperform its ML counterparts and to be competitive with the best performing ML algorithm, on the datasets studied.

In this work we extend and describe in greater detail, the GEML framework as previously proposed in [17]. In the remainder of this section we will briefly explain some of the concepts employed.

1.1 Clustering

Clustering involves the categorisation of a set of samples into groups or subsets called clusters, such that samples allocated to the same cluster are similar in some way. There are various types of clustering algorithms capable of generating different types of cluster arrangements, such as flat or hierarchical clustering. One of the best known clustering algorithms is K-means clustering which works in an iterative fashion by creating a number of centroids (aspirationally cluster centres). The algorithm groups samples depending on their proximity to these centres and then measuring the distance between the data and the nearest centroids – K-means iteratively minimises the sum of squared distances by

changing the centroid in each iteration and reassigning samples to possibly different groups. Of the EC work in the existing literature which combines GP or GE with unsupervised methods, K-means is the most popular of those used, as is outlined in Sect. 2.

1.2 Decision Tree Learning

A decision tree is a hierarchical model that can be used for decision-making. The tree is composed of internal decision nodes and terminal leaf nodes. In the case of classification for example, internal decision nodes represent attributes, whereas the leaf nodes represent an assigned class label. Directed edges connect the various nodes forming a hierarchical, tree-like structure. Each outgoing edge from an internal node corresponds to a value or a range of values of the attribute represented by that particular node. Tree construction is a filtering and refining process which aims to gradually separate samples into the various classes with possibly multiple routes through the decision process for a particular class assignment.

1.3 Unsupervised, Semi-supervised and Supervised Learning

In simple terms, supervised, semi-supervised and unsupervised learning methods are differentiated by the amount of ground truth information that is available to the learning system: in supervised learning systems the 'answer' which may, for example, be a target variable or a class label is known to the system; semi-supervised systems may have access to such information for a limited number of samples or may involve revalidation of the automated prediction with expert knowledge; and unsupervised learners do not have any ground truth information with which to guide the learning process.

Although classification and clustering are conceptually similar, in practice the techniques are usually used in fundamentally different ways: clustering methods are generally applied to unsupervised tasks and do not require either training data or ground truth label information, whereas classification is usually a supervised task which requires both. At a basic level the goal of clustering is to group similar things together without reference to the name of the group or what membership of a group represents, other than the fact that the members are similar in some way, whereas the objective of classification is to learn, from examples, relationships in the data which facilitate the mapping of training instances to class labels, such that when presented with a new unseen instance the classification system may assign a class label to that instance based on rules/relationships learned in the training phase.

2 Previous Work

An exhaustive review of the application of EAs to clustering methods and decision tree induction is beyond the scope of this paper. Here we have chosen to

focus on the most recent work and that which we determined to be most relevant to the current study. For a comprehensive survey of EAs applied to clustering, the interested reader is directed to [23], whereas a detailed review of EAs applied to decision tree induction can be found in [5].

Relative to the volume of existing research on supervised learning in the field of Evolutionary Computation (EC), unsupervised and semi-supervised learning have received little attention. Of the existing work, a significant proportion in the area of unsupervised learning recommends the use of clustering methods for feature selection [28,31,33], and the majority of this work recommends a traditional K-means approach.

[36] used clustering was used in an interesting way whereby a Differential Evolution (DE) algorithm with built-in clustering functionality was proposed. They studied its effectiveness on an image classification task, and compared their results with several well known algorithms such as K-means but reported statistically indistinct results.

A different unsupervised GP approach was proposed in [35] where a novel fitness function was used in feature selection for the purpose of identifying redundant features. The authors reported superior results when performance was compared with several state of the art algorithms. GP was again employed in [21] where it was used to develop low level thin edge detectors. In that work the authors demonstrated that edge detectors trained on a single image (without ground truth) could outperform a popular edge detector on the task of detecting thin lines in unseen images.

Another novel application of K-means was proposed by [25] who integrated it into GP and used this hybrid approach for problem decomposition – grouping fitness cases into subsets. They applied their strategy to several symbolic regression problems and reported superior results to those achieved using standard GP. They later developed a similar approach [26] for time series prediction.

On the subject of multi-class classification problems, there have been several interesting approaches using tree based GP including strategies for decomposing the task into multiple binary problems [48], treating MCC problems as regression tasks [11] and experimenting with various thresholding schemes such as [51]. Other methods have been proposed which utilise GP variants including multi level GP (MLGP) [50], Parallel linear GP [16] and probability based GP [47] to name a few.

There have been several other evolutionary approaches to MCC including self-organising swarm (SOSwarm) which was described in [38]. In that work, particle swarm optimisation (PSO) was used to generate a mapping which had some similarities to a type of artificial neural network known as a self organising map. SOSwarm was studied on several well known classification problems and while the average performance seemed to degrade as the number of classes increased – the best performing solutions were competitive with the state of the art.

DTs have previously been combined with GE in [13]. The algorithm was applied to the binary classification task of detecting gene-gene interactions in

genetic association studies. The researchers reported good results when their GE with DT (GEDT) system was compared with the C.45 DT algorithm. Our suggested approach has some similarities to this work. However, that research focused on a supervised binary task where attribute values were restricted to a common set of 3 items.

Clustering methods have also been applied to MCC problems. A hybrid method which combined a GA with local search and clustering was suggested in [41], where it was applied to a multi-class problem on gene expression data. The results of that investigation showed that their method (HGACLUS) delivered a competitive performance when compared with K-means and several earlier GA approaches described in [12,30]. GP was again combined with K-means clustering for MCC in [1] where the researchers used the K-means algorithm to cluster the GP program semantics in order to determine the predicted class labels.

Competitive results were also reported in [34] in which K-Means clustering was again used with GP for MCC. There, clustering was combined with a multigenic GP approach in which each individual was composed of several solution parse trees having a common root node.

Concerning DTs, [5] concluded that good performance of EAs for decision tree induction in terms of predictive accuracy had been empirically established. They recommended that investigation of these algorithms on *synthetic data* should be pursued and also the possibility of using evolutionary computation for the evolution of decision tree induction *algorithms*. In this paper we address the first of these recommendations. The candidate solutions evolved by GE are computer programs which emulate decision trees, and these computer programs are produced using a grammar template capable of generating a multitude of different solutions. Thus, it could be argued that the proposed approach does, at least in some sense, also meet the second objective – the evolution of DT induction algorithms.

The novel contributions of this study are the proposal of a technique for unsupervised learning using an EA where the evolved learning hypotheses are in human readable form, and the extension of this to the development of a hybrid GE framework which can also be used for supervised and semi-supervised learning. The new system which we call GEML is successfully applied to the problem of multi-class classification. We also investigate the effectiveness of extending the GEML method through the construction of ensemble, majority vote learners.

3 Proposed Method

In ML DTs generally employ the concept of *information gain* to inform branching decisions during the construction of a decision tree, where the measure of information gain used relies on knowledge of the ground truth labels. While it is normal practice to make use of the ground truth information in the training data for a supervised learner, this is not possible for unsupervised methods and to a limited extent for semi-supervised methods as there is no such data available in the unsupervised case, and only limited aces to ground truth labels in

the semi-supervised domain. Instead, we construct of an *if then else* structure where the if component may be used to test various conditions pertaining to the data, whereby the learning system has access to both the attribute values and also to the variance of each attribute on the training data. Thus, by design our system does not currently implement DTs according to a strict definition of the algorithm, as using label information precludes unsupervised learning.

3.1 Grammatical Evolution

Grammatical Evolution (GE) [40] is a flexible EC paradigm which has several advantages over other evolutionary methods including standard GP. In common with its traditional GP relative, GE involves the generation of candidate solutions in the form of executable computer programs. The difference is that GE does this using powerful grammars whereas GP operates with a much more limited tool-set.

A key aspect of the GE approach is *genotype phenotype separation* whereby the genotype is usually (but not necessarily) encoded as a vector of integer *codons*, some or all of which are *mapped* to production rules defined in a user specified grammar (usually in Backus-Naur-Form). This mapping results in a phenotype executable program (candidate solution). GE facilitates focused search through the encoding of domain knowledge into the grammar and the separation of search and solution space such that the search component is independent of the representation and may, in principle, be carried out using any suitable algorithm – a genetic algorithm is often used but other search algorithms such as PSO [39] and DE [37] have also been used to good effect.

The role of the user-defined grammar is key to guiding the evolutionary search towards desirable solutions. The grammar is essentially a specification of what *can be evolved* and it is up to the evolutionary system to determine which of the many possible solutions which can be generated using the specification *should be evolved* [45]. A small change in a grammar may induce drastically different behaviour. In this work we have designed a grammar, shown in Fig. 1, which facilitates the assignment of data instances to clusters based on the results of applying simple 'if then else' decision rules. While the individual rules are quite simple, the grammar allows for the construction of powerful expressions capable of representing both simple and complex relationships between attributes as demonstrated in Fig. 2.

3.2 Objective Functions

For each of the three learning problems: supervised, unsupervised and semi-supervised, we employ a different objective function to drive evolutionary progress. In the supervised case we use classification accuracy which is simply the proportion of instances correctly classified by the system. Since this study uses balanced data sets, we simply use the number of correct predictions to measure system performance. Accuracy values range between 0 and 1 where

```
<expr> ::=<label> if <cond> else <label>
  | <label>if<cond>else<expr>
  | <label>if<cond>and<cond>else <expr>
  | <label>if<cond>or<cond>else<expr>
  | <expr> if <cond> else <expr>
  | <expr> | <label>

<label> ::= 0 | 1 | 2 | 3 | 4

<cond> ::= <attr><relop><const>
          | <subExpr><relop><const>
          | <subExpr><relop><subExpr>
          | <var><relop><var>
          | <attr><relop><attr>

<subExpr> ::= <attr><op><const>
     | <attr><op><var>
     | <attr><op><attr>

<op>  ::= + | - | * | /
<relop> ::= <=
<const> ::= <digit>.<digit>
    | -<digit>.<digit>

<digit> ::= 0 | 1 | 2 | 3 | 4
    | 5 | 6 | 7 | 8 | 9

<attr> ::= x[0] | x[1] | x[2]
<var>  ::= v[0] | v[1] | v[2]
```

Fig. 1. Example grammar for five class problem with three attributes ($< attr >$). The $< var >$ entries represent the variance in the training data across each attribute. The 'then if else' format is designed to simplify the syntax required in a python environment – as the system evolves python expressions. The division operation is protected in the implementation.

```
2 if x[0]*v[1]<=-0.2 and v[2]<=v[1]
else 1 if (x[1]*v[0])<=(x[0]*-0.8)
else 0 if x[2]/v[2]<= x[0]*-2.0 or x[2]<=x[1]
else 1 if x[1]<=x[0]
else 1 if x[0]*x[2]<=x[1]/x[0] and x[1]/v[2]<=-0.3)
else 2 if x[2]/v[2]<=x[0]*2.9 or x[2]<=x[1]
else 1 if x[0]<=-0.1 else 0
```

Fig. 2. Example expression generated for a three class, three attribute classification task.

1 represents perfect classification. The system is configured with an objective function designed to maximise fitness.

For the unsupervised task we have chosen to use a metric of clustering performance known as a *silhouette co-efficient* or *silhouette score* (SC) as the objective function. The SC is a metric which does not require knowledge of the ground truth which makes it suitable for use in an unsupervised context. For each data point two measures are calculated: a. the average distance (according to some distance metric) between it and every other point in the same cluster and b. the mean distance between it an all of the points in the nearest cluster that is not it's own cluster. The silhouette score over all points is calculated according to the formula shown in Fig. 3. Our system tries to maximise this value during the evolutionary process. Note that this approach, which aims at optimising the SC rather than cluster centroids, is quite different from the other EC methods outlined in Sect. 2 where clustering tasks have generally been tackled using the K-means algorithm.

The silhouette score ranges between −1 and 1, where a negative value implies that samples are not assigned to the correct clusters, a value close to 0 indicates that there are overlapping clusters and a score close to 1 means that clusters are cohesive and well separated.

$$(b-a)/max(a,b) \tag{1}$$

Fig. 3. Silhouette co-efficient.

To calculate the silhouette co-efficient it is first necessary to choose an appropriate distance metric from the many and varied options available in the literature. In this work we have used *cosine distance* also known as *cosine similarity* as it is suitable for determining the similarity between vectors of features and obviates the need for data normalisation. Also, we experimented with several metrics including euclidean and mahalanobis distance before choosing *cosine distance* – as its use resulted in the best results over a range of synthetic classification problems. The results for cosine distance were better in terms of classification accuracy when cluster assignments were converted to class labels using a standard approach.

Semi-supervised learning is suitable for classification situations where some but not all ground truth labels are available. It may be the case, for example, that scarce or expensive human expertise is required to determine the labels. In these cases, it is usually possible to improve unsupervised performance by adding a small number of labelled examples to the system. Although, our synthetic data is fully labelled, we simulate partial labelling by only considering a random subset of training data (20 % of the full data set) to be labelled; the rest of the data set is treated as unlabelled. We compute prediction accuracy on the labelled data and the silhouette score on the unlabelled set. We then add the two measures to get a final score and strive to maximise this score during evolution.

To summarise, candidate solutions are generated using a specification described in a grammar such as the one shown in Fig. 1, and the same grammar is used for all of the GEML problem configurations. Then, applying the decision rules defined in the grammar problem instances are assigned to clusters as shown in Fig. 4 and then depending on whether the task is supervised, unsupervised, or semi-supervised the system tries to optimise the classification accuracy, the silhouette score or a combination of the two. The key point here, is that the same grammar is used for each type of learning – only the objective function is different.

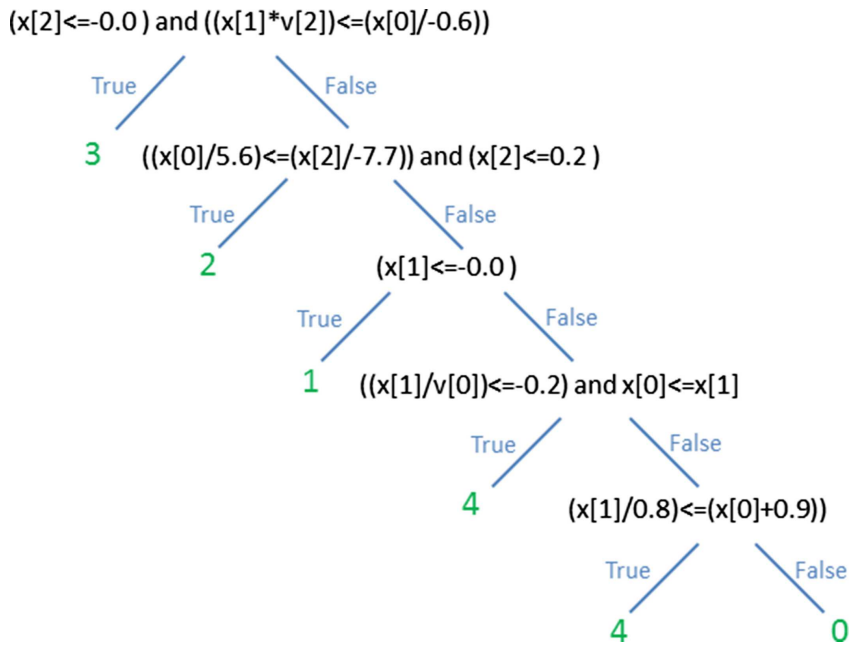

Fig. 4. Example expression tree.

Figure 4 illustrates the expression tree of an example solution for a five-class task. Similar to a DT, the internal nodes of the tree represent branching decision points and the terminal nodes represent cluster assignments.

4 Experiments

In this section we outline the construction of our experiments including the parameters, datasets and benchmarks used. We also detail the results of these experiments together with the results achieved on the same problems with our chosen benchmark algorithms. Details of the naming convention for the various experimental configurations are shown in Table 1.

4.1 Benchmark Algorithms

As we incorporate ideas from DT learning and clustering methods into our hybrid GEML approach, it is appropriate that we benchmark the proposed approach

against Decision Trees [10] as a supervised method and against the K-means clustering algorithm [49] as an unsupervised paradigm. For semi-supervised learning we compare with a label propagation (LP) [43] algorithm. The idea behind LP is similar to k-Nearest Neighbours (k-NN) [2] and was originally proposed for detecting community structures in networks.

We also compare with support vector machines (SVMs) [8] for supervised learning as the method may provide a useful benchmark as it is known to achieve good results with balanced datasets, which is the case here, and while SVMs are inherently binary classifiers they can perform multi-class classification in various ways, most commonly using a "one versus all" strategy.

For comparison purposes we choose simple classification accuracy as a performance metric. It has been empirically established in the GP literature that simple classification accuracy is not a reliable measure of classification on *unbalanced* datasets [7], and that other measures such as average accuracy or Matthews Correlation Co-efficient might be more appropriate especially if combined with a sampling approach [18]. However, in this preliminary investigation, the classes *are balanced* which allows us to consider simple classification accuracy as a reasonable measure, particularly as we want to be able to observe differences in performance across the various levels of learning.

Table 1. Experimental configurations.

Configuration	Explanation
GEML-SUP	Supervised GEML
GEML-SEMI	Semi-supervised
GEML-UN	Unsupervised GEML
DT	Decision Tree Learning
LP	Label Propagation
KM	K-means Clustering
SVM	Support Vector Machine

We adopt a popular mechanism to determine which predicted label represents which a priori class label: the predicted class is mapped to the a priori class which has the majority of instances assigned to it, e.g. for a binary task with 1000 training instances, if predicted class 1 has 333 members of ground truth class 1 assigned to it and 667 instances of class 0, then predicted class 1 is determined to represent the a priori class 0. It is important to note that this method is used to calculate the accuracy metric and used only for reporting and comparison purposes across all tasks and methodologies. The measure (classification accuracy) is the main driver of the evolutionary process in the supervised tasks, is used on only a percentage of the training instances in the semi-supervised case. In all cases, the same mapping from cluster assignment to

class label determined during the training phase also applies when evaluating performance on test data.

For each of the GEML methods the evolved solutions have similar form to the example shown in Fig. 2. This is essentially a python expression that can be evaluated for each training and each test instance. The result of evaluating the expression on a given instance is an integer which is converted into first a cluster assignment and then a class label, using the method previously described. Although the objective functions used to determine fitness and drive evolution differ according to the type of learning model as detailed in Sect. 3.2, we calculate the classification accuracy for each unevaluated individual at each generation on training and test data. At no time is the test data used in the learning process.

For each problem, for each dataset and each learner, the algorithm was run fifty times using the same synthetic datasets and train/test splits. A different random seed was used for each run of the same algorithm and these same random seeds were used for the corresponding run of each algorithm. The popular scikit-learn [42] python library for machine learning was used for all of these ML experiments.

4.2 Datasets

The various algorithms were tested on several synthetic multi-class datasets which were produced using the scikit-learn library [42] which provides functionality for the generation of datasets with the aid of various configurable parameters. For this study we investigate balanced multi-class problems of two, three, four and five classes each. The library facilitates user control of the number, type and nature of features selected for experiments. For example, features can be informative, duplicate or redundant. We have chosen to use informative features only for the current work.

Given a problem configuration (number of classes), for each run of each algorithm a dataset of 1000 instances was generated and then split into training and test sets of 700 and 300 instances respectively. Identical random seeds were used for the corresponding run for each configuration, such that the same dataset was generated for each setup for a particular run number.

We have chosen to use synthetic datasets: 1000 instances were generated, without added noise, and with few features, each of which is informative. Employing synthetic datasets allows us to configure the data to have *informative* features such that it is, as far as possible, amenable to being clustered or classified. We have made these choices in an effort to ensure that the data is not biased to favour any particular algorithm or learning paradigm. For example, DTs are known to over-fit and not generalise well where there are a large number of features and few instances.

The decision to use synthetic datasets also delivers on the recommendation of [5] to use synthetic data for decision tree induction, as described in Sect. 2.

Table 2. Evolutionary parameters.

Parameter	Value
Population size	500
Replacement strategy	Generational
Number of generations	100
Crossover probability	0.9
Mutation probability	0.01

4.3 Evolutionary Parameters

Important parameters used in these experiments are outlined in Table 2. Evolutionary search operators in GE are applied at the genotypic level, and in this work each individual's genotype is a linear genome represented by a vector of integers. The mutation operator operates by replacing a single integer with a new one randomly generated within a predefined range. One point crossover is used, whereby a single crossover point is randomly and independently selected from each of the two parents (that is, the two points are likely to correspond to different locations) and two new offspring are created by splicing parental segments together. In both cases, these operations take place in the effective portion of the individual, i.e. the segment of the integer vector that was used in the genotype to phenotype mapping process – sometimes a complete phenotype is generated before requiring the full integer vector.

4.4 Experimental Results

Results for average and best training and test accuracy can be seen in Table 3, where for convenience the best result in each category is in bold text. For comparison purposes we are interested in comparing the supervised methods with each other and the unsupervised approaches with the other unsupervised methods etc. Thus we compare GEML-SUP with both DT and SVM, GEML-UNS with KM and GEML-SEMI with LP. However, we are also interested in observing the relative performances of the three different levels of learning.

Looking first at the supervised approaches, we can see that the SVM approach performs well across all of the problems studied with regard to average classification accuracy on both training and test data. Encouragingly, the GEML-SUP configuration is very competitive with SVM on the first three problems and outperforms DT on each of those tasks.

On the semi-supervised experiments GEML-SEMI outperforms LP on all problems for both training and test data in terms of average classification accuracy.

Finally, with regard to the unsupervised set-ups GEML-UN outperforms KM for average accuracy on training and test data on all problems.

For each algorithm, the performance of the various configurations degrades as the number of classes increases which is not surprising as adding more classes

increases the difficulty of the problem to be solved. Overall, the SVM algorithm suffered least from this issue, which is again not surprising given that the implementation used here [42] solves MCC problems using a binary decomposition strategy.

Reviewing the results in Table 3, we see that values for *best overall* training and test accuracy on the binary and three class tasks for each of the GEML methods are not competitive with the other approaches. For example, on the three class task, the average test accuracy for K-means is 0.74 whereas the best result is 0.99 compared with GEML-UN which has an average test accuracy of 0.75 and a best result of 0.86 and the GEML-SUP which has an average test accuracy of 0.92 and a best result of 0.95. The results for each of the GEML setups show that the reported standard deviation is lower than for the other algorithms.

It is unclear to us whether this phenomenon is associated with the GE paradigm itself, the nature of the MCC problem or some other factors. However, it could be argued that the behaviour is not necessarily a negative result, as having a larger standard deviation with a higher extreme value can also mean that the algorithm is unreliable. After all, a good test set performance is only valuable if it is consistently achieved, not as an exceptional case.

Due to the stochastic nature of GE one might hypothesise that there is a higher probability of many individuals achieving good results across many runs on the easier one and two class problems than on the more difficult problems where individuals have to learn to incorporate a larger number of class labels: due to the added complexity there are likely to be fewer fit solutions early in the evolutionary process and thus fewer opportunities to improve through the application of genetic operators. One can easily imagine that there could be significant variability across runs depending on the quality of the initial population, and the existence of fewer highly fit solutions reduces the probability of truly excellent ones emerging.

Looking at the generalisation performance of each method in terms of the variance component, we adopt a simple measure whereby the variance error is simply the difference in performance of the various learning hypothesis between training and test data. In this respect, of the algorithms studied only the LP approach exhibits high variance. The various GEML methods all produce good generalisation performance. This is quite interesting as its close relation GP is known to exhibit a low bias high variance behaviour [27]. We can hypothesise that possible contributing factor to this contrast in behaviour is due to the grammar used, even if it contains recursive rules it is likely to constrain the size of the evolved programs. [3] demonstrated empirically that while program size tends to increase steadily during GP runs, the average size of GE genomes remains roughly static after an initial period of growth or shrinkage. In those experiments, GP genomes were consistently larger than GE genomes after only fifty generations. It may be the case that these effects are preventing the evolved models from becoming over complex. Recent results presented in [3] would suggest

that GE does not over-fit on regression problems, where the required grammar would be fundamentally different, either.

If we analyse the difference in performance between the various supervised, semi-supervised and unsupervised algorithms, it is not surprising that in all cases the supervised algorithms produced the best results and that the semi-supervised algorithms performed better than the unsupervised ones. Of course, the unsupervised and semi-supervised methods are not usually evaluated in the same way as supervised classification approaches: using accuracy as a performance metric. We have chosen to do so here as a convenient and practical way to gain some insight into the likely relative performance of our hybrid technique when it is applied to the three learning approaches.

The results suggest that while the performance of all of the algorithms deteriorates as the number of classes increases, this effect is even more evident for the unsupervised and semi-supervised methods where the performance of GEML-UN drops from 90 % on the binary task to 66 % for the five class problem, although this is still better than the corresponding LP algorithm. Again, we can hypothesis that while adopting a binary decomposition approach may seem attractive, this would be very challenging in an unsupervised context. However, there may be some scope for the strategy in the semi-supervised paradigm.

Statistical Analysis. We carried out tests for statistical significance on the test results using the non-parametric Mann-Whitney U-Test. This revealed that statistical significance of results sometimes varied depending on the problem. Any differences between SUP and SVM were not significant for the two and three class problems but for the four and five 5 ones SVM is significantly better with 99 % confidence. Comparing SUP against DT, the differences are significant at the 95 %, 99 % and 99 % confidence levels for the two, three and four class problems respectively (SUP is better), but not significant for the five class task. For the semi-supervised tasks, any differences are not significant for the binary task but the GEML-SEMI results are significantly better at the 99 % confidence level for the other three problems. Finally, the analysis comparing GEML-UNS with K-Means behaves similarly, where GEML-UNS is significantly better on the two, four and five class tasks having confidence levels of 99 %, 95 % and 95 % respectively, and with a p-value of 0.58 there was no significant difference on the three class task.

5 Ensemble Approaches

The results demonstrate that while the GEML approach is competitive with the best ML algorithms on the two and three class supervised tasks, SVMs outperform GEML on the four and five class problems. As GE is a non-deterministic algorithm we hypothesised that it may be possible to improve its relative performance on the more difficult four and five class tasks, by generating GE *majority voting classifiers* which have previously [9,15,20] been shown to be effective in

Table 3. Average and best classification accuracy on training and test data.

Task	Method	Avg. training accuracy	StdDev	Best training accuracy	Avg. test accuracy	StdDev	Best test accuracy
C2	GEML-SUP	**0.96**	0.01	0.98	**0.96**	0.01	0.97
	DT	0.93	0.04	0.99	0.93	0.04	0.99
	SVM	0.95	0.03	0.99	0.95	0.03	0.99
	GEML-SEMI	0.90	0.01	0.93	0.91	0.01	0.92
	LP	0.90	0.06	0.99	0.88	0.07	0.99
	GEML-UN	0.90	0.01	0.92	0.91	0.01	0.93
	KM	0.84	0.08	0.99	0.84	0.08	0.99
C3	GEML-SUP	**0.93**	0.01	0.94	**0.92**	0.02	0.95
	DT	0.87	0.04	0.97	0.88	0.05	0.99
	SVM	0.92	0.03	0.98	**0.92**	0.04	0.99
	GEML-SEMI	0.88	0.04	0.94	0.87	0.04	0.92
	LP	0.83	0.05	0.96	0.79	0.08	0.93
	GEML-UN	0.76	0.04	0.87	0.75	0.04	0.86
	KM	0.75	0.07	0.91	0.74	0.08	0.99
C4	GEML-SUP	0.86	0.01	0.88	0.86	0.02	0.89
	DT	0.82	0.04	0.94	0.83	0.04	0.93
	SVM	**0.88**	0.03	0.94	**0.88**	0.03	0.96
	GEML-SEMI	0.78	0.05	0.84	0.78	0.05	0.85
	LP	0.77	0.05	0.88	0.71	0.07	0.85
	GEML-UN	0.71	0.04	0.79	0.71	0.04	0.81
	KM	0.65	0.06	0.83	0.67	0.07	0.84
C5	GEML-SUP	0.77	0.06	0.85	0.75	0.04	0.83
	DT	0.77	0.04	0.88	0.79	0.05	0.89
	SVM	**0.85**	0.03	0.93	**0.86**	0.03	0.94
	GEML-SEMI	0.72	0.04	0.76	0.75	0.06	0.82
	LP	0.71	0.05	0.84	0.65	0.07	0.83
	GEML-UN	0.66	0.06	0.77	0.69	0.07	0.82
	KM	0.63	0.05	0.78	0.63	0.06	0.79

improving classifier generalization. In general, these approaches operate by combining a large number of classifiers and then classifying each instance with the class label of the class which receives the greatest number of votes.

A *weak learner* is one whose accuracy in labelling examples may be only slightly better than random guessing whereas a strong learner is one whose predictions are strongly correlated with the true labels. It has been well established in the literature [24,46] that rather surprisingly, *weak learners* can be combined to produce much stronger models. Indeed the strategy is so successful that it has been widely adopted in evolutionary computation and other ML algorithms. See [44] for a comprehensive review.

We have chosen to investigate two different strategies for combining our GEML models. In the first instance we combine the predictions of the best of run models of each run giving a total of fifty models to form an ensemble and we call this configuration ensBest. Secondly, we explored an approach whereby we combined every model from each generation whose accuracy exceeded a predefined threshold and we refer to this approach as ensPop. Using this second approach, which may generate thousands of models to add to the ensemble we are interested to discover if the combined approach may achieve better accuracy that our single best model.

For these initial experiments within the GE runs, we initially chose to apply a weak threshold of 0.60 accuracy whereby any individual whose fitness was greater would have its predictions added to the ensemble. However, the accuracy scores produced by these generated ensembles were not at all encouraging – often several percent worse than the best individual score. In the final experiments we set the threshold to be a value which was 10 % lower than the average training accuracy of the population as per Table 3.

As the evolutionary system converges the population becomes dominated over time by very similar or identical individuals. We chose not to allow duplicates to join the ensemble. Once an ensemble is constructed, we examine the prediction correlation between the candidate predictions using the Pearson correlation coefficient and then eliminate potential solutions which were greater than 90 % correlated with more than two thirds of their ensemble mates. These choices were designed to promote diversity in the ensemble which has been determined to be a necessary condition for constructing an effective ensemble [14]. However, the values chosen are somewhat ad-hoc, and it is likely that they could be improved upon. The construction and modification of the ensemble membership is carried out on the training data and then the test predictions of the final ensemble members are compared with the ground truth.

The results obtained shown in Table 4 indicate that the ensembles constructed from the fifty best-of-run individuals, for the four class problem, achieved the same average test accuracy but did not improve on the result for the single best individual previously reported. For the five class task the ensBest ensemble was significantly better that the average test result and produced a slight improvement on the overall best individual score.

Table 4. Ensemble results.

Task	Average test	Best test	ensBest	Average ensPop
4Class	0.86	0.89	0.86	0.87
5Class	0.75	0.83	0.84	0.80

Similarly, the larger ensembles, which may be constructed from predictions of thousands of members, even after the duplicates and highly correlated ones have been removed, produce test results which are better than the end-of-run population average. There may be potential to further improve the performance of the ensPop ensemble construction by, for example, determining which candidates are well correlated with the true training labels and assigning a higher weight to the predictions of those individuals on the test data. It is interesting to note that both ensemble approaches did comparatively better on the more difficult five class task than on the four class problem.

5.1 Discussion

This is a simple study into the potential of the GEML system to tackle multi-class classification tasks which may be supervised, unsupervised or semi-supervised in nature. Although the results are quite encouraging we feel that there is potential for improvement in the existing system. The obvious place to look for improvement is the all-important grammar. Our next steps will be to examine this to see how we can make it more effective. As a first move in that direction we will analyse the best individuals from our existing runs to determine which rules are contributing most and which are not performing. We will then modify the system applying this new information and use it to tackle a large, potentially noisy real-world medical dataset.

In the results section of this paper we have compared with several multi-class classification algorithms, and the reported results demonstrate that the most successful supervised technique is SVM. However, it is perhaps fair to point out that SVMs are not inherently multi-class, rather the algorithm usually (but not always) implements multi-class problems in either a "one versus one" or a "one versus all" approach, which in fact was how SVMs were implemented in this study. Thus, the performance of GEML and SVMs are, in one sense, not directly comparable. Given that the average performance of GEML on the two and three class problems is very competitive with that of the SVM, it is reasonable to hypothesise that equivalent performance to SVMs which use binary decomposition, might be expected on problems with greater numbers of classes if the GEML method were adapted to also perform multi-classification by way of binary decomposition. It may also be worth re-iterating that unlike SVMs the GEML setups all provide human readable solutions, which is an important consideration in many problem domains.

We have seen in this investigation that the GEML system which incorporates ideas from decision tree and cluster based learning has produced some statistically significant results. As GE is such a flexible paradigm there is no reason why

alternative ML algorithms could not be incorporated instead. Once the candidate ML algorithm has some aspect which requires optimisation it should be suitable for an evolutionary approach.

6 Conclusions

In this paper we described a novel hybrid approach for solving multi-class problems in supervised, semi-supervised and unsupervised domains. The system which we call GEML, combines elements of decision tree logic and clustering techniques and incorporates these into a flexible grammatical evolution framework.

We have described the GEML framework in detail together with a set of experiments comparing GEML with several other state of the art ML algorithms. Results of these experiments were presented and discussed and we noted that the proposed system delivered competitive and generalizable accuracy which was shown to be statistically significant on all of the problems studied.

Our initial ensemble approaches have delivered encouraging results and in future work we may investigate other strategies which may be used to optimise the various parameters including the threshold values used both to determine which models to include in the original ensemble and which correlated models to exclude. Finally, it may be interesting to determine if improved classification accuracy may be achieved through creating a master ensemble from the separate ensembles generated from each GP run.

Acknowledgement. We gratefully acknowledge the support of Science Foundation Ireland. Grant number 10/IN.1/I3031.

References

1. Al-Madi, N., Ludwig, S.A.: Improving genetic programming classification for binary and multiclass datasets. In: Hammer, B., Zhou, Z.H., Wang, L., Chawla, N. (eds.) IEEE Symposium on Computational Intelligence and Data Mining, CIDM 2013. pp. 166–173. Singapore, 16–19 April 2013
2. Altman, N.S.: An introduction to kernel and nearest-neighbor nonparametric regression. Am. Stat. **46**(3), 175–185 (1992)
3. Azad, R.M.A., Ryan, C.: The best things don't always come in small packages: constant creation in grammatical evolution. In: Nicolau, M., Krawiec, K., Heywood, M.I., Castelli, M., García-Sánchez, P., Merelo, J.J., Rivas Santos, V.M., Sim, K. (eds.) EuroGP 2014. LNCS, vol. 8599, pp. 186–197. Springer, Heidelberg (2014). doi:10.1007/978-3-662-44303-3_16
4. Banzhaf, W.: Evolutionary computation and genetic programming. In: Lakhtakia, A., Martin-Palma, R.J. (eds.) Engineered Biomimicry, chap. 17, pp. 429–447. Elsevier, Boston (2013). http://www.sciencedirect.com/science/article/pii/B9780124159952000179
5. Barros, R.C., Basgalupp, M.P., De Carvalho, A.C., Freitas, A., et al.: A survey of evolutionary algorithms for decision-tree induction. IEEE Trans. Syst. Man Cybern. Part C Appl. Rev. **42**(3), 291–312 (2012)

6. Belhassen, S., Zaidi, H.: A novel fuzzy c-means algorithm for unsupervised heterogeneous tumor quantification in pet. Med. Phy. **37**(3), 1309–1324 (2010)
7. Bhowan, U., Johnston, M., Zhang, M.: Developing new fitness functions in genetic programming for classification with unbalanced data. IEEE Trans. Syst. Man Cybern. Part B Cybern. **42**(2), 406–421 (2012)
8. Boser, B.E., Guyon, I.M., Vapnik, V.N.: A training algorithm for optimal margin classifiers. In: Proceedings of the Fifth Annual Workshop on Computational Learning Theory, pp. 144–152. ACM (1992)
9. Breiman, L.: Bagging predictors. In: Machine Learning, pp. 123–140 (1996)
10. Breiman, L., Friedman, J., Stone, C.J., Olshen, R.A.: Classification and Regression Trees. CRC Press, New York (1984)
11. Castelli, M., Silva, S., Vanneschi, L., Cabral, A., Vasconcelos, M.J., Catarino, L., Carreiras, J.M.B.: Land cover/land use multiclass classification using GP with geometric semantic operators. In: Esparcia-Alcázar, A.I. (ed.) EvoApplications 2013. LNCS, vol. 7835, pp. 334–343. Springer, Heidelberg (2013). doi:10.1007/978-3-642-37192-9_34
12. Cowgill, M.C., Harvey, R.J., Watson, L.T.: A genetic algorithm approach to cluster analysis. Comput. Math. Appl. **37**(7), 99–108 (1999)
13. Deodhar, S., Motsinger-Reif, A.: Grammatical evolution decision trees for detecting gene-gene interactions. In: Pizzuti, C., Ritchie, M.D., Giacobini, M. (eds.) EvoBIO 2010. LNCS, vol. 6023, pp. 98–109. Springer, Heidelberg (2010). doi:10.1007/978-3-642-12211-8_9
14. Dietterich, T.: Ensemble methods in machine learning. In: Maimon, O., Rokach, L. (eds.) MCS 2000. LNCS, vol. 1857, pp. 1–15. Springer, Heidelberg (2000)
15. Dietterich, T.G., Bakiri, G.: Solving multiclass learning problems via error-correcting output codes. J. Artif. Intell. Res. **2**(1), 263–286 (1995)
16. Downey, C., Zhang, M., Liu, J.: Parallel linear genetic programming for multi-class classification. Genet. Programm. Evolvable Mach. **13**(3), 275–304 (2013). Special issue on selected papers from the 2011 European conference on genetic programming
17. Fitzgerald, J., Azad, R.M.A., Ryan, C.: GEML: Evolutionary unsupervised and semi-supervised learning of multi-class classification with grammatical evolution. In: Rosa, A., Merelo, J.J., Dourado, A., Cadenas, J.M., Madani, K., Ruano, A., Filipe, J. (eds.) ECTA. 7th International Conference on Evolutionary Computation Theory and Practice, paper 31. SCITEPRESS - Science and Technology Publications, Lisbon, Portugal, 12–14 November 2015
18. Fitzgerald, J., Ryan, C.: A hybrid approach to the problem of class imbalance. In: Matousek, R. (ed.) 19th International Conference on Soft Computing, MENDEL 2013, pp. 129–137, Brno, Czech Republic, 26–28 June 2013
19. Fogel, D.B.: What is evolutionary computation? IEEE Spectr. **37**(2), 26–28 (2000)
20. Freund, Y., Schapire, R.E., et al.: Experiments with a new boosting algorithm. In: ICML, vol. 96, pp. 148–156 (1996)
21. Fu, W., Johnston, M., Zhang, M.: Unsupervised learning for edge detection using genetic programming. In: Coello, C.A.C. (ed.) Proceedings of the 2014 IEEE Congress on Evolutionary Computation, pp. 117–124, Beijing, China, 6–11 July 2014
22. Greene, D., Tsymbal, A., Bolshakova, N., Cunningham, P.: Ensemble clustering in medical diagnostics. In: 17th IEEE Symposium on Computer-Based Medical Systems, CBMS 2004, Proceedings, pp. 576–581. IEEE (2004)

23. Hruschka, E.R., Campello, R.J., Freitas, A., De Carvalho, A.C., et al.: A survey of evolutionary algorithms for clustering. IEEE Trans. Syst. Man Cybern. Part C: appl. Rev. **39**(2), 133–155 (2009)
24. Ji, C., Ma, S.: Combinations of weak classifiers. IEEE Trans. Neural Netw. **8**(1), 32–42 (1997)
25. Kattan, A., Agapitos, A., Poli, R.: Unsupervised problem decomposition using genetic programming. In: Esparcia-Alcázar, A.I., Ekárt, A., Silva, S., Dignum, S., Uyar, A.Ş. (eds.) EuroGP 2010. LNCS, vol. 6021, pp. 122–133. Springer, Heidelberg (2010). doi:10.1007/978-3-642-12148-7_11
26. Kattan, A., Fatima, S., Arif, M.: Time-series event-based prediction: an unsupervised learning framework based on genetic programming. Inf. Sci. **301**, 99–123 (2015). http://www.sciencedirect.com/science/article/pii/S0020025515000067
27. Keijzer, M., Babovic, V.: Genetic programming, ensemble methods and the bias/variance tradeoff – introductory investigations. In: Poli, R., Banzhaf, W., Langdon, W.B., Miller, J., Nordin, P., Fogarty, T.C. (eds.) EuroGP 2000. LNCS, vol. 1802, pp. 76–90. Springer, Heidelberg (2000). doi:10.1007/978-3-540-46239-2_6
28. Kim, Y., Street, W.N., Menczer, F.: Feature selection in unsupervised learning via evolutionary search. In: Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 365–369. ACM (2000)
29. Koza, J.R.: Genetic programming: a paradigm for genetically breeding populations of computer programs to solve problems. Technical report (1990)
30. Maulik, U., Bandyopadhyay, S.: Genetic algorithm-based clustering technique. Pattern Recogn. **33**(9), 1455–1465 (2000)
31. Mierswa, I., Wurst, M.: Information preserving multi-objective feature selection for unsupervised learning. In: Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation, pp. 1545–1552. ACM (2006)
32. Mojsilović, A., Popović, M.V., Nešković, A.N., Popović, A.D.: Wavelet image extension for analysis and classification of infarcted myocardial tissue. IEEE Trans. Biomed. Eng. **44**(9), 856–866 (1997)
33. Morita, M., Sabourin, R., Bortolozzi, F., Suen, C.Y.: Unsupervised feature selection using multi-objective genetic algorithms for handwritten word recognition. In: 2013 12th International Conference on Document Analysis and Recognition, vol. 2, pp. 666–666. IEEE Computer Society (2003)
34. Muñoz, L., Silva, S., Trujillo, L.: M3GP – multiclass classification with GP. In: Machado, P., Heywood, M.I., McDermott, J., Castelli, M., García-Sánchez, P., Burelli, P., Risi, S., Sim, K. (eds.) EuroGP 2015. LNCS, vol. 9025, pp. 78–91. Springer, Heidelberg (2015). doi:10.1007/978-3-319-16501-1_7
35. Neshatian, K., Zhang, M.: Unsupervised elimination of redundant features using genetic programming. In: Nicholson, A., Li, X. (eds.) AI 2009. LNCS (LNAI), vol. 5866, pp. 432–442. Springer, Heidelberg (2009). doi:10.1007/978-3-642-10439-8_44
36. Omran, M.G., Engelbrecht, A.P., Salman, A.: Differential evolution methods for unsupervised image classification. In: The 2005 IEEE Congress on Evolutionary Computation, vol. 2, pp. 966–973. IEEE (2005)
37. O'Neill, M., Brabazon, A.: Grammatical differential evolution. In: Arabnia, H.R. (ed.) Proceedings of the 2006 International Conference on Artificial Intelligence, ICAI 2006, vol. 1, pp. 231–236, CSREA Press, Las Vegas, Nevada, USA, 26-29 June 2006. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.91.3012
38. O'Neill, M., Brabazon, A.: Self-organizing swarm (SOSwarm): a particle swarm algorithm for unsupervised learning. In: IEEE Congress on Evolutionary Computation, CEC 2006, pp. 634–639. IEEE (2006)

39. O'Neill, M., Leahy, F., Brabazon, A.: Grammatical swarm: a variable-length particle swarm algorithm. In: Nedjah, N., de Macedo Mourelle, L. (eds.) Swarm Intelligent Systems, Studies in Computational Intelligence, vol. 28, pp. 59–74. Springer, Heidelberg (2006) Chap. 5
40. O'Neill, M., Ryan, C.: Automatic generation of programs with grammatical evolution. In: Bridge, D., Byrne, R., O'Sullivan, B., Prestwich, S., Sorensen, H. (eds.) Artificial Intelligence and Cognitive Science AICS 1999, No. 10, University College Cork, Ireland, 1–3 September 1999. http://ncra.ucd.ie/papers/aics99.ps.gz
41. Pan, H., Zhu, J., Han, D.: Genetic algorithms applied to multi-class clustering for gene expression data. Bioinf. Genom. Proteomics **1**(4), 279–287 (2003)
42. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: machine learning in Python. J. Mach. Learn. Res. **12**, 2825–2830 (2011)
43. Raghavan, U.N., Albert, R., Kumara, S.: Near linear time algorithm to detect community structures in large-scale networks. Phys. Rev. E **76**(3), 036106 (2007)
44. Ren, Y., Zhang, L., Suganthan, P.: Ensemble classification and regression-recent developments, applications and future directions. IEEE Comput. Intell. Mag. **11**(1), 41–53 (2016)
45. Ryan, C., O'Neill, M.: How to do anything with grammars. In: Barry, A.M. (ed.) GECCO 2002: Proceedings of the Bird of a Feather Workshops, Genetic and Evolutionary Computation Conference. pp. 116–119. AAAI, New York, 8 Jul 2002. http://www.grammatical-evolution.org/gews2002/howto.ps
46. Schapire, R.E.: The strength of weak learnability. Mach. Learn. **5**(2), 197–227 (1990)
47. Smart, W., Zhang, M.: Probability based genetic programming for multiclass object classification. Technical report, CS-TR-04-7, Computer Science, Victoria University of Wellington, New Zealand (2004). http://www.mcs.vuw.ac.nz/comp/Publications/archive/CS-TR-04/CS-TR-04-7.pdf
48. Smart, W., Zhang, M.: Using genetic programming for multiclass classification by simultaneously solving component binary classification problems. In: Keijzer, M., Tettamanzi, A., Collet, P., Hemert, J., Tomassini, M. (eds.) EuroGP 2005. LNCS, vol. 3447, pp. 227–239. Springer, Heidelberg (2005). doi:10.1007/978-3-540-31989-4_20
49. Steinhaus, H.: Sur la division des corps matériels en parties. Bull. Acad. Pol. Sci. Cl. III **4**, 801–804 (1957)
50. Wu, S.X., Banzhaf, W.: Rethinking multilevel selection in genetic programming. In: Krasnogor, N., Lanzi, P.L., Engelbrecht, A., Pelta, D., Gershenson, C., Squillero, G., Freitas, A., Ritchie, M., Preuss, M., Gagne, C., Ong, Y.S., Raidl, G., Gallager, M., Lozano, J., Coello-Coello, C., Silva, D.L., Hansen, N., Meyer-Nieberg, S., Smith, J., Eiben, G., Bernado-Mansilla, E., Browne, W., Spector, L., Yu, T., Clune, J., Hornby, G., Wong, M.L., Collet, P., Gustafson, S., Watson, J.P., Sipper, M., Poulding, S., Ochoa, G., Schoenauer, M., Witt, C., Auger, A. (eds.) GECCO 2011: Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation, pp. 1403–1410. ACM, Dublin, Ireland, 12–16 July 2011, best paper
51. Zhang, M., Smart, W.: Multiclass object classification using genetic programming. In: Raidl, G.R., Cagnoni, S., Branke, J., Corne, D.W., Drechsler, R., Jin, Y., Johnson, C.G., Machado, P., Marchiori, E., Rothlauf, F., Smith, G.D., Squillero, G. (eds.) EvoWorkshops 2004. LNCS, vol. 3005, pp. 369–378. Springer, Heidelberg (2004). doi:10.1007/978-3-540-24653-4_38