Maximum Margin Decision Surfaces for Increased Generalisation in Evolutionary Decision Tree Learning

Alexandros Agapitos, Michael O'Neill, Anthony Brabazon

Financial Mathematics and Computation Research Cluster
Natural Computing Research and Applications Group
Complex and Adaptive Systems Laboratory
University College Dublin, Ireland

Abstract

Decision tree learning is one of the most widely used and practical methods for inductive inference. We present a novel method that increases the generalisation of genetically-induced classification trees, which employ linear discriminants as the partitioning function at each internal node. Genetic Programming is employed to search the space of oblique decision trees. At the end of the evolutionary run, a (1+1) Evolution Strategy is used to geometrically optimise the boundaries in the decision space, which are represented by the linear discriminant functions. The evolutionary optimisation concerns maximising the decision-surface margin that is defined to be the smallest distance between the decision-surface and any of the samples. Initial empirical results of the application of our method to a series of datasets from the UCI repository suggest that model generalisation benefits from the margin maximisation, and that the new method is a very competent approach to pattern classification as compared to other learning algorithms.

1 Introduction

This paper introduces a novel, hybrid approach to the evolutionary learning of Decision Trees (DT), by means of Genetic Programming (GP) [1], that improves their generalisation performance. The evolutionary method is based on the concept of maximum margin linear discriminant functions to search among a number of potential decision surfaces for the one for which the margin is maximised. This concept is borrowed from the Support Vector Machine (SVM) [2] approach to overfitting avoidance. The goal of training a SVM is to find the separating hyperplane with the largest margin; it is expected that the larger

the margin, the better generalisation of the classifier. Initially, oblique DTs are learned by means of GP using the classification accuracy as the fitness measure. At the end of the evolutionary run the best classifier is further optimised to improve its generalisation. The optimisation concerns maximising the margin, which is defined to be the smallest distance between the decision surface and any of the samples. Finding the maximally-separating decision hyperplane is a constrained optimisation problem that is tackled by means of quadratic programming in the case of SVMs. Our method is instead using an evolutionary optimisation algorithm (EA), namely a (1+1) Evolution Strategy (ES). The proposed methodology is compared in this study against the axis-parallel DT induction method C4.5, SVM (without any use of kernel functions), and naive Bayesian on a series of benchmark classification datasets from the UCI machine learning repository. Initial empirical results suggest that increased generalisation is accrued from margin maximisation, and that the new approach compares favourably, often outperforming other learning algorithms.

The rest of the paper is organised as follows. We first present relevant literature for inducing DTs using GP, and discuss the issue of model generalisation, motivating the need for maximum margin classifiers. The geometry of linear discriminant function is analysed in Section 2, which defines the main tool for calculating distances in the feature space. Section 3 presents the margin optimisation algorithm that is based on a novel distance-based fitness function. Section 4 outlines the experimental approach, describes the grammar-based GP system, and details the benchmark datasets. Section 5 presents the experimental results, and Section 6 draws our conclusions.

1.1 Model Generalisation in Genetically Programmed DTs

Genetic Programming, inherently adopting an expression-tree representation, has been an obvious choice for an application of stochastic search to the space of decision-tree structures, seeking to maximise some sort of classification performance metric. GP-induced partitioning functions can be composed of both linear and non-linear combinations of features, therefore oblique and non-linear splits can be represented as easily and efficiently as univariate splits. In the domain of axis-parallel decision trees the works of [3, 4, 5] evolve DTs using classification accuracy as the fitness function to drive the search. Oblique decision trees have been evolutionarily constructed in [6, 7], whereas work on non-linear DTs has been reported in [8, 9, 10].

A crucial aspect of artificial learning systems is their ability to extract a precise underlying representation of the concept that is being inferred in a supervised learning task via a set of training instances, so as to be able to generalise well to unseen examples from that concept. Decision tree learning, which is based on adaptive tree-structures that are being iteratively refined on a set of training instances, suffers from the problem of model overfitting inherent in any process of data-driven modelling. The highly expressive representation offered by tree-structures often results in a close fit on the training instances that does not allow for effective generalisation if their complexity is not somehow kept un-

der control. An approximate inductive bias of classical DT growing algorithms is that shorter trees are preferred over larger ones [11]. Dominant approaches to control the size of a DT are reduced-error pruning and rule post-pruning [11]. In GP-based DT induction, research on improving model generalisation has focused on controlling the size of the evolved structures mainly by modifying the fitness function to exhibit a bias towards smaller expression-trees [12, 13, 14, 15, 10]. A different approach to the problem of overfitting in genetically-induced DTs is presented in [16]. In that study, a statistical significance test of each program's performance is employed, and multi-objective fitness functions are designed to bias the evolutionary search towards better generalising individuals.

The generalisation performance of a learning machine can be studied by means of uniform convergence bounds, using a technique introduced by Vapnik and Chervonenkis [17]. The theoretical motivations behind this approach lie in the data-dependent *structural risk minimisation* (SRM) [18] principle. The central concept is the *effective capacity* of the class of hypotheses accessible by the machine: the richer such a class, the higher the risk of overfitting. This feature of a learning machine is referred to as its complexity or *capacity*. The statistical learning principle of SRM provides an upper bound to the generalisation error of the classifier in terms of its training error, the number of training examples and the model capacity. In this regard, SRM is just another way to express generalisation error as a tradeoff between training error and model complexity. In the case where a linear discriminant function in trained on a linearly-separable dataset, there exist an indefinite number of partitioning hyperplanes that attain a perfect split. However, there is no guarantee that any of these hyperplanes will generalise well on new patterns. Maximum margin classifiers approach the decision surface generalisation-potential through the concept of the *margin*, which is defined to be the smallest distance between the decision surface and any of the samples. In [2], it is shown that the capacity of a linear model is inversely related to its margin. Models with small margins have higher capacities because they are more flexible and can fit many training sets, unlike models with large margins. However, according to the SRM principle, as the capacity increases, the generalisation error bound will also increase. Therefore, it is desired to design linear discriminant functions that maximise the margins of their decision boundaries in order to ensure that their worst-case generalisation errors are minimised. The state-of-the-art maximum margin classifier is the support vector machine (SVM) [2].

2 Geometry of a Linear Discriminant Function

Oblique splits are essentially represented by linear discriminant functions of the form $y(x) = w^T x + w_0$, where w is called the weight vector, and w_0 is a bias. For this class of discriminant functions, the decision surfaces are $(D-1)$-dimensional hyperplanes within the D-dimensional feature space. An oblique DT can be therefore regarded as a collection of linear discriminants instrumented in such a way so as to provide a classification technique for both binary and

multi-category pattern classification tasks.

The simplest representation of a linear discriminant function is obtained by taking a linear function of the input vector x so that $y(x) = w^T x + w_0 = \sum_{i=1}^{D} w_i x_i + w_0$, where w^T is the transpose of the weight vector, w_0 is a bias, and $w^T x$ represent the inner product of vectors w^T and x. Assuming a binary classification task, an input vector x is assigned to class C_1 if $y(x) > 0$, and to C_2 otherwise. The corresponding decision surface is defined by the expression $y(x) = w^T x + w_0 = 0$, which corresponds to a $(D-1)$-dimensional hyperplane within the D-dimensional input space. Consider two points x_A and x_B that lie on the hyperplane. Because $y(x_A) = y(x_B) = 0$ we have:

$$\begin{aligned} w^T x_A + w_0 = w^T x_B + w_0 = 0 \quad \Rightarrow \\ w^T (x_A - x_B) = 0 \end{aligned} \tag{1}$$

and hence the vector w is perpendicular to every vector lying within the decision surface, given that their inner product is equal to zero. Thus, vector w determines the orientation of the decision surface. Similarly, if x is a point on the decision surface, then $y(x) = 0$, and so the normal distance from the origin to the decision surface is given by the following:

$$\frac{w^T x}{||w||} = -\frac{w_0}{||w||} \tag{2}$$

where $||w||$ represents the magnitude or Euclidean norm of vector w. We therefore see that the bias w_0 determines the displacement of the decision surface from the axis origin. These properties are illustrated for the case of $D = 2$ in Figure 1(a).

Furthermore, we note that the value of $y(x)$ gives a signed measure of the perpendicular distance r of the point x from the decision surface. To illustrate this, consider an arbitrary point x, and let $x_\perp$ be its orthogonal projection onto the decision surface. The perpendicular distance r of x from the decision surface is given as follows:

$$\begin{aligned} x = x_\perp + r\frac{w}{||x||} \quad \Rightarrow \\ wx + w_0 = wx_\perp + r\frac{w}{||x||}w + w_0 \end{aligned} \tag{3}$$

Given that $y(x) = w^T x + w_0 = 0$ and $y(x_\perp) = w^T x_\perp + w_0 = 0$, Equation 3 becomes:

$$\begin{aligned} y(x) = y(x_\perp) + r\frac{w}{||x||}w \quad \Rightarrow \\ y(x) = r\frac{w}{||x||}w \end{aligned} \tag{4}$$

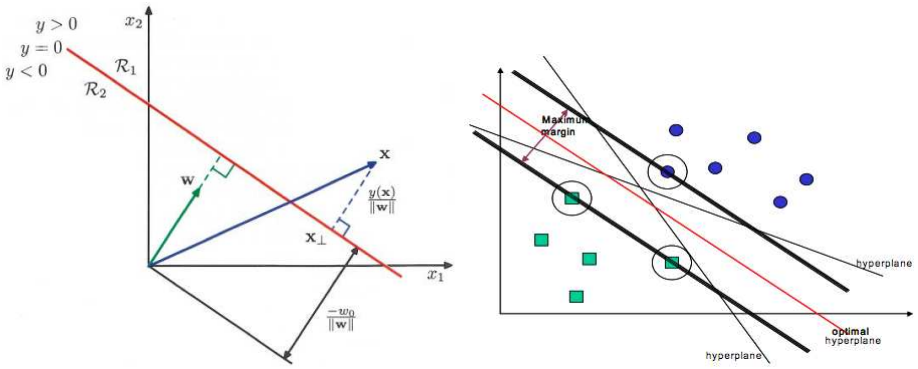
$$\tag{5}$$

Given that the inner product of two vectors a and b is given by $a \cdot b = ||a||||b||cos\theta$, where θ is a measure of the angle between a and b, and that the angle between vector w an itself is zero, we have:

$$y(x) = \frac{r}{||w||}||w||||w||cos(0) \quad \Rightarrow$$
$$r = \frac{y(x)}{||w||} \tag{6}$$

This result is illustrated in Figure 1(a). For this case where $D = 2$, the absolute distance r of point $x = [x_1, x_2]$ from the decision hyperplane $y(x)$ with $w = [w_1, w_2]$ becomes:

$$r = \frac{|y(x)|}{||w||} = \frac{|w_1 x_1 + w_2 x_2 + w_0|}{\sqrt{{w_1}^2 + {w_2}^2}} \tag{7}$$



(a) (b)

Figure 1: (a) Illustration of the geometry of a linear discriminant function in two dimensions. The decision surface, shown in red, is perpendicular to w, thus its orientation depends on it, whereas its displacement from the axis origin is controlled by the bias parameter w_0. The signed orthogonal distance of a general point x from the decision surface is given by $y(x)/||w||$. Figure taken from [19] (page 182). (b) Possible decision surfaces for a linearly separable dataset. The margin is defined as the perpendicular distance between the decision surface and the closest data points. Maximising the margin leads to a particular choice of decision boundary, as shown with the red line. The location of the boundary is determined by a subset of the data points, which is indicated by the circles.

3 Evolutionary Optimisation of a Maximum Margin Decision Surface via an ES(1+1)

Decision boundaries with large margins tend to have lower generalisation errors than those with small margins. Intuitively, if the margin is small, then any slight perturbations to the decision boundary can have quite a significant impact on its classification. Classifiers that produce decision boundaries with small margins are therefore susceptible to model ovefitting. Figure 1(b) illustrate various decision hyperplanes for a linearly separable dataset. It should be clear that in this case the solution region incorporates a number of hyperplanes, with arbitrary orientation and displacement, that satisfy the requirement of providing a perfect classification on the training patterns. In the example of Figure 1(b), we illustrate three possible hyperplanes that would result in a 100% accurate classifier. One way to impose additional requirements in order to bias the selected hyperplane towards better generalisation is to introduce the concept of margin, which is defined as the perpendicular distance between the decision surface and the closest data points. Given this definition, we observe that if we wish to maximise this margin, the location of the required hyperplane is determined by a subset of the closest data points, which is indicated by the circles in Figure 1(b).

We adopted an evolutionary approach to optimise the margin of a decision surface using an ES(1+1) without self-adaptive mutations. Margin maximisation can be regarded as a constrained optimisation problem that requires the hyperplane to freely wander within the solution region of arbitrarily oriented hyperplanes for the one with optimal margin, without causing any drop in the classification accuracy measured originally on the training instances. We transform the constrained optimisation problem into a multi-objective problem by adding a penalty function that evaluates constraint violations realised through misclassifications. So far, we have assumed that the training data points are linearly separable in the feature space. In practice, however, the class-conditional distributions may overlap, in which case exact linear separation of the training data may not be possible, and if finally some kind of separation does succeed it usually leads to poor generalisation. We therefore need a way to allow some of the training points to be misclassified. Data points are allowed to be on the "wrong" side of the margin boundary, but with a penalty that increases proportionally to the distance from the boundary. A distinguishing characteristic of maximum margin classification approaches is that these learning algorithms have *sparse* solutions so that the location of the decision surface is determined by a subset of data points.

In order to quantify the size of the margin we start by determining the closest pair of data points (in terms of Euclidean distance), where each data point belongs to a different class and lies on the correct side of the margin. Then the distance of these points from the hyperplane is calculated using Equation 6; let us call these distances as cpd_{classA} and cpd_{classB}. We also use the same equation to calculate the distances between each correctly-classified data point

and the hyperplane, for both classes; let us call the smallest of these distances as sv_{classA} and sv_{classB} for data points of classes A and B respectively. The distance-based objective function that quantifies a hyperplane's margin size is defined as follows:

$$f = \left[\frac{\mu_{cpd}}{\sigma_{cpd}} + sv_{classA} + sv_{classB}\right] - \frac{1}{N}\sum_{i=1}^{N} penalty(x_i) \tag{8}$$

where μ_{cpd} is the mean distance of cpd_{classA} and cpd_{classB} distances, σ_{cpd} is their standard deviation, N is the total number of points for classes A and B , and penalty(x) is defined as:

$$penalty(x) = \begin{cases} \frac{|y(x)|}{||w||} & \text{if } predicted \neq actual \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

Maximisation of this objective function leads to maximisation of the hyperplane's margin. The first compartment of this function requires that the fraction $\frac{\mu_{cpd}}{\sigma_{cpd}}$ is maximised along with the distances sv_{classA} and sv_{classB}, leading to a margin that maximises the minimum distance of the samples from the surface, while ensuring that the separating hyperplane is equally spaced in-between those closest samples. At the same time, the amount of misclassification penalties is required to be kept at a minimum by the second compartment.

The first step in optimising the margin of the decision boundaries represented by the linear discriminant functions serving as predicate nodes in a DT, is to analyse the expression-tree and map each of the linear discriminant functions to pairs of classes that they discriminate. For that purpose we instrument the fitness evaluation process in order to be able to track down the path that is being followed during a DT's execution with an input training pattern. For each fitness evaluation, we log the execution-trace by recording each linear discriminant that has been visited starting from the root node until we reach a final classification leaf-node. Having mapped the linear-discriminant functions to possible classifications, we then create all possible classification pairs that a linear discriminant attempts to partition. For every such classification pair we determine the size separating hyperplane's margin size using the objective function in Equation 8. The size of the margin for the overall DT is given by averaging the resulting margin sizes of all linear discriminants that compose the expression-tree.

We have chosen an ES(1+1), a simple but powerful random hill-climbing method to search the solution region of a separating hyperplane for the maximum possible margin. The candidate hyperplane is represented as a vector of coefficients that combines the components of vector w (determining the orientation) with the bias w_0 (determining the displacement from origin). The number of optimisation iterations are governed by a user-defined parameter. At each iteration a coefficient or a number of coefficients are being perturbed using a Gaussian mutation with known mean and standard deviation. Currently, we do

Table 1: Benchmarking Datasets

Dataset	Size	No. of Classes	No. of numerical pattern features
Wisconsin Breast Cancer (WBC)	699	2	9
BUPA liver disorders	345	2	6
PIMA diabetes	768	2	8
IRIS	150	3	4
Vehicle	846	4	18

not allow for self-adaptation of the mutation rates. The sequence in which coefficients are perturbed, as well as their number is either deterministic or random, with a real-valued parameter governing the probability of application of these two coefficient-examination schemes, set to 0.7 is favour of the deterministic sequential examination.

4 Methods

4.1 Experimental Approach

This empirical study attempts to hybridise the traditional method of constructing oblique DTs using GP, with the approach taken in maximum margin classification methodologies. The concept of learning revolves around the notion of generalisation, and it is highly desired that methods reinforcing model generalisation are extensively studied in the scope of evolutionary learning. We are contrasting four different learning algorithms in terms of their generalisation ability in a series of five benchmark classification datasets from the UCI machine learning repository [20]. These algorithms are: (a) GP hybridised with margin maximisation (GP-MM), (b) standard GP (GP-ST), (c) axis-parallel DT inductor C4.5, (d) SVM (without any use of kernel functions), and (e) naive Bayesian, all taken from WEKA software [21]. Performance statistics are evaluated through the process of 10-fold cross validation. Table 1 presents a description of the benchmarking datasets. Prior to classification all numeric features were normalised within the range of [0.0, 1.0].

4.2 Grammar-based Genetic Programming

We employ a grammar-based GP system to evolve DTs. The context-free grammar that is used to type the DT representation language is presented bellow:

```
<DT> ::= <if>
<if> ::= <ldf> <expr> <expr>
<expr> ::= <if> | <classification>
<classification> ::= classA | classB | ... | classZ
<ldf> ::= <constant> * <feature> + <constant> * <feature> + <constant> > 0
<constant> ::= <constant> <op> <constant> | double in the range of [-1.0, 1.0]
<op> ::= + | - | * | /
<feature> ::= depending on the number of features in the problem
```

Using this grammar, a DT is represented as a set of two-dimensional linear discriminant functions. We deliberately constrained the system to the induction of two-dimensional linear discriminants in order to study the effectiveness of the proposed methodology on the simplest representation setting possible. Future extensions will employ a more expressive representation that allows the composition of multi-dimensional separating hyperplanes.

The GP algorithm employs a panmictic, generational, elitist genetic algorithm. The algorithm uses tournament selection with a tournament size of 12. The population size is set to $1,000$ individuals. Ramped-half-and-half tree creation with a maximum depth of 5 is used to perform a random sampling of DTs during run initialisation. Throughout evolution, expression-trees are allowed to grow up to depth of 8. The evolutionary search employs a mutation-based variation scheme, where subtree mutation is combined with point-mutation; a probability governing the application of each, set to 0.6 in favour of subtree mutation. No reproduction nor recombination were used.

In the case where margin maximisation is included in the learning process (GP-MM), GP is run for 100 generations using classification accuracy as the fitness function (phase I), and optimisation proceeds for an additional $10,000$ hill-climbing iterations (equally divided among the linear discriminants of a DT) using the objective function of Equation 8 (phase II), totalling $110,000$ fitness evaluations. The Gaussian mutation in ES(1+1) has a mean of 0.0 and a standard deviation of 0.01. In the case of standard GP runs (GP-ST), evolution proceeds for 110 generations to allow for a comparison on the same number of fitness evaluations. Classification accuracy is similarly used as a fitness function.

5 Results

For the evolutionary learning algorithms we performed 50 independent runs to calculate performance statistics. Table 2 summarises the out-of-sample classification performance accrued from the process of 10-fold cross validation. In the case of evolutionary algorithms we also present the average accuracy for phases I and II, corresponding to the values obtained during GP and ES(1+1) procedures respectively. Recall that for the case of GP-MM phase I lasts for 100 generations, whereas for GP-ST it lasts for 110 generations. Results suggest a superiority of evolutionary methods compared to other learning algorithms, which becomes more pronounced in the BUPA and VEHICLES datasets, with the EAs largely outperforming the rest of the classifiers. A comparison between GP-MM and GP-ST, further suggests that margin maximisation does indeed play an important role in increasing the model generalisation performance. The percentage of increase in model generalisation ranges from 2.8% in WBC dataset to 13.0%, suggesting that there are cases where margin maximisation is a very effective method to deal with the problem of model overfitting. The circumstances under which such an additional optimisation is fruitful remains to be seen from the application of our technique to a wider range of multi-category classification problems, by extending the expressiveness of linear discriminant function rep-

Table 2: Comparison of test classification accuracy obtained by different learning algorithms in 10-fold cross validation. Standard errors in parentheses.

Dataset	Learning algorithm	Best accuracy	Average accuracy (phase I)	Average accuracy (phase II)	Generalisation increase
WBC	GP-MM	98.5%	94.6% (0.008)	97.5% (0.006)	2.8% (0.004)
	GP-ST	97.9%	95.1% (0.008)	-	-
	C4.5	96.1%	-	-	-
	SVM	96.6%	-	-	-
	Bayesian	97.4%	-	-	-
BUPA	GP-MM	81.3%	65.8% (0.01)	73.8% (0.01)	13.0% (0.02)
	GP-ST	71.8%	66.1% (0.01)	-	-
	C4.5	63.1%	-	-	-
	SVM	70.3%	-	-	-
	Bayesian	63.2%	-	-	-
PIMA	GP-MM	81.8%	75.3% (0.008)	78.7% (0.007)	4.8% (0.005)
	GP-ST	77.6%	74.9% (0.008)	-	-
	C4.5	75.5%	-	-	-
	SVM	78.1%	-	-	-
	Bayesian	75.0%	-	-	-
IRIS	GP-MM	100%	95.6% (0.01)	97.7% (0.008)	2.3% (0.004)
	GP-ST	98.2%	95.9% (0.01)	-	-
	C4.5	95.3%	-	-	-
	SVM	97.1%	-	-	-
	Bayesian	94.0%	-	-	-
VEHICLE	GP-MM	78.0%	66.7% (0.008)	71.2% (0.008)	6.7% (0.003)
	GP-ST	72.4%	66.3 % (0.007)	-	-
	C4.5	72.6%	-	-	-
	SVM	49.6%	-	-	-
	Bayesian	60.5%	-	-	-

resentation, allowing for separating hyperplanes in a much higher-dimensional feature space. An additional interesting observation in Table 2 is that we found no significant differences between the average classification accuracies in phase I of GP-MM and GP-ST, suggestive of no evident model overfitting from model overtraining. This is particularly interesting and surely warrants further study. Finally, we observe that for the four-class problem (VEHICLE), SVMs have a very low performance indicative of the inherent difficulty to tackle multi-class problems using this methodology. SVMs are traditionally binary classifiers, and practitioners need to rely on *one-versus-the-rest*, or *one-versus-one* binary problem decomposition strategies. On the other hand, tree-based structures can naturally represent multi-class problems.

6 Conclusion

The decision surface optimisation criterion revolves around the notion of a margin either side of a hyperplane that separates two data classes. Maximising

the margin and thereby creating the largest possible distance between the separating hyperplane and the instances on either side of it has been theoretically shown and practically proven by SVMs to reduce the upper bound on the expected generalisation error. This study applies the methodology of maximum margin classifiers to evolutionary search, resulting in a hybrid method that learns oblique DTs via GP, and subsequently optimises the geometry of decision hyperplanes using an ES(1+1). Results are very encouraging, suggesting a superiority of the new approach as compared against other learning algorithms. An interesting point to note is that at the moment the ES optimises the geometry of lines in a two-dimensional feature space. In addition, the solution uses the support of all patterns in the training set. Further research is required to quantify the time-efficiency of the proposed approach when dealing with large datasets and high-dimensional feature spaces.

A substantial advantage of evolutionary DT induction is the inherent ability of the learning algorithm to perform feature construction and/or feature selection, while simultaneously searching for a good classification strategy. Therefore, with a careful design of an EA, the dimensionality of the feature space can be made invariant of the classification performance. Feature spaces that are linearly non-separable can take advantage of built-in or evolved kernels combined with margin-maximisation, providing a natural approach to a multi-category classification methodology that embodies the best practices of the state-of-the-art methods in pattern classification.

Acknowledgement

This publication has emanated from research conducted with the financial support of Science Foundation Ireland under Grant Number 08/SRC/FM1389.

References

[1] J.R. Koza, *Genetic Programming: on the programming of computers by means of natural selection*, MIT Press, Cambridge, MA, (1992).

[2] Vapnik Vladimir, *The nature of statistical learning theory*, Springer Verlag, 2nd edition, 1999.

[3] John R. Koza, "Concept formation and decision tree induction using the genetic programming paradigm", in *Parallel Problem Solving from Nature - Proceedings of 1st Workshop, PPSN 1*, H.-P. Schwefel and R. Männer, Eds., Dortmund, Germany, 1-3 October 1991, vol. 496 of *Lecture Notes in Computer Science*, pp. 124–128, Springer-Verlag.

[4] Gianluigi Folino, Clara Pizzuti, and Giandomenico Spezzano, "Genetic programming and simulated annealing: A hybrid method to evolve decision trees", in *Genetic Programming, Proceedings of EuroGP'2000*, 2000.

[5] Jeroen Eggermont, "Evolving fuzzy decision trees with genetic programming and clustering", in *Genetic Programming, Proceedings of the 5th European Conference, EuroGP 2002*, Kinsale, Ireland, 3-5 April 2002, vol. 2278 of *LNCS*, pp. 71–82, Springer-Verlag.

[6] S. E. Rouwhorst and A. P. Engelbrecht, "Searching the forest: Using decision trees as building blocks for evolutionary search in classification databases", in *Proceedings of the 2000 Congress on Evolutionary Computation CEC00*, 2000, vol. 1, pp. 633–638.

[7] Martijn Bot and William B. Langdon, "Application of genetic programming to induction of linear classification trees", in *Proceedings of the Eleventh Belgium/Netherlands Conference on Artificial Intelligence (BNAIC'99)*, 1999.

[8] Robert E. Marmelstein and Gary B. Lamont, "Pattern classification using a hybrid genetic program decision tree approach", in *Genetic Programming 1998: Proceedings of the Third Annual Conference*, 1998.

[9] Athanasios Tsakonas, "A comparison of classification accuracy of four genetic programming-evolved intelligent structures", *Information Sciences*, vol. 176, no. 6, pp. 691–724, 22 March 2006.

[10] E. M. Mugambi, Andrew Hunter, Giles Oatley, and Lee Kennedy, "Polynomial-fuzzy decision tree structures for classifying medical data", *Knowledge-Based Systems*, vol. 17, no. 2-4, pp. 81–87, 2004.

[11] Tom Mitchel, *Machine Learning*, McGraw-Hill, 1997.

[12] Jesus K. Estrada-Gil, Juan C. Fernandez-Lopez, Enrique Hernandez-Lemus, Irma Silva-Zolezzi, Alfredo Hidalgo-Miranda, Gerardo Jimenez-Sanchez, and Edgar E. Vallejo-Clemente, "GPDTI: A genetic programming decision tree induction method to find epistatic effects in common complex diseases", *Bioinformatics*, vol. 13, no. 13, pp. i167–i174, 2007.

[13] Chan-Sheng Kuo, Tzung-Pei Hong, and Chuen-Lung Chen, "Applying genetic programming technique in classification trees", *Soft Computing*, vol. 11, no. 12, pp. 1165–1172, October 2007.

[14] Haruyama S and Qiangfu Zhao, "Designing smaller decision trees using multiple objective optimization based gps", in *IEEE International Conference on Systems, Man and Cybernetics*, 2002, vol. 6, p. 5.

[15] Gianluigi Folino, Clara Pizzuti, and Giandomenico Spezzano, "Improving induction decision trees with parallel genetic programming", in *Proceedings 10th Euromicro Workshop on Parallel, Distributed and Network-based Processing*, Canary Islands, 9-11 January 2002, pp. 181–187, IEEE.

[16] Alexandros Agapitos, Michael O'Neill, and Anthony Brabazon, "Evolutionary learning of technical trading rules without data-mining bias", in *PPSN 2010 11th International Conference on Parallel Problem Solving From Nature*, Robert Schaefer, Carlos Cotta, Joanna Kolodziej, and Guenter Rudolph, Eds., Krakow, Poland, 11-15 September 2010, vol. 6238 of *Lecture Notes in Computer Science*, pp. 294–303, Springer.

[17] V. N. Vapnik and A. Ya. Chervonenkis, "On the uniform convergence of relative frequencies of events to their probabilities", *Theory of Probability and its Applications*, vol. 16, no. 2, pp. 264–280, 1971.

[18] Shawe-Taylor J, Bartlett P.L, Williamson R.C., and Anthony M, "Structural risk minimization over data-dependent hierarchies", *IEEE Transactions on Information Theory*, vol. 44, no. 5, 1998.

[19] Christopher M. Bishop, *Pattern Recognition and Machine Learning*, Springer, 2006.

[20] D. J. Newman, S. Hettich, C. L. Blake, and C. J. Merz, "UCI repository of machine learning databases", 1998.

[21] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten, "The weka data mining software: an update", *SIGKDD Explor. Newsl.*, vol. 11, pp. 10–18, November 2009.