Expert Systems With Applications 137 (2019) 392–404

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Decision tree underfitting in mining of gene expression data. An evolutionary multi-test tree approach

Marcin Czajkowski[*], Marek Kretowski

Faculty of Computer Science, Bialystok University of Technology, Wiejska 45a, Bialystok 15-351, Poland

ARTICLE INFO

Article history:
Received 23 January 2019
Revised 28 June 2019
Accepted 9 July 2019
Available online 10 July 2019

Keywords:
Data mining
Evolutionary algorithms
Decision trees
Underfitting
Gene expression data

ABSTRACT

The problem of underfitting and overfitting in machine learning is often associated with a bias-variance trade-off. The underfitting most clearly manifests in the tree-based inducers when used to classify the gene expression data. To improve the generalization ability of decision trees, we are introducing an evolutionary, multi-test tree approach tailored to this specific application domain. The general idea is to apply gene clusters of varying size, which consist of functionally related genes in each splitting rule. It is achieved by using a few simple tests that mimic each other's predictions and built-in information about the discriminatory power of genes. The tendencies to underfit and overfit are limited by the multi-objective fitness function that minimizes tree error, split divergence and attribute costs. Evolutionary search for multi-tests in internal nodes, as well as the overall tree structure, is performed simultaneously.

This novel approach called Evolutionary Multi-Test Tree (EMTTree) may bring far-reaching benefits to the domain of molecular biology including biomarker discovery, finding new gene-gene interactions and high-quality prediction. Extensive experiments carried out on 35 publicly available gene expression datasets show that we managed to significantly improve the accuracy and stability of decision tree. Importantly, EMTTree does not substantially increase the overall complexity of the tree, so that the patterns in the predictive structures are kept comprehensible.

1. Introduction

In machine learning, generalization often refers to the ability of a predictive model to match unseen data (Hastie, Trevor, Tibshirani, Robert, Friedman, 2009). If the model matches the training set well, but fails to predict new instances in the problem area, we are typically dealing with so-called overfitting. This happens when the model concentrates on too much detailed information from the training data, which can occur in the form of noise or accidental fluctuations, which negatively affects the ability of the models to generalize. Underfitting is in opposition to overfitting as the underfitted model is not complicated enough and too little focuses on training data. As a result, it can neither fit the training set nor generalize new data well.

Decision trees (DT)s (Kotsiantis, 2013) are one of the main techniques for discriminant analysis in knowledge discovery. Due to their non-parametric and flexible algorithm, DTs are at some extent prone to overfitting (Loh, 2014; Sez, Luengo, & Herrera, 2016). There are also known to be instable, as small variations in the training set can result in different trees and non-repeatable predictions. While this is an unquestionable advantage when using multiple trees, it is a problem when a classifer based on a single tree is used. Both greater generalization ability and stability can be improved, for example, by learning multiple models from bootstrap samples of the training data, but such an ensemble approach makes the extracted knowledge less understandable.

This paper tackles the problem of underfitting of DT in the classification of gene expression data. In such data a ratio of features to observations is very high, which creates serious problems for the standard univariate decision trees (Chen, Wang, & Zhang, 2011; Czajkowski & Kretowski, 2014). The learning algorithms may find tests that perfectly separates the training data, but these splits often correspond to noise. This situation is more likely at intermediate and lower levels of the tree, where the number of instances is reduced with each tree level and may be several orders of magnitude smaller than the number of available features. For this reason, most of univariate DT inducers produce considerably simple trees that successfully classify the training data, but fail to classify unseen instances (Grzes & Kretowski, 2007). This may lead to underfitting as a small number of attributes is used in such trees and, therefore, their models are not complex enough and cause poor generalization (Hastie, Trevor, Tibshirani, Robert, Friedman, 2009).

* Corresponding author.
E-mail addresses: m.czajkowski@pb.edu.pl (M. Czajkowski), m.kretowski@pb.edu.pl (M. Kretowski).

https://doi.org/10.1016/j.eswa.2019.07.019
0957-4174/© 2019 Elsevier Ltd. All rights reserved.

The production of larger trees does not solve the problem, because in case of gene expression, small trees already classify the training data perfectly. This indicates that one can opt for the issue of split complexity, as little can be obtained from larger univariate DTs with this type of data.

A gene cluster is a part of a gene family, which is a set homologous genes within one organism. It is composed of two or more genes found within an organism's DNA that encode for similar polypeptides, or proteins, which collectively share a generalized function. It has been shown (Yi, Sze, & Thon, 2007) that polypeptides, or proteins are also encoded by a group of functionally related genes not a single one. In addition, the use of information on subgroups of attributes is particularly important in the problem of classification and selection of genomic data (Kar, Sharma, & Maitra, 2015; Wong & Liu, 2010). Therefore, we believe, that focusing on a tree split based on gene clusters rather than a single gene improves not only classifiers generalization ability but also provides interesting patterns that may appear in each multi-tests. This direction of research is continued in our study.

The main contribution of the work is a new evolutionary multi-test tree algorithm called Evolutionary Multi-Test Tree (EMTTree). It aims to improve single-tree classifiers in context of prediction accuracy and stability with a redefined and extended multi-test split approach (Czajkowski, Grześ, & Kretowski, 2014). In contrast to existing solutions, we propose a concept of a gene cluster in order to split instances in each non-terminal node of the tree. Each cluster consists of tests that mimic each other's predictions, and each test is an univariate test built on a selected attribute. Novelty of the EMTTree covers:

- an evolutionary tree induction as an alternative to the greedy top-down which was used in our previous works. Thanks to this global approach we were able to search for the tree structure and multi-test splits simultaneously, and resign from the flawed pruning procedure;
- a new algorithm for searching multi-test splits: specialized EA in a combination with local optimizations allows searching for most uniform multi-tests with the top-ranked genes;
- introducing gene cluster concept to the multi-test and adding a new dimension to its structure: information about the discriminatory power of genes is associated with every univariate test that constitutes a multi-test;
- a unique fitness function that focuses on minimizing the tree error, but not on the tree size, which is the standard procedure for DT. In addition, we incorporate information on gene ranking and resemblance of splits in order to prevent the predictor from underfitting and overfitting to data, especially in the lower parts of the tree.

An extensive set of computational experiments using 35 real-world gene-expression data sets has shown that the EMTTree solution now appears to be one of the top decision tree-like classifiers in the field of gene expression data.

The paper is organized as follows. The next section provides a brief background on DTs in the context of gene expression data analysis. Section 3 describes the concept of multi-test and the proposed evolutionary approach. All experiments are presented in Section 4 and the last section contains conclusions and plans for future work.

2. Background

With the rapid development and popularity of genomic technology, a large number of gene expression datasets have become publicly accessible (Lazar et al., 2012). The availability of these datasets opens up new challenges for existing tools and algorithms. However, traditional solutions often fail due to high features/observations ratios and huge gene redundancy.

2.1. Decision tree

Decision trees (also known as classification trees) have a long history in predictive modeling (Kotsiantis, 2013). The success of the tree-based approach can be explained by its ease of use, speed of classification and effectiveness. In addition, the hierarchical structure of the tree, where appropriate tests are applied successively from one node to the next, closely resembles the human way of making decisions.

DT has a knowledge representation structure made up of nodes and branches, where: each internal node is associated with a test on one or more attributes; each branch represents the test result; and each leaf (terminal node) is designed by a class label. Most of tree inducing algorithms partition the feature space with axis-parallel hyperplanes. Trees of this type are often called univariate because a test in each non-terminal node usually involves a single attribute, which is selected according to a given goodness of split. There are also algorithms that apply multivariate tests (Brodley & Utgoff, 1995) based mainly on linear combinations of multiple dependent attributes. The oblique split causes a linear division of the feature space by a non-orthogonal hyperplane. DTs, which allow multiple features to be tested in a node, is potentially smaller than those which are limited to single univariate splits, but have much higher computational cost and are often difficult to interpret (Brodley & Utgoff, 1995).

Induction of optimal DT is known NP-complete problem (Hyafil & Rivest, 1976). As a consequence, the practical DT learning algorithms must be heuristically enhanced. The most popular type of tree induction is based on a top-down greedy search (Rokach & Maimon, 2005). It starts with the root node, where the locally optimal split (test) is searched according to the given measure of optimality. Then the training instances are redirected to the newly created nodes and this process is repeated for each node until the stop condition is met. In additionally, post-pruning (Esposito, Malerba, & Semeraro, 1997) is usually used after induction to avoid the problem of overfitting to the training data and to improve the generalizing power of the predictive model. The two most popular representatives of top-down DT inducers are CART (Breiman, Friedman, Olshen, & Stone, 2017) and C4.5 (Quinlan, 1992). The CART system generates recursively a binary tree, and the quality of a split is measured either by the Gini index or the Twoing criterion. The C4.5 algorithm applies multi-way splits instead of a typical binary strategy and uses the gain ratio criterion to split the nodes. Inducing DT through a greedy strategy is fast and generally efficient in many practical problems, but usually provides locally optimal solutions.

In order to mitigate some of the negative effects of locally optimal decisions, a wide range of meta-heuristics for the introduction of DT was examined (Barros, Basgalupp, De Carvalho, & Freitas, 2012; Czajkowski & Kretowski, 2014; 2016). They are able to globally search for the tree structure and tests in internal nodes. Such a global induction is of course much more computationally complex, but it can reveal hidden patterns that are often undetectable by greedy methods (Lv, Peng, Chen, & Sun, 2016). Different recent approaches to improving the predictive performance of decision trees include fuzziness (Wang, Liu, Pedrycz, & Zhang, 2015), uncertainties (Cao & Rockett, 2015), discretization (Saremi & Yaghmaee, 2018) or variable selection (Painsky & Rosset, 2017).

2.2. Gene expression data classification with decision trees

Microarrays and RNA-seq analysis can simultaneously measure the expression level of thousands of genes within a particular mRNA sample. The application of a mathematical apparatus and computations tools is indispensable here, since gene expression observations are represented by high dimensional feature vectors. However, the genomic data is still challenging and there are several culprits responsible, mainly: (i) Bellmans curse of dimensionality (too many features); (ii) the curse of dataset sparsity (too few samples); (iii) the irrelevant and noise genes; (iv) bias from methodological and technical factors. Each observation is described by a high dimensional feature vector with a number of features that reach into the thousands, but the number of observations is rarely higher than 100.

Univariate decision trees represent a white-box approach, and improvements to such models have considerable potential for genomic research and scientific modeling of the underlying processes. There are not so many new solutions in the literature that focus on the classification of gene expression data with comprehensive DT models. One of the latest proposals is the FDT (Fuzzy Decision Tree) algorithm (Ludwig, Picek, & Jakobovic, 2018) for classifying gene expression data. The authors compare FDT with the classic DT algorithm (J48) on five popular cancer datasets and have shown some benefits from the use of data uncertainty. Alternative studies are presented in Barros, Basgalupp, Freitas, and De Carvalho (2014) where the authors propose an evolutionary DT inducer called HEAD-DT. Detailed experiments carried out on 35 real-world gene expression datasets have shown the superiority of the algorithm in terms of predictive accuracy compared to well-known DT systems such as C4.5 and CART. An expert system has also been proposed to classify gene expression data using a gene selection by decision tree (Horng et al., 2009). However, existing attempts have shown that decision tree algorithms often induce classifiers with inferior predictive performance (Barros et al., 2014; Ge & Wong, 2008). Current DT-inducing algorithms with their prediction models limited to splits composed from one attribute use only a fraction of the available information. It results in a tendency to underfit as their models have a small bias on the training set, but often fail to classify well the new high-dimensional data. On the other hand, there are algorithms which apply multivariate tests (Brown, Pittard, & Park, 1996) based mostly on linear combination splits. However, the main flaws of such systems are huge complexity as well as the biological and clinical interpretation of the output models is very difficult, if not impossible.

Nowadays, much more interest is given in trees as sub-learners of an ensemble learning approach, such as Rotation or Random Forests (Chen & Ishwaran, 2012; Lu, Yang, Yan, Xue, & Gao, 2017). These solutions alleviate the problem of low accuracy by averaging or adaptive merging of multiple trees. One of the recent examples is the multi-objective genetic programming-based ensemble of trees is proposed in Nag and Pal (2016). The authors present an integrated algorithm for simultaneous selection of features and classification. However, when modeling is aimed at understanding basic environmental processes, such methods are not so useful because they generate more complex and less understandable models (Piltaver, Luštrek, Gams, & Martinčić-Ipšić, 2016). Nevertheless, important knowledge can still be drawn from ensemble methods, e.g. to identify reduced sets of relevant variables in a given microarray (Lazzarini & Bacardit, 2017).

A solution called Multi-Test Decision Tree (MTDT) (Czajkowski et al., 2014) can be placed between one-dimensional and oblique trees. It uses several one-dimensional tests in each node, which on the one hand increases the complexity of the model and on the other hand still allows for relatively easy interpretation of the decision rules. There were, however, a few other flaws and limitations of MTDT, which were addressed and removed with the proposed EMTTree solution, in particular:

- the lack of flexibility in the structure of multi-test - the fixed size of multi-tests in all tree nodes;
- the limited search space - only a few highest-rated attributes were taken into account when building the multi-test (performance reasons);
- the high number of crucial parameters to be defined ad-hoc, including the size of multi-test, the number of alternative multi-tests and the homogeneity of multi-tests;
- greedy top-down induction: meta-heuristic searches (Barros et al., 2012) could be expected to improve classification accuracy and reveal new patterns in the data.

This way, the proposed EMTTree solution that can self-adapt its structure to the currently analyzed data. The undoubted strength of our solution is the higher prediction accuracy and improved stability of the model. The minor weakness of the EMTTree are the results of using an evolutionary approach, mainly the slow tree induction time and a number of input parameters that can be adjusted. However, the gene expression data are still relatively small and as we show in the experimental section, the number of parameters that need to be tuned is small.

2.3. Concept of multi-test

The general concept of the multi-test split, marked *mt*, was introduced for the first time in Multi-Test Decision Tree (MTDT) algorithm (Czajkowski et al., 2014), which induces a DT in a top-down manner. The main idea was to find a split in each non-terminal node that is composed of several univariate tests that branch out the tree in a similar way. The reason for adding further tests was that the use of a single univariate test based on a single attribute may cause the classifier to underfit the learning data due to the low complexity of the classification rule. Each multi-test consists of a set with at least one univariate test. One test in the set is marked as *primary test* (*pt*), and all remaining tests are called *surrogate tests* (*st*). The role of surrogate tests is to support the division of training instances carried out by the primary test with the use of remaining features. In order to determine the surrogate tests, we have adopted the solution proposed in the *CART* system (Breiman et al., 2017). Each surrogate test is constructed on a different attribute and mimics the primary test in terms of which and how many observations go to the corresponding branches. In the majority voting that determines the outcome of the multi-test, the individual weights of each test are equal. This way surrogate tests have a considerable impact (positive or negative) on multi-test decisions, as they can prevail over the primary test decision. It is also possible that a multi-test, without a test with the highest gain ratio, can be the most accurate split.

The experimental evaluation (Czajkowski et al., 2014) showed a significant improvement in classification accuracy and a reduction in underfitting compared to popular DT systems. Results from several real gene expression datasets suggest that the knowledge discovered by MTDT is supported by biological evidence in the literature and can be easily understood and interpreted.

Let's consider a binary classification problem, in which a node contains instances from two classes (*ClassA* and *ClassB*) and instances should be divided into two leaves according to a test. The Fig. 1a illustrates the possible assign of instances to leaves according to the T test performed on a single a attribute. Desired split should place the instances from Class A in left leaf and the instances from Class B in right leaf. Each cell represents a single instance with a defined class, and each row shows how instances are arranged in the leaves after performing the test. From the Fig. 1a it is clear that a single T_1 test on the a_1 attribute has the highest goodness of split, because 13 out of 17 instances are classified correctly. In a typical system, this test should be selected as a split.

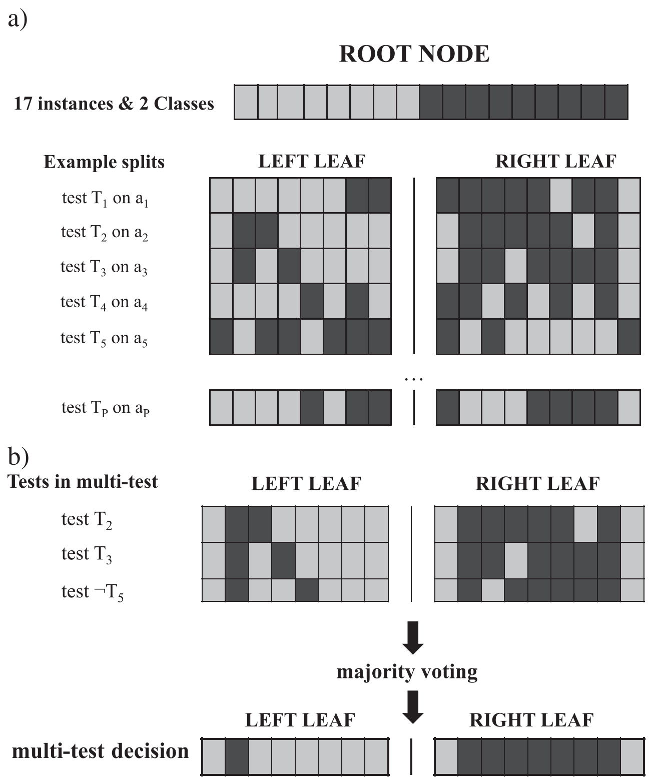

Fig. 1. Possible division scenarios of instances in a root node into two leaves with univariate splits (a) and a multi-test split (b).

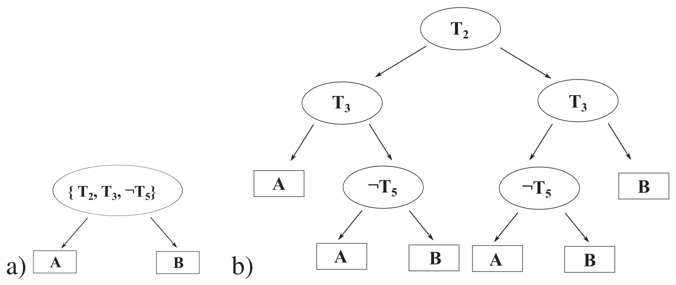

Fig. 2. An example of a multi-test tree (a) and one of its univariate equivalent (b).

In the multi-test search, this top test can also be considered as *pt*. However, there may be a problem with finding appropriate surrogates, because none of the split outcomes in Fig. 1a is similar to the T_1 result.

This is not obvious at first glance, but within the data in Fig. 1a, a better split may be found (see Fig. 1b). Let's assume that the T_2 test on a_2 is the primary test (*pt*). In this case, one of the surrogates may be the T_3 test, which divides instances in a similar way (only 4 instances are assigned differently). Additionally, the T_5 test can also be used in building a multi-test as a surrogate after reversing the relationship between the a_5 attribute and the threshold (marked as $\neg T_5$). Fig. 1b shows the final prediction for the multi-test *mt* consisting of T_2, T_3 and $\neg T_5$. This simple example shows the concept of multi-test and how combined tests can outperform the top T_1 test.

The Fig. 2a illustrates a possible multi-test decision tree for the aforementioned problem with a root node composed from the tests illustrated in Fig. 1b. Since instances are distributed by majority voting, the multi-test node can be easily extended to a univariate tree (Fig. 2). In such a univariate tree, each object is tested no more than the size of the multi-test. Of course, the generated univariate tree is larger, but it is still relatively easy to interpret the resulting tree. It should also be emphasized that the induction of such a univariate tree by a typical decision-making system is very unlikely because all tests within multi-test perform the split in a similar way, which is not common in any known splitting criterion.

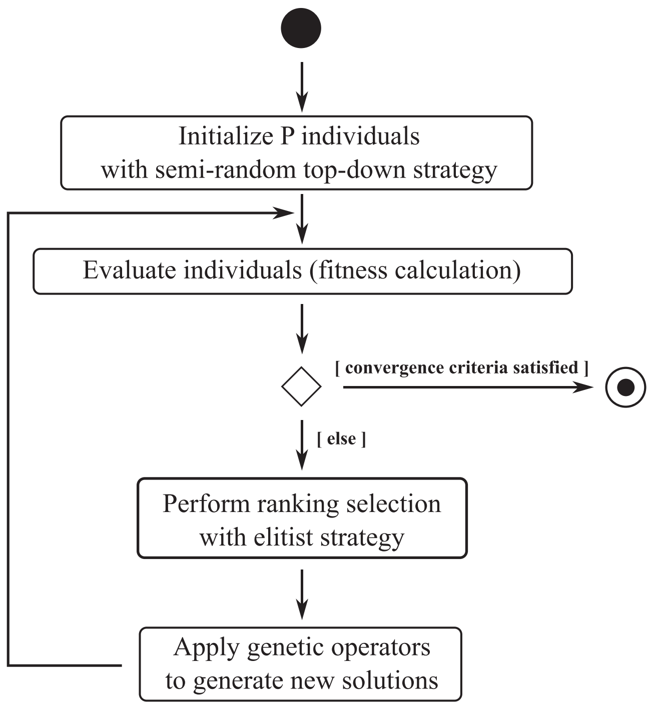
Initialize P individuals
with semi-random top-down strategy
Evaluate individuals (fitness calculation)
[convergence criteria satisfied]
[else]
Perform ranking selection
with elitist strategy
Apply genetic operators
to generate new solutions

Fig. 3. The *EMTTree* process diagram.

3. Evolutionary multi-Test tree

In this section, we present an evolutionary approach to inducing multi-test decision tree. The proposed solution has been integrated into a framework called the Global Decision Tree (GDT). Its overall structure is based on a typical evolutionary algorithm (EA) schema (Michalewicz, 1994) with an unstructured population and generational selection. The GDT framework can be used to induce classification (Grzes & Kretowski, 2007), regression (Czajkowski & Kretowski, 2016) and model trees (Czajkowski & Kretowski, 2014). The process diagram of the *EMTTree* algorithm is illustrated in Fig. 3.

3.1. Representation

The type of EA can be identified by the way in which individuals in populations are represented. The genetic algorithm is usually considered when solutions are encoded in a fixed-length linear string. Tree-encoding schemes usually imply genetic programming (GP), where the solution encodes data and functions (Woodward, 2003); however, the boundary between different types of EAs is unclear and debatable. DT is a complex tree structure in which the number of nodes, the type of tests, and even the number of test outcomes is not known in advance for a given dataset. Therefore, a tree representation may be more appropriate, especially if the entire tree is searched in one EA run. In the GDT system, DTs are not specially encoded and are represented in their actual form as classification trees.

The general structure of a multi-test tree is similar to standard DTs, e.g. C4.5 (Quinlan, 1992). The only difference is that instead of a single univariate test in each non-terminal node split, EMTTree uses a multi-test approach. Fig. 4 illustrates an example of tree

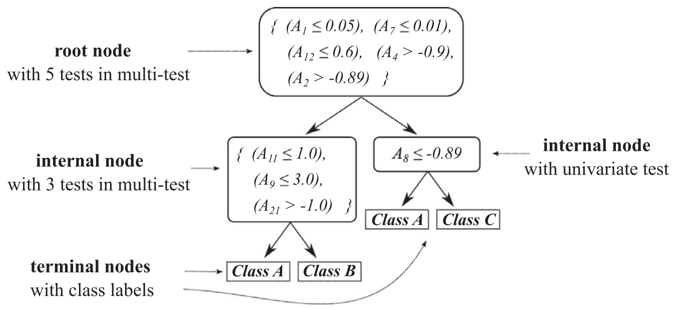

Fig. 4. An example representation of *EMTTree*.

representation of an induced multi-test tree. Each internal node may have a multi-test with a different number of univariate tests that form a split. Typical inequality tests with two outcomes are used, but only on the pre-calculated candidate thresholds (Fayyad & Irani, 1992). The splitting criterion is guided by a majority voting mechanism in which all univariate components of the multi-test have the same weight. To avoid a draw, the number of tests in the multi-test should be odd. The fact that only univariate tests are used in multi-test splits ensures that *EMTTree* can be treated as an axis parallel DT rather than a typical multivariate one, even though more than one test exists in each split. This is important in the context of the later interpretation of the tree.

In addition, each node stores information about training instances related to the node. This allows the algorithm to perform more effectively local modifications of the structure and tests during the application of genetic operators.

3.1.1. A new multi-test split

We define a single multi-test mt as a gene cluster in which genes are tightly linked and could participate in a common pathway. The measure of similarity denoted as resemblance (r_{ij}) for a multi-test located in the ith node (mt_i) between the j surrogate test (st_{ij}) and the primary test (pt_i) is the number of observations routed in the same way to all observations in the node:

$$r_{ij} = \frac{|X_{st_{ij}}|}{|X_i|}, \qquad (1)$$

where $|X_{st_{ij}}|$ is the number of instances routed by st_{ij} in the same way as pt_i, and (X_i) is the set of instances in the i-th node. In case of a binary classification problem, we also consider surrogate tests, which direct instances in the opposite way to their primary test. In the case of such tests, we reverse the relation between the attribute and the interval midpoint and recalculate the score.

$$r_{ij} = \begin{cases} \frac{|X_{st_{ij}}|}{|X_i|}, & \text{if } |X_{st_{ij}}| \geq 0.5 * |X_i| \\ 1 - \frac{|X_{st_{ij}}|}{|X_i|}, & \text{otherwise.} \end{cases} \qquad (2)$$

The EMTTree algorithm requires ranks of all genes from the dataset as its input. These ranks can be perceived as knowledge of discrimination power of each gene and later applied in:

- population initialization;
- multi-test searches and evaluation;
- different variants of genetic operators and fitness.

Ranking can be calculated with any algorithm that assigns ranks to each attribute according to some importance criterion. In our research, we used the Relief-F (Robnik-Šikonja & Kononenko, 2003) algorithm, which is commonly applied in feature selection of gene expression data (Hira & Gillies, 2015). If necessary, the list of ranked genes submitted to the EMTTree can also be manually modified, for example, to focus on biomarker genes for a given disease.

Let the a_{ij} attribute be the jth attribute used in the ith multi-test. Then the $C(a_{ij})$ function returns the cost of the attribute (gene) based on the selected ranking. The values returned by the C function range from 0 and 1, while 0 corresponds to the highest ranked gene and 1 is equal to the worst ranked gene. It should be noted that at this step no attributes are automatically excluded from the dataset, so the EMTTree solution can work on all available genes. In this way, the algorithm is able to find interesting relationships also in low ranked genes. This would be not possible if the standard feature selection was applied as it takes place in most studies.

3.2. Initialization

In general, an initial population should be randomly generated in order to ensure sufficient diversity and cover the whole range of possible solutions. Due to the large search space, the use of greedy heuristics in the initialization phase is often considered as a way of reducing computation time. The disadvantage of this strategy is that EA can be trapped in local optima. Therefore, while creating the initial population, a good trade-off between a high degree of heterogeneity and relatively short computation time is usually desirable. In the GDT system, in order to maintain a balance between exploration and exploitation, initial individuals are created by using a simple top-down algorithm with randomly selected sub-samples of original training data (10% of data, but not more than 500 examples).

3.2.1. Building a multi-test split

In each non-terminal tree node, the EMTTree system divides the training instances that reach this particular point. The algorithm for creating a new multi-test mt_i works as follows. First, the attribute for the primarypt_i test is selected. Due to a large number of possible genes, we have applied an exponential ranking selection (Blickle & Thiele, 1996) to more often include top genes from the data. For the selected attribute a list of all candidate thresholds is created and sorted according to their gain ratio. Then, the algorithm draws pt_i from the list using exponential ranking selection, thanks to which even low-ranked thresholds may appear in the split.

Next, a random even number (default: $j < 8$) of the st_{ij} surrogate tests is created, each on a different attribute. The procedure for searching a surrogate test attribute is analogous to the pt_i test, but a threshold is chosen in a slightly different way. The list of candidate thresholds for the selected attribute is sorted by resemblance (r) to pt_i. The algorithm uses the ranking selection to find the st_{ij} surrogate tests in such a way that it is more likely to select tests that branch instances like pt_i.

3.3. Genetic operators

In order to preserve genetic diversity, the GDT system applies two specialized genetic meta-operators corresponding to the classical mutation and crossover. Both operators have an influence on the tree structure and the multi-tests in the non-terminal nodes. They are applied with a given probability to the tree (the default value is 0.8 for mutation and 0.2 for cross-over). Effective use of any operator makes it necessary to relocate the learning instances between parts of the tree rooted in modified nodes. All mutation and crossover variants are part of the GDT framework described in detail in Czajkowski and Kretowski (2014) and Czajkowski and Kretowski (2016). In this section we include a brief listing of genetic operators used in EMTTree, however, for in-depth description, application and probability of use, please refer to our previous work (Czajkowski & Kretowski, 2014).

3.3.1. Crossover

Each crossover begins with a random selection of two individuals to be affected. The next step is to choose random positions (nodes) in both individuals. Depending on the recombination variant, the selected nodes can:

- exchange the subtrees starting with randomly selected nodes;
- exchange of multi-tests related to randomly selected internal nodes;
- duplicate the subtrees with high accuracy and replace the nodes with a high classification error (asymmetric crossover (Czajkowski & Kretowski, 2014)).

In the last variant, an additional mechanism is used to decide which node will be affected. The algorithm ranks all tree nodes in both individuals according to their classification accuracy. The probability of selection is proportional to the rank in a linear manner. Nodes, for example, with a small classification error, per instance, are more likely to be donors, while weak nodes (with a high classification error) are more likely to be replaced by donors (and become a recipient). We also allow receiver nodes to be replaced by the subtree that starts in the donor node of the best individual. Only one individual is affected in such recombination.

3.3.2. Mutation

The mutation operator starts with a random selection of the node type (equal probability of selecting a leaf node or an internal node). A ranking list of nodes of the selected type is then created and once again a mechanism analogous to the ranking selection is used to decide which node will be affected. Depending on the type of node, the ranking takes into account:

- location (level) of the inner node in the tree - it is obvious that the modification of the test in the root node affects the whole tree and has a big impact, while the mutation of the inner node in the lower part of the tree has only a local impact. Therefore, the internal nodes in the lower parts of the tree are mutated with a higher probability;
- the number of misclassified objects - nodes with a higher error, per instance, are more likely to be mutated. Moreover, pure nodes (nodes with all instances from one class) are not mutated.

Modifications made by the mutation operator depend on the node type and include different variants that:

a) modify the tree structure:
   - prune the internal nodes;
   - expand leaves that contain objects from different classes;
   - replace the multi-test in the internal node with a new one with a random even number of surrogates;
b) modify the selected multi-test:
   - add 2 additional surrogate tests with different attributes. Tests with higher *r* to *pt* are selected more likely;
   - removes (if possible) 2 surrogate tests. Tests with a lower *r* to *pt* are selected more likely;
   - shift the *pt* threshold and update all (thresholds and *r*) surrogates;
   - replace current *pt* with a newly created *pt* on a new attribute and update surrogates;
   - switch within a multi-test *pt* with one of the randomly selected *st* and update surrogates.

3.4. Fitness function

The evolutionary search process is very sensitive to the correct definition of the fitness function, which drives the evolutionary search process, measuring how good an individual is in terms of meeting the problem objective. In the context of DT, a direct minimization of accuracy measured on the learning data usually leads to an overfitting problem and poor performance on unseen, test observations due to overgrown trees. In typical top-down induction (Rokach & Maimon, 2005), this problem is partially mitigated by performing a stop condition and applying post-pruning (Esposito et al., 1997). In the case of evolutionary induced DT, this problem may be controlled by a complexity term incorporated into the fitness function.

In general, it is recommended to maximize the accuracy and minimize the complexity of the output tree (Czajkowski & Kretowski, 2019). However, in the case of gene expression data, these criteria cannot be applied directly. The main reason is the large disproportion between the number of instances and the attributes, which may cause the classifier to underfit the learning data. If complexity is minimized or ignored, then the multi-test tree could become a univariate one because such trees often classify the training data perfectly. On the other hand, if the fitness function promotes more complex multi-tests, the output tree will have overgrown splits in internal nodes, which may be difficult to analyze and interpret.

Considering our motivations and goals, the desired tree should have multi-test splits consisting of several highly ranked tests that branch out the nodes in a similar way. Therefore, the proposed fitness function should promote individuals with:

a) high accuracy on the training set;
b) relatively large size of multi-tests;
c) high resemblance of the univariate tests that constitute multi-tests;
d) low cost of attributes in multi-tests.

Therefore, the EMTTree system maximizes the fitness function, which has the following form:

$$Fitness(T) = Q(T) + \alpha * R(T) + \beta * Cost(T), \tag{3}$$

where: $Q(T)$ is the accuracy, $R(T)$ is the sum of $R(T_i)$ in all multi-tests of the T tree, $Cost(T)$ is the sum of the costs of attributes constituting multi-tests. The default parameters values are: $\alpha = 1.0$ and $\beta = -0.5$, and more information on tuning these parameters can be found in the next section (see Section 4.2).

Let us consider an internal T_i node of the T tree with mt_i multi-test. Then:

$$R(T_i) = \frac{|X_i|}{|X|} * \sum_{j=1}^{|mt_i|-1} r_{ij}, \tag{4}$$

where X is a learning set, X_i is a set of instances in ith node, and $|mt_i|$ is the size of a multi-test. If a multi-test is composed of a single test, then $R(T_i)$ equals 0.

The cost of attributes in the multi-test mt_i depends on their rank, and the number of instances that reach the i node:

$$Cost(T_i) = \frac{|X|}{|X_i|} * \sum_{j=1}^{|mt_i|} C(a_{ij}), \tag{5}$$

where a_{ij} is the jth attribute of the ith multi-test and $C(a_{ij})$ is the cost of the a_{ij} attribute. The reason why $Cost(T_i)$ increases when the number of instances in a node decreases is to avoid the overfitting in the lower parts of the tree, as this will eventually limit the size of the multi-test. However, in contrast to Czajkowski et al. (2014), the maximum number of univariate tests that make up a multi-test is not defined.

3.5. Selection, and terminal condition

The linear ranking selection (Michalewicz, 1994) is used as a selection mechanism. Additionally, in each iteration, one individual with the highest fitness in the current population is copied to the next one (elite strategy). The evolution ends when the fitness of the best individual in the population does not improve during the fixed number of generations (default: 1000). In the case of slow convergence, the maximum number of generations is also defined (default: 10000), which limits the computation time.

4. Experimental results

In this section, we present a detailed experimental analysis to evaluate the relative performance of the proposed evolutionary multi-test tree approach. In Section 4.1 we describe algorithms that will be compared with EMTTree and briefly characterize 35 gene expression datasets used in the analysis. Section 4.2 shows our strategy for tuning algorithm parameters. Section 4.3 presents and discusses the results of the comparison with the original multi-test tree and other classifiers. Finally, Section 4.4 comments on detailed EMTTree statistics based on three dataset examples.

4.1. Algorithms and datasets

In order to make a proper comparison, we have confronted the proposed EMTTree solution with the state-of-the-art and the latest algorithms in the literature. Performed experiments encompas various decision tree inducers:

a) EMTTree: proposed evolutionary multi-test tree approach;
b) MTDT (Czajkowski & Kretowski, 2014): original concept of a multi-test tree used in the top-down inducer;
c) HEAD-DT (Barros et al., 2014): hyper-heuristic EA for designing decision-tree algorithms tailored to gene expression datasets;
d) CART (Breiman et al., 2017): one of the most well known top-down algorithm for decision-tree induction. We employed its java version available from the Weka machine learning toolkit (Frank, Hall, & Witten, 2017) under the name of SimpleCART;
e) C4.5 (Quinlan, 1992): popular state-of-the-art tree learner, Weka implementation under the name of J48;
f) REPTree: variation of C4.5 that employs the reduced error pruning.

We use a set of 35 publicly available gene expression datasets originally described in de Souto, Costa, de Araujo, Ludermir, and Schliep (2008) and available at de Souto, Costa, de Araujo, Ludermir, and Schliep (2019). The data relates to different types or subtypes of cancer. The authors of HEAD-DT randomly divided 35 datasets into two groups (Barros et al., 2014): parameter optimization and experiments (Table 1). We used exactly the same datasets so we can directly compare their published results with our approach. Since the datasets were not pre-divided into training and testing parts, 10-fold cross-validation was applied and an average score of 50 runs was presented. We have also included information about the standard deviation.

4.2. Parameter tuning

The characteristics of the data for a specific problem domain may differ, so we have adjusted all the base algorithms (except HEAD-DT) on 15 gene expression datasets. The results of HEAD-DT are only cited as the solution has already been tuned using exactly the same datasets (Barros et al., 2014).

In the case of the MTDT solution, the size of multi-tests in tree nodes must be set in advance using the tuning set. The performance of the MTDT classifier is tested using six values of the N parameter (as in the original publication (Czajkowski et al., 2014)):

Table 1
Summary of 15 gene expression datasets for tuning and 20 for algorithm testing. Each dataset is described by its name, the total number of instances (I), attributes (A), and classes (C).

Tuning datasets				Testing datasets			
Dataset	I	A	C	Dataset	I	A	C
armstrong-v2	72	2193	3	alizadeh-v1	42	1094	2
bredel	50	1738	3	alizadeh-v2	62	2092	3
dyrskjot	40	1202	3	alizadeh-v3	62	2092	4
garber	66	4552	4	armstrong-v1	72	1080	2
golub-v2	72	1867	3	bhattacharjee	203	1542	5
gordon	181	1625	2	bittner	38	2200	2
khan	83	1068	4	chen	179	84	2
laiho	37	2201	2	chowdary	104	181	2
nutt-v3	22	1151	2	golub-v1	72	1867	2
pomeroy-v2	42	1378	5	lapointe-v1	69	1624	3
ramaswamy	190	1362	14	lapointe-v2	110	2495	4
su	174	1570	10	liang	37	1410	3
tomlins-v2	92	1287	4	nutt-v1	50	1376	4
yeoh-v1	248	2525	2	nutt-v2	28	1069	2
yeoh-v2	248	2525	6	pomeroy-v1	34	856	2
				risinger	42	1770	4
				shipp-v1	77	797	2
				singh	102	338	2
				tomlins-v1	104	2314	5
				west	49	1197	2

Table 2
Default EMTTree parameters.

Parameter	Value
Population size	50 individuals
Crossover rate	20% assigned to the tree
Mutation rate	80% assigned to the tree
Elitism rate	2% of the population (1 individual)
Max generations	10 000

1, 3, 5, 7, 9 and 11, which means a maximum number of univariate tests in the multi-test. It is worth noting that MTDT with a single test in a multi-test node ($N = 1$) behaves similarly to the standard C4.5 algorithm. Both algorithms use the gain ratio criterion and pessimistic pruning. There is, however, a slight difference in calculating the exact threshold value, which is described in Czajkowski et al. (2014). The best average results on the tuning datasets were achieved when the maximum size of the multi-test is 5, so the classifier with this setting denoted as MTDT$_5$ will be used in subsequent experiments.

As for state-of-the-art solutions (CART, C4.5, REPTree) we have tested a large number of parameters as recommended in Auto_WEKA 2.0 (Kotthoff, Thornton, Hoos, Hutter, & Leyton-Brown, 2016). For all three cases, the unpruned trees reached the highest average prediction performance on 15 datasets. Average classification accuracy for unpruned trees: CART, C4.5, and REPTree increased compared to the baseline settings by: 0.3%, 1.3% and 3.7%, respectively.

Considering that EMTTree is regular generational EA, parameters such as population size, a maximum number of generations, elitism rate, crossover and mutation probability must be selected before evolution. Table 2 contains a brief listing of the main parameters that have been experimentally evaluated and described in details in our previous publications on GDT framework (Czajkowski & Kretowski, 2014; 2016).

In the EMTTree algorithm, however, there are two additional parameters that should be experimentally tuned. The fitness function described in Eq. (3) has three objectives: accuracy, the resemblance of surrogate tests and cost of attributes. It is clear why the accuracy should have highest possible value, but the question that may be raised is why $R(T)$ is twice as important as $Cost(T)$. Let us consider the general definition of the proposed fitness function

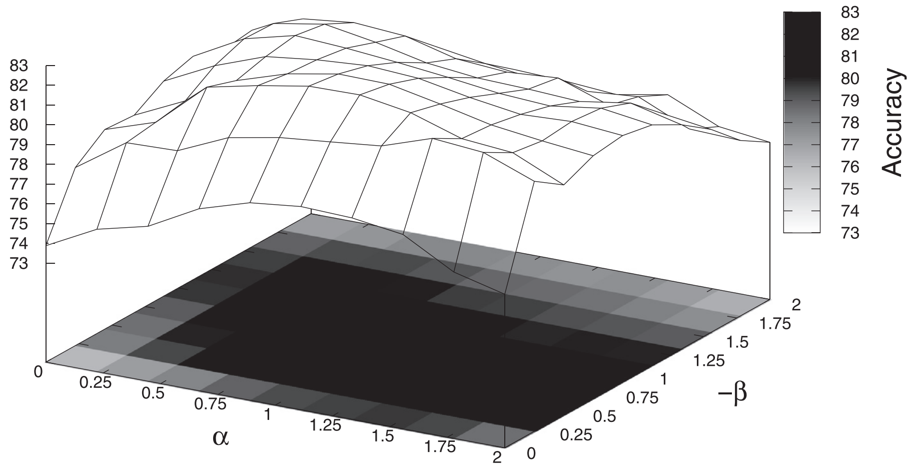

Fig. 5. Impact of the resemblance: α and cost: β parameters on EMTTree system performance.

Table 3
Comparison of EMTTree accuracy to popular decision tree inducers.

	Evolutionary inducers		Top-down inducers			
Dataset	EMTTree	HEAD-DT	MTDT$_5$	CART	C4.5	REPTree
alizadeh-v1	**0.91 ± 0.12**	0.77 ± 0.11	0.86 ± 0.16	0.67 ± 0.21	0.68 ± 0.23	0.65 ± 0.20
alizadeh-v2	**0.96 ± 0.07**	0.88 ± 0.06	0.94 ± 0.10	0.89 ± 0.11	0.91 ± 0.12	0.91 ± 0.11
alizadeh-v3	0.71 ± 0.13	**0.74 ± 0.05**	0.71 ± 0.10	0.71 ± 0.18	0.70 ± 0.20	0.71 ± 0.14
armstrong-v1	**0.95 ± 0.08**	0.90 ± 0.04	0.93 ± 0.05	0.89 ± 0.09	0.88 ± 0.10	0.91 ± 0.09
bhattacharjee	**0.90 ± 0.06**	**0.90 ± 0.03**	**0.90 ± 0.03**	**0.90 ± 0.08**	0.89 ± 0.06	0.85 ± 0.07
bittner	**0.70 ± 0.27**	0.61 ± 0.09	0.62 ± 0.16	0.62 ± 0.21	0.61 ± 0.20	0.63 ± 0.24
chen	**0.88 ± 0.07**	0.85 ± 0.04	0.87 ± 0.06	0.85 ± 0.08	0.85 ± 0.08	0.83 ± 0.09
chowdary	**0.95 ± 0.06**	**0.95 ± 0.03**	0.94 ± 0.05	0.93 ± 0.07	0.93 ± 0.07	0.92 ± 0.07
golub-v1	**0.95 ± 0.07**	0.88 ± 0.03	0.93 ± 0.04	0.85 ± 0.10	0.86 ± 0.12	0.89 ± 0.11
lapointe-v1	**0.73 ± 0.18**	0.66 ± 0.08	0.72 ± 0.16	0.70 ± 0.16	0.70 ± 0.17	0.71 ± 0.16
lapointe-v2	0.69 ± 0.11	0.62 ± 0.05	**0.72 ± 0.11**	0.63 ± 0.13	0.63 ± 0.14	0.60 ± 0.13
liang	**0.93 ± 0.11**	0.89 ± 0.09	0.92 ± 0.12	0.78 ± 0.18	0.83 ± 0.21	0.83 ± 0.17
nutt-v1	0.54 ± 0.20	0.53 ± 0.08	0.54 ± 0.20	0.54 ± 0.23	**0.56 ± 0.20**	0.55 ± 0.22
nutt-v2	**0.87 ± 0.09**	0.84 ± 0.08	0.80 ± 0.22	0.80 ± 0.26	0.77 ± 0.28	0.80 ± 0.26
pomeroy-v1	0.83 ± 0.18	**0.88 ± 0.08**	0.74 ± 0.21	0.82 ± 0.23	0.75 ± 0.21	0.73 ± 0.22
risinger	**0.64 ± 0.20**	0.58 ± 0.15	0.60 ± 0.24	0.53 ± 0.19	0.50 ± 0.19	0.51 ± 0.24
shipp-v1	0.86 ± 0.12	**0.91 ± 0.05**	0.79 ± 0.12	0.76 ± 0.12	0.76 ± 0.14	0.81 ± 0.14
singh	**0.89 ± 0.09**	0.78 ± 0.04	0.76 ± 0.16	0.79 ± 0.13	0.80 ± 0.12	0.77 ± 0.12
tomlins-v1	**0.66 ± 0.14**	0.59 ± 0.07	0.64 ± 0.10	0.60 ± 0.14	0.59 ± 0.14	0.61 ± 0.13
west	0.86 ± 0.11	**0.89 ± 0.08**	0.82 ± 0.14	0.87 ± 0.15	0.88 ± 0.15	0.82 ± 0.15
average	**0.82 ± 0.12**	0.78 ± 0.07	0.79 ± 0.13	0.76 ± 0.15	0.75 ± 0.16	0.74 ± 0.15

presented in Eq. (3). The role of parameters α and β is to control the relative importance of each objective. Fig. 5 shows the parameter tuning experiment varying α and -β within {0.0, 0.25, 0.5,0.75, 1.0, 1.25, 1.5, 1.75, 2.0}. To select the best value of α and β we used 15 sets of gene expression data belonging to parameter optimization group (tuning data in Table 1). The aim of the experiment is not to optimize the parameters for a given dataset, but to find robust values that generally work well in the domain. We then used these founded values of α and β to evaluate the generalizing ability of the fitness function in the new datasets (testing datasets in Table 1).

4.3. Comparison of EMTTree to popular decision trees

This section compares EMTTree to the popular decision tree inducers described in Section 4.1 on 20 gene expression datasets belonging to the experiment group (Table 1). The results represented by classification accuracies and standard deviations are shown in Table 3 and the highest accuracy for each dataset is bold.

Statistical analysis using the Friedman's test showed that there are significant statistical differences between algorithms (significance level is equal to 0.05, P-value < 0.0004, F-statistic $= 22.37$). According to Dunn's multiple comparison test (significance level of 0.05) EMTTree managed to significantly outperform all tested state-of-the-art decision tree solutions: CART, C4.5, and REPTree. No statistical differences were observed for HEAD-DT and MTDT$_5$. We believe this is a good result, especially considering that the Dunns test is the most conservative option (less likely to find a significant difference) among all multiple comparison procedures (Demsar, 2006). In addition, it should be noted that there are no other significant differences between algorithms, e.g. between HEAD-DT and CART/C4.5/REPTree.

Table 4 illustrates the size of the induced trees. We can see that EMTTree on average induces smaller trees than its competitors. However, the overall complexity of EMTTree is higher because instead of using a single univariate test in each non-terminal node, as in the other systems, the proposed algorithm applies multi-tests with about 5 univariate tests within each split. In this way, EMTTree induces trees that are about 2.5 times more complex than state-of-the-art top-down induced systems (if we compare the total number of univariate tests) but smaller than MTDT$_5$. Unfortunately, in the publication (Barros et al., 2014) there is no information about the sizes of trees induced by HEAD-DT solution. It should be noted, that the tuned parameters for CART, C4.5 and

Table 4
Comparison of EMTTree complexity with other classifiers.

Dataset	EMTTree nodes × MT	$MTDT_5$ nodes × MT	CART nodes	C4.5 nodes	REPTree nodes
alizadeh-v1	2.1 × 2.9	3.5 × 5.0	5.0	5.0	5.0
alizadeh-v2	3.2 × 5.9	3.0 × 5.0	5.0	5.0	5.0
alizadeh-v3	6.3 × 4.9	5.6 × 5.0	9.0	9.0	9.0
armstrong-v1	2.1 × 7.2	2.7 × 5.0	3.8	3.8	3.8
bhattacharjee	7.4 × 6.9	8.8 × 5.0	12.2	11.3	12.0
bittner	2.4 × 3.6	3.9 × 5.0	5.1	5.0	5.0
chen	3.3 × 9.4	14.0 × 5.0	13.4	17.1	15.0
chowdary	2.0 × 10.8	4.8 × 5.0	5.1	5.9	7.6
golub-v1	2.0 × 3.1	3.2 × 5.0	4.6	4.6	4.6
lapointe-v1	5.9 × 3.5	7.0 × 5.0	9.6	10.2	9.3
lapointe-v2	10.7 × 3.4	13.4 × 5.0	16.9	18.7	17.1
liang	4.0 × 1.2	3.0 × 5.0	5.0	5.0	5.0
nutt-v1	6.8 × 3.4	6.9 × 5.0	10.5	9.7	10.7
nutt-v2	2.0 × 8.7	3.1 × 5.0	4.4	4.4	4.4
pomeroy-v1	2.1 × 2.9	3.9 × 5.0	3.2	4.0	3.4
risinger	5.4 × 3.0	6.2 × 5.0	8.6	8.6	8.8
shipp-v1	2.1 × 9.8	5.7 × 5.0	7.2	9.4	7.1
singh	2.1 × 9.1	10.2 × 5.0	9.5	11.0	10.3
tomlins-v1	11.5 × 3.8	12.2 × 5.0	18.8	18.2	18.4
west	2.1 × 6.3	4.2 × 5.0	4.7	4.7	4.7
average	4.2 × 5.2	6.3 × 5.0	8.1	8.5	8.3

REPTree algorithms include using unpruned trees in order to improve classification accuracy. The average tree sizes for pruned CART, C4.5 and REPTree systems with default parameters are equal 5.4, 8.1 and 4.3 respectively.

Given that traditional DT systems induce relatively very small trees, we believe that our goal has been achieved as such an increase in complexity does not strongly affect the comprehensibility of the output tree. In addition, a multi-test that acts as a gene cluster can be biologically meaningful and interesting on its own. In the following case study, we will examine whether the discovered genes from the classifier model are supported by biological evidence in the literature.

As we argue at the beginning of the article, the underfitting is one of the reasons why traditional tree inducers fail to effectively classify the gene expression data. Underfitting to the data is caused by oversimplified rules generated from the output trees, and the results clearly indicate that the application of multi-tests improves prediction accuracy. Table 3 shows that EMTTree is the most stable, with no occasional heavy failures, which is in contrast to all other tested solutions. EMTTree has achieved, on average over 20 datasets, the highest classification accuracy: more than 6% higher than state-of-the-art solutions and 4% higher than the latest algorithm (HEAD-DT) designed for gene expression data.

When evaluating an algorithm only in terms of predictive performance (ignoring the interpretability of the classification model), ensemble methods such as Random Forests (RF), Bagging (BG) or Adaboost (ADA) are usually considered to be the state-of-the-art classification algorithms for gene expression data and generally outperform standard single-tree approaches. To give just an idea of the level of predictive performance that can be obtained for our data, we carry out additional tests using the WEKA toolkit (Frank et al., 2017). Among ensembles, RF managed to achieve the best average accuracy (0.85 ± 0.12) and is significantly better than all competitors, including BG (0.78 ± 0.13) and ADA (0.81 ± 0.14). Statistical analysis showed that only EMTTree is not statistically worse than RF. In addition, the proposed solution, despite the lower average accuracy compared to RF, managed to outvote (or be equal) in 10 of the 20 tested datasets. It should be noted, however, that ensemble classifiers generate more complex predictive models, which are more difficult to analyze and interpret.

Finally, the use of EA entails higher calculation costs than classical procedures. The average induction time for EMTTree ranged from one to several minutes (an average equals 12 min) for each dataset on a typical PC (Intel Core I5, 4GB RAM). The execution of the EMTTree solution, however, on test instances is very fast. As we have shown in our previous work (Jurczuk, Czajkowski, & Kretowski, 2017), it is easy to speed up our algorithm even hundreds of times using the GPGPU approach. In addition, gene expression data do not really require real-time decision-making, so the computational time is usually less irrelevant.

4.4. Case study

In this section, we would like to discuss more details of the EMTTree results on three representative datasets: Alizadeh-v2, Liang and Singh. There are two reasons why these particular datasets were selected. Firstly, they all are very different in term of the number of instances, attributes, and classes. Secondly, the trees induced by the EMTTree system are also different in term of tree size and the multi-test size.

In the following figures (6–9), we want to show wherever the multi-test and the cost information affect the basic statistics of the best individual founded so far in the evolution. We track the following statistics:

- tree size (number of internal nodes);
- average multi-test size (number of univariate tests in each internal node);
- accuracy on the testing set;
- average resemblance of the multi-tests;
- average percent of univariate tests within multi-tests built on top-ranked attributes (top 20 attributes with the highest rank / lowest cost);
- average percent of univariate tests within multi-tests built on low-ranked attributes (attributes with a rank over 200);

for:

- EMTTree: proposed system with default parameters;
- EMTTreeNC: the EMTTree system with no cost information (cost of each attribute is equal);
- ETree: the EMTTree system with a single univariate test instead of a multi-test (multi-test size is equal to one). The algorithm applies only univariate tests in each internal node and may be considered as an evolutionary version of C4.5 algorithm;

In Figs. 6–9 the average results (through 50 algorithms runs) for the best individual in each generation are showed. For all performed experiments exactly 10 000 iterations are presented, although, most of the “action” happens in the first thousands of iterations. Accuracy on the training part of the cross-validated set is not shown as it achieves 100% in the first 50–100 iterations (except Singh dataset in which it takes a few times longer) and might obscure the figures. In addition, Fig. 10 shows the example of trees induced by the EMTTree solution for each dataset.

Most of recently proposed data mining methods for omics data generate complex rules that constrain the process of uncovering new biological understanding that, after all, is the ultimate goal of data-driven biology. However, it is not enough to simply produce good outcomes but to provide logical reasoning just as clinicians do for medical treatments. That is why, we also have evaluated the knowledge derived from EMTTree prediction models to check if it is biologically meaningful. However, it should be noted that explicitly connecting classification results and molecular mechanism for particular diseases is a major challenge. However, proposed research anticipates the longer-term goal of translational medicine by incorporating generated decision rules in delineating candidate biomarkers.

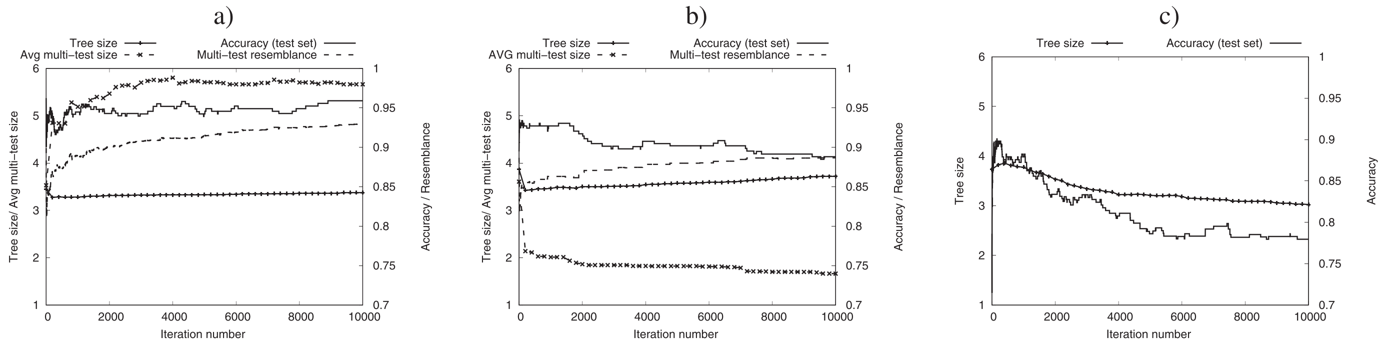

Fig. 6. The performance of the best individual founded so far on Alizadeh-v2 dataset for a) EMTTree, b) EMTTreeNC and c) ETree.

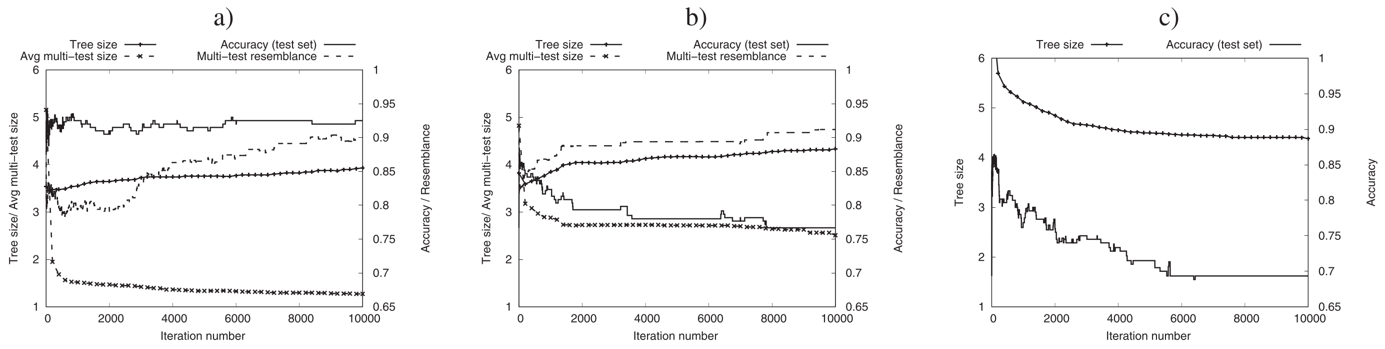

Fig. 7. The performance of the best individual founded so far on Liang dataset for a) EMTTree, b) EMTTreeNC and c) ETree.

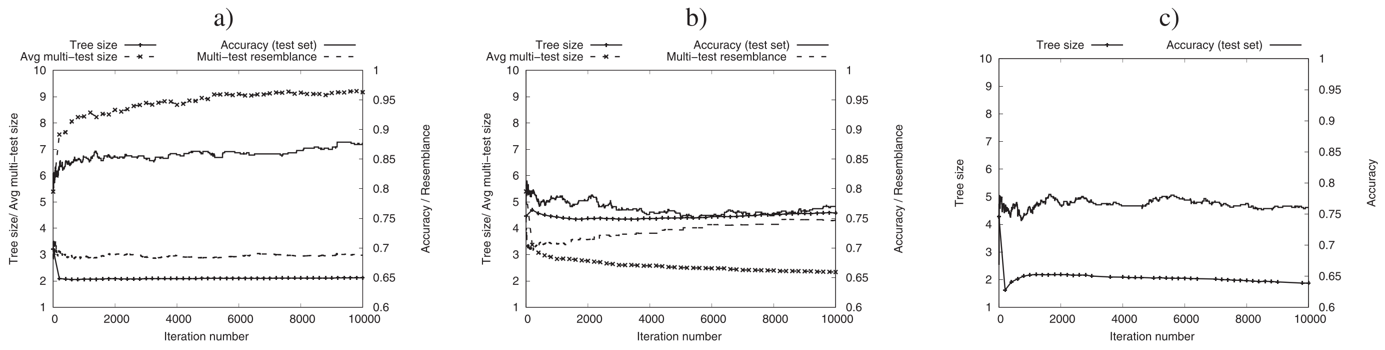

Fig. 8. The performance of the best individual founded so far on Singh dataset for a) EMTTree, b) EMTTreeNC and c) ETree.

4.4.1. Alizadeh-v2 dataset

The Alizadeh dataset (Alizadeh et al., 2000) characterizes the gene expression patterns of the three most prevalent adult lymphoid malignancies: diffuse large B-cell lymphoma (DLBCL), follicular lymphoma (FL) and chronic lymphocytic leukemia (CLL). The dataset contains 2092 expression profiles for 62 samples: 42 DLBCL, 9 FL, and 11 CLL.

Fig. 6 illustrates the statistics for the best individuals found so far for three studied algorithms: EMTTree, EMTTreeNC, and ETree. The EMTTree solution induces the most accurate trees (see Fig. 6a). The final size of the tree is found in less than a hundred iterations, but the multi-test search required more time. A longer multi-test adjustment is due to a large number of degrees of freedom - its size, the resemblance of the tests and the cost of the attributes. However, despite multi-test changes, the prediction performance is stable, which confirms the robustness of trees induced by the EMTTree algorithm. In addition, Fig. 9a shows the percentage share of the highest/lower rank attributes in tests. Figs. 6 and 9a reveal that the EMTTree tree uses splits consisting of 5 univariate tests similar in more than 90% and based on 85% of the top ranked attributes. Such high-quality multi-tests improves the stability of internal nodes and thus the whole classifier.

Fig. 6b and c illustrate how our solution would work without incurring costs (rank of attributes). We see that unlike EMTTree, the best individual for EMTTreeNC has a similar tree size, but much smaller multi-tests. As the evolution progresses, the classifier reduces its complexity and improves the resemblance of the multi-tests. With the iteration number, the prediction error on a testing set increases, suggesting that the classifier slowly underfits to the training parts of the cross-validated data for which the training error reached zero in less than 100 iterations. In addition, Fig. 9a reveals that the percentage of top attributes in multi-tests is around 30% which is almost 3-times less than in the case of EMTTree. Both the low complexity of the output tree and the noisy or insignificant attributes in tests are probably responsible for the poor EMTTreeNC performance.

Exactly the same problems as with EMTTreeNC occur for ETree solution that uses a single univariate test in the split nodes (see Fig. 6c). After finding in the first 50–100 iterations the best individual, which perfectly classifies the training data, for the rest

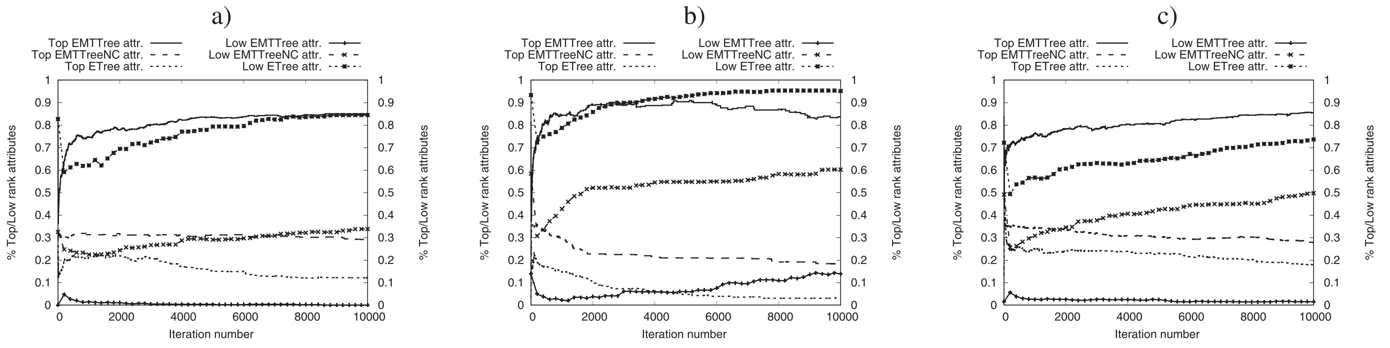

Fig. 9. Top and low-ranked genes used in the best individual founded so far for a) Alizadeh-v2, b) Liang and c) Singh dataset.

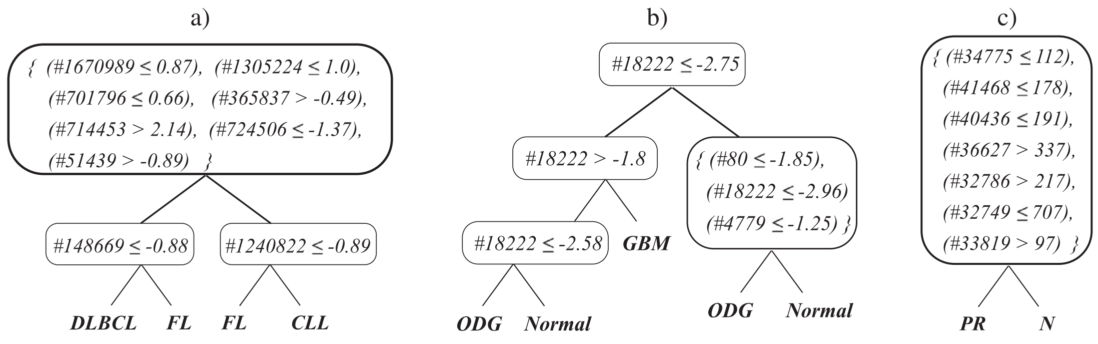

Fig. 10. An example tree induced by EMTTree for a) Alizadeh-v2, b) Liang and c) Singh dataset. Each classifier predicts perfectly the training and testing parts of the cross-validated sets.

of the evolution the algorithm searches for the smallest possible tree. This has a negative effect on the forecasting performance of the test set, because such a small and simple tree underfits the training data. Fig. 9a also shows that in most of the splits the medium-ranked attributes were chosen (only 12% of the tests used top-ranked attributes).

An example EMTTree tree is shown in Fig. 10a. The accuracy results for the training and test set for this particular tree are 100%. We see that a multi-test with 7 tests appears in the root node and two simple univariate tests in the tree sub-nodes.

Based on the description of the dataset (GSE60 series) from GenBank NCBI (Benson et al., 2018) we conducted a thorough examination of each gene in the context of a possible gene cluster. A search for known marker genes revealed that the proposed model is supported by biological evidence in literature. Of 9 tests used in the prediction model, 7 were performed on genes directly related to lymphoid malignancies. For example, the PCDH9 gene (#51439) is down-regulated in non-nodal mantle cell lymphoma and glioblastoma as a result of gene copy number alterations, and that exogenous expression of PCDH9 could inhibit tumor cell migration (Wang et al., 2012). Importantly, some root node tests are part of a gene cluster that focuses on the same mutations e.g. #701796 and #714453, which is a novel driver alternation IL4R (Vigano et al., 2018) for a distinct subtype of diffuse large B-cell lymphoma called primary mediastinal large B-cell lymphoma (PMBCL).

4.4.2. Liang

Glioblastoma multiforme (GBM) is the most common form of malignant glioma, characterized by genetic instability, intratumoral histopathological variability, and unpredictable clinical behavior. The Liang dataset (Liang et al., 2005) contains global gene expression in surgical samples of brain tumors. Gene expression profiling revealed large differences between normal brain samples (class Normal) and tumor tissues: GBMs and lower-grade oligodendroglial (ODG) tumors.

Results for the Liang dataset are shown in Figs. 7 and 9b. Again, the performance of the EMTTree tree clearly outperforms the rest of the competitors. When comparing the statistics of the best individuals generated by EMTTree and EMTTreeNC systems (Figs. 7a and 6b), a few things can be seen. In contrast to Alizadeh dataset, EMTTreeNC induced this time more complex trees with multi-tests more consistent to those in EMTTree. So why the predictive performance of the best individual induced by EMTTreeNC is still much lower than that induced by EMTTree? The answer can be found in Fig. 9b. We see that the EMTTree solution used around 85% of top attributes, while EMTTreeNC only 20%. Surprisingly, the EMTTree tree also applied some of the low-ranked attributes in the output model, which may suggest that they are somehow useful. In the case of the traditional univariate tree (ETree) there are no changes - the classifier underfits to the data (Fig. 7c) because of its low complexity.

An example of the EMTTree output tree is illustrated in Fig. 10b. We see that the multi-tests can also occur in the lower parts of the tree, but despite the additional complexity, the classification process is still easy to understand and interpret. In addition, we can observe that attribute #18222 is used in all the splits, which may suggest its importance. To confirm this, we checked in the NCBI dataset series GSE4058 the meaning of #18222 gene symbol, and then reviewed recent publications on significant drivers in GBM. It turned out that the found COL1A2 gene is an identified key cancer-related gene and a potential target gene for diagnosing the glioblastoma (Long et al., 2017).

4.4.3. Singh

The Singh dataset (Singh et al., 2002) contains gene expression patterns from 52 prostate tumors (PR) and 50 normal prostate specimens (N). The results for the best individuals (see Figs. 8 and

9c) confirm the discussion and conclusions from the previous case studies. An important fact that can be noticed is that the inclusion of the cost of attributes in the evolutionary induction affects not only the predictive performance, but also the entire tree structure. This time our solution (Fig. 8a) induces a decision tree with a complex multi-test (see Fig. 10c), while the EMTTreeNC algorithm builds a larger tree, but with simple splits (Fig. 8b). However, the prediction performance is still in favor of the tree induced by the EMTTree solution.

With the NCBI GPL8300 series, we checked what actual genes are used in the prediction model. We discovered that all 7 genes applied in the multi-test seem to be important biomarkers in a prostate cancer. What's more, we also checked whether the rules for these genes are relevant in some way. Of course, the individual test thresholds are not identical to those described in medical research, but in all results, the effect of the tests sign in the multi-test: $>$ and $\leq$ on classification is the same as the effect of up and down-regulated genes on prostate cancer prediction. For example, in the presented prediction model (see Fig. 10c), if the TSPAN1 gene expression is low (test: #34775 $\leq$ 112), it votes for the class PR (prostate cancer) and in the literature we may find that the decreased expression for the TSPAN1 gene promotes the prostate cancer progression and is considered a crucial biomarker (Xu et al., 2016).

5. Conclusion

Achieving high classification accuracy of models for gene expression datasets is still a major problem for the tree-based inducers. State-of-the-art and modern algorithms that induce univariate single-tree solutions have a tendency to underfit to the training data and fail to achieve high accuracy on the unseen instances. We address this problem through an evolutionary multi-test tree approach and have managed to significantly improve the performance of decision trees with a slight increase in their complexity. Our solution redefines the concept of multi-test and focuses on the concept of gene clusters. The proposed evolutionary induction algorithm for multi-test DT induction enables a global search for tree structure and multi-tests with various sizes. Additional information related to the cost of attributes improves algorithm robustness and EA convergence to the global optimum. Extensive experimental validation performed on 35 publicly available gene expression datasets shows that the proposed EMTTree solution is significantly better in terms of prediction performance than state-of-the-art decision tree inducers. Equal, if not more important, is the fact that the decisions triggered by EMTTree are still simple and can have direct applicability in the field of gene expression analysis.

The knowledge discovered by EMTTree is supported by biological evidence in the literature. A biologist can, therefore, benefit from this white box approach, as it can produce accurate and biologically meaningful classification models and reveal new patterns in biological data. In particular, it may be promising to study multi-tests in internal nodes in the context of searching for genes forming a possible gene cluster.

We see many promising directions for future research. In particular, from the point of view of machine learning, we are focusing on improving the algorithm efficiency, especially for large data via GPGPU parallelization (Jurczuk et al., 2017). On the other side, we are currently working with biologists and bioinformaticians to better understand the rules generated by EMTTree. We also want to adopt the proposed classification algorithm to work with protein expression and metabolic databases as well as any other specific application area.

Conflict of interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Credit authorship contribution statement

Marcin Czajkowski: Conceptualization, Methodology, Software, Validation, Formal analysis, Writing - original draft. **Marek Kretowski:** Conceptualization, Writing - review & editing, Supervision, Funding acquisition.

Acknowledgments

This work was supported by the grant W/WI/1/2017 (first author) and S/WI/2/18 (second author) from BUT founded by Polish Ministry of Science and Higher Education.

References

Alizadeh, A. A., Elsen, M. B., Davis, R. E., Ma, C. L., Lossos, I. S., Rosenwald, A., et al. (2000). Distinct types of diffuse large B-cell lymphoma identified by gene expression profiling. *Nature, 403*(6769), 503–511. doi:10.1038/35000501.

Barros, R. C., Basgalupp, M. P., De Carvalho, A. C., & Freitas, A. A. (2012). A survey of evolutionary algorithms for decision-tree induction. doi:10.1109/TSMCC.2011.2157494.

Barros, R. C., Basgalupp, M. P., Freitas, A. A., & De Carvalho, A. C. (2014). Evolutionary design of decision-tree algorithms tailored to microarray gene expression data sets. *IEEE Transactions on Evolutionary Computation, 18*(6), 873–892. doi:10.1109/TEVC.2013.2291813.

Benson, D. A., Cavanaugh, M., Clark, K., Karsch-Mizrachi, I., Ostell, J., Pruitt, K. D., et al. (2018). GenBank. *Nucleic Acids Research, 46*(D1), D41–D47. doi:10.1093/nar/gkx1094.

Blickle, T., & Thiele, L. (1996). A comparison of selection schemes used in evolutionary algorithms. *Evolutionary Computation, 4*(4), 361–394. doi:10.1162/evco.1996.4.4.361.

Breiman, L., Friedman, J. H., Olshen, R. A., & Stone, C. J. (2017). *Classification and regression trees*. doi:10.1201/9781315139470.

Brodley, C. E., & Utgoff, P. E. (1995). Multivariate decision trees. *Machine Learning, 19*(1), 45–77. doi:10.1023/A:1022607123649.

Brown, D. E., Pittard, C. L., & Park, H. (1996). Classification trees with optimal multivariate decision nodes. *Pattern Recognition Letters, 17*(7), 699–703. doi:10.1016/0167-8655(96)00033-5.

Cao, Y., & Rockett, P. I. (2015). The use of vicinal-risk minimization for training decision trees. *Applied Soft Computing Journal, 31*, 185–195. doi:10.1016/j.asoc.2015.02.043.

Chen, X., & Ishwaran, H. (2012). Random forests for genomic data analysis. doi:10.1016/j.ygeno.2012.04.003.

Chen, X., Wang, M., & Zhang, H. (2011). The use of classification trees for bioinformatics. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 1*(1), 55–63. doi:10.1002/widm.14.

Czajkowski, M., Grześ, M., & Kretowski, M. (2014). Multi-test decision tree and its application to microarray data classification. *Artificial Intelligence in Medicine, 61*(1), 35–44. doi:10.1016/j.artmed.2014.01.005.

Czajkowski, M., & Kretowski, M. (2014). Evolutionary induction of global model trees with specialized operators and memetic extensions. *Information Sciences, 288*(1), 153–173. doi:10.1016/j.ins.2014.07.051.

Czajkowski, M., & Kretowski, M. (2016). The role of decision tree representation in regression problems An evolutionary perspective. *Applied Soft Computing Journal, 48*, 458–475. doi:10.1016/j.asoc.2016.07.007.

Czajkowski, M., & Kretowski, M. (2019). A multi-objective evolutionary approach to pareto-optimal model trees. *Soft Computing, 23*(5), 1423–1437. doi:10.1007/s00500-018-3646-3.

Demsar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research 7*. doi:10.1016/j.jecp.2010.03.005.

Esposito, F., Malerba, D., & Semeraro, G. (1997). A comparative analysis of methods for pruning decision trees. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 19*(5), 476–491. doi:10.1109/34.589207.

Fayyad, U. M., & Irani, K. B. (1992). On the handling of continuous-valued attributes in decision tree generation. *Machine Learning, 8*(1), 87–102. doi:10.1023/A:1022638503176.

Frank, E., Hall, M. A., & Witten, I. H. (2017). *The WEKA workbench*. doi:10.1016/B978-0-12-804291-5.00024-6.

Ge, G., & Wong, G. W. (2008). Classification of premalignant pancreatic cancer mass-spectrometry data using decision tree ensembles. *BMC Bioinformatics, 9*. doi:10.1186/1471-2105-9-275.

Grzes, M., & Kretowski, M. (2007). Decision tree approach to microarray data analysis. *Biocybernetics and Biomedical Engineering, 27*(3), 29–42.

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The elements of statistical learning the elements of statistical learning data mining, inference, and prediction*. doi:10.1007/978-0-387-84858-7.

Hira, Z. M., & Gillies, D. F. (2015). A review of feature selection and feature extraction methods applied on microarray data. *Advances in Bioinformatics*. doi:10.1155/2015/198363.

Horng, J. T., Wu, L. C., Liu, B. J., Kuo, J. L., Kuo, W. H., & Zhang, J. J. (2009). An expert system to classify microarray gene expression data using gene selection by decision tree. *Expert Systems with Applications, 36*(5), 9072–9081. doi:10.1016/j.eswa.2008.12.037.

Hyafil, L., & Rivest, R. L. (1976). Constructing optimal binary decision trees is NP-complete. *Information Processing Letters, 5*(1), 15–17. doi:10.1016/0020-0190(76)90095-8.

Jurczuk, K., Czajkowski, M., & Kretowski, M. (2017). Evolutionary induction of a decision tree for large-scale data: A GPU-based approach. *Soft Computing, 21*(24), 7363–7379. doi:10.1007/s00500-016-2280-1.

Kar, S., Sharma, K. D., & Maitra, M. (2015). Gene selection from microarray gene expression data for classification of cancer subgroups employing PSO and adaptive K-nearest neighborhood technique. *Expert Systems with Applications, 42*(1), 612–627. doi:10.1016/j.eswa.2014.08.014.

Kotsiantis, S. B. (2013). Decision trees: A recent overview. *Artificial Intelligence Review, 39*(4), 261–283. doi:10.1007/s10462-011-9272-4.

Kotthoff, L., Thornton, C., Hoos, H. H., Hutter, F., & Leyton-Brown, K. (2016). Auto-WEKA 2.0: Automatic model selection and hyperparameter optimization in WEKA. *Journal of Machine Learning Research*. doi:10.1016/0022-1694(93)90238-5.

Lazar, C., Taminau, J., Meganck, S., Steenhoff, D., Coletta, A., Molter, C., et al. (2012). A survey on filter techniques for feature selection in gene expression microarray analysis. *IEEE/ACM Transactions on Computational Biology and Bioinformatics, 9*(4), 1106–1119. doi:10.1109/TCBB.2012.33.

Lazzarini, N., & Bacardit, J. (2017). RGIFE: A ranked guided iterative feature elimination heuristic for the identification of biomarkers. *BMC Bioinformatics, 18*(1). doi:10.1186/s12859-017-1729-2.

Liang, Y., Diehn, M., Watson, N., Bollen, A. W., Aldape, K. D., Nicholas, M. K., et al. (2005). Gene expression profiling reveals molecularly and clinically distinct subtypes of glioblastoma multiforme. *Proceedings of the National Academy of Sciences, 102*(16), 5814–5819. doi:10.1073/pnas.0402870102.

Loh, W. Y. (2014). Fifty years of classification and regression trees. *International Statistical Review, 82*(3), 329–348. doi:10.1111/insr.12016.

Long, H., Liang, C., Zhang, X., Fang, L., Wang, G., Qi, S., et al. (2017). Prediction and analysis of key genes in glioblastoma based on bioinformatics. *BioMed Research International, 2017*. doi:10.1155/2017/7653101.

Lu, H., Yang, L., Yan, K., Xue, Y., & Gao, Z. (2017). A cost-sensitive rotation forest algorithm for gene expression data classification. *Neurocomputing, 228*, 270–276. doi:10.1016/j.neucom.2016.09.077.

Ludwig, S. A., Picek, S., & Jakobovic, D. (2018). Classification of cancer data: Analyzing gene expression data using a fuzzy decision tree algorithm. In *International series in operations research and management science: 262* (pp. 327–347). doi:10.1007/978-3-319-65455-3_13.

Lv, J., Peng, Q., Chen, X., & Sun, Z. (2016). A multi-objective heuristic algorithm for gene expression microarray data classification. *Expert Systems with Applications, 59*, 13–19. doi:10.1016/j.eswa.2016.04.020.

Michalewicz, Z. (1994). *Genetic algorithms + data structures = evolution programs*. doi:10.1007/978-3-662-07418-3.

Nag, K., & Pal, N. R. (2016). A multiobjective genetic programming-based ensemble for simultaneous feature selection and classification. *IEEE Transactions on Cybernetics, 46*(2), 499–510. doi:10.1109/TCYB.2015.2404806.

Painsky, A., & Rosset, S. (2017). Cross-validated variable selection in tree-based methods improves predictive performance. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 39*(11), 2142–2153. doi:10.1109/TPAMI.2016.2636831.

Piltaver, R., Luštrek, M., Gams, M., & Martinčić-Ipšić, S. (2016). What makes classification trees comprehensible? *Expert Systems with Applications, 62*, 333–346. doi:10.1016/j.eswa.2016.06.009.

Quinlan, J. R. (1992). Learning with continuous classes. In *Proceedings of the ai'92, 5th Australian conference on artificial intelligence*. 10.1.1.34.885.

Robnik-Šikonja, M., & Kononenko, I. (2003). Theoretical and empirical analysis of ReliefF and RReliefF. *Machine Learning, 53*(1–2), 23–69. doi:10.1023/A:1025667309714.

Rokach, L., & Maimon, O. (2005). Top-down induction of decision trees classifiers - A survey. *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews, 35*(4), 476–487. doi:10.1109/TSMCC.2004.843247.

Saremi, M., & Yaghmaee, F. (2018). Improving evolutionary decision tree induction with multi-interval discretization. *Computational Intelligence, 34*(2), 495–514. doi:10.1111/coin.12153.

Singh, D., Febbo, P. G., Ross, K., Jackson, D. G., Manola, J., Ladd, C., et al. (2002). Gene expression correlates of clinical prostate cancer behavior. *Cancer Cell, 1*(2), 203–209. doi:10.1016/S1535-6108(02)00030-2.

de Souto, M. C., Costa, I. G., de Araujo, D. S., Ludermir, T. B., & Schliep, A. (2008). Clustering cancer gene expression data: A comparative study. *BMC Bioinformatics, 9*. doi:10.1186/1471-2105-9-497.

de Souto, M. C., Costa, I. G., de Araujo, D. S., Ludermir, T. B., & Schliep, A. (2019). Datasets repository. https://schlieplab.org/Static/Supplements/CompCancer/datasets.htm. [Online; accessed 15-June-2019].

Sez, J. A., Luengo, J., & Herrera, F. (2016). Evaluating the classifier behavior with noisy data considering performance and robustness: The equalized loss of accuracy measure. *Neurocomputing, 176*, 26–35. doi:10.1016/j.neucom.2014.11.086. Recent Advancements in Hybrid Artificial Intelligence Systems and its Application to Real-World Problems.

Vigano, E., Gunawardana, J., Mottok, A., Van Tol, T., Mak, K., Chan, F. C., et al. (2018). Somatic IL4R mutations in primary mediastinal large B-cell lymphoma lead to constitutive JAK-STAT signaling activation. *Blood, 131*(18), 2036–2046. doi:10.1182/blood-2017-09-808907.

Wang, C., Yu, G., Liu, J., Wang, J., Zhang, Y., Zhang, X., et al. (2012). Downregulation of PCDH9 predicts prognosis for patients with glioma. *Journal of Clinical Neuroscience, 19*(4), 541–545. doi:10.1016/j.jocn.2011.04.047.

Wang, X., Liu, X., Pedrycz, W., & Zhang, L. (2015). Fuzzy rule based decision trees. *Pattern Recognition, 48*(1), 50–59. doi:10.1016/j.patcog.2014.08.001.

Wong, T. T., & Liu, K. L. (2010). A Probabilistic mechanism based on clustering analysis and distance measure for subset gene selection. *Expert Systems with Applications, 37*(3), 2144–2149. doi:10.1016/j.eswa.2009.07.028.

Woodward, J. R. (2003). GA or GP? That is not the question. In *2003 congress on evolutionary computation, CEC 2003 - proceedings: 2* (pp. 1056–1063). doi:10.1109/CEC.2003.1299785.

Xu, F., Gao, Y., Wang, Y., Pan, J., Jianjun, S., Shao, X., et al. (2016). Decreased TSPAN1 promotes prostate cancer progression and is a marker for early biochemical recurrence after radical prostatectomy. *Oncotarget, 5*(0). doi:10.18632/oncotarget.11448.

Yi, G., Sze, S. H., & Thon, M. R. (2007). Identifying clusters of functionally related genes in genomes. *Bioinformatics, 23*(9), 1053–1060. doi:10.1093/bioinformatics/btl673.