Global Induction of Oblique Model Trees: An Evolutionary Approach

Marcin Czajkowski and Marek Kretowski

Faculty of Computer Science, Bialystok University of Technology
Wiejska 45a, 15-351 Białystok, Poland
{m.czajkowski,m.kretowski}@pb.edu.pl

Abstract. In this paper we propose a new evolutionary algorithm for global induction of oblique model trees that associates leaves with multiple linear regression models. In contrast to the typical top-down approaches it globally searches for the best tree structure, splitting hyperplanes in internal nodes and models in the leaves. The general structure of proposed solution follows a typical framework of evolutionary algorithms with an unstructured population and a generational selection. We propose specialized genetic operators to mutate and cross-over individuals (trees). The fitness function is based on the Bayesian Information Criterion. In preliminary experimental evaluation we show the impact of the tree representation on solving different prediction problems.

Keywords: data mining, evolutionary algorithms, model trees, oblique trees, global induction.

1 Introduction

Decision trees [26] are one of the most popular and frequently applied prediction technique in classification and regression problems. Regression and model trees are now popular alternatives to classical statistical techniques like standard regression or logistic regression [12].

In this paper we want to investigate a global approach to oblique model tree induction based on a specialized evolutionary algorithm. This solution extends our previous research on evolutionary univariate regression and model trees.

1.1 Background

Data mining [10] is a process of extracting useful information, relationships and hidden patterns from large databases. The tree-based approaches are gaining in popularity because they are easy to interpret, visualize and their decisions can be easily explained. The ease of application, fast operation and what may be the most important, the effectiveness of decision trees, makes them powerful and popular tool [14]. Decision trees are also applicable when the data does not satisfy rigorous assumptions required by more traditional methods [12].

L. Rutkowski et al. (Eds.): ICAISC 2013, Part II, LNAI 7895, pp. 1–11, 2013.

The problem of learning the optimal decision tree is known to be NP-complete [21]. Consequently, classical decision-tree learning algorithms are based on heuristics such as greedy top-down approach [25]. Starting from the root node it searches for the locally optimal split (test) according to the given optimality measure. Next, the training data is redirected to newly created nodes. This procedure is recursively repeated until stopping criteria are met. Finally, the post-pruning is applied to improve generalization power of the predictive model.

Most of the tree inducing algorithms partition the feature space with axis-parallel hyperplanes. These types of trees are called univariate decision trees since each split in non-terminal node involves a single feature. For continuous-valued features usually inequality tests with binary outcomes are applied and for nominal features mutually exclusive groups of feature values are associated with the outcomes. When more than one feature is taken into account to build a test in internal node, then we deal with multivariate decision trees. The most common form of such test is an oblique split, which is based on linear combination of features (hyper-plane). The decision tree which applies only oblique tests is often called oblique or linear, whereas heterogeneous trees with univariate, linear and other multivariate (e.g., instance-based) tests can be called mixed decision trees [18]. It should be emphasized that computational complexity of the multivariate induction is generally significantly higher than the univariate induction.

Regression and model trees [12] may be considered as a variant of decision trees, designed to approximate real-valued functions instead of being used for classification tasks. The main difference between regression tree and model tree is that, in the latter, constant value in the terminal node is replaced by a regression plane. Each leaf of the model tree holds a linear (or nonlinear) model whose output is the final prediction value. One of the first and most known regression tree solutions is the *CART* system [4]. It finds a split that minimizes the sum of squared residuals and builds a piecewise constant model with each terminal node fitted by the training sample mean. The accuracy of prediction was later improved by replacing single values in the leaves by more advanced models: *M5* [29] proposed by Quinlan, induces a univariate model tree that contains at leaves multiple linear models analogous to piecewise linear functions; *HTL* presented in [28] goes further and evaluates linear and nonlinear models in the terminal nodes.

1.2 Motivation

The linear regression is a global model in which a single predictive function holds over the entire data-space [12]. However, many regression problems cannot be solved by single regression models especially when the data has many features which interact in complicated ways. Recursively partitioning the data and fitting local models to the smaller regions, where the interactions are more simple, is a good alternative to complicated, nonlinear regression approaches. Such technique is fast and generally efficient in many practical problems, but obviously does not guarantee the optimal solution. Due to the greedy nature, algorithms may not generate the smallest possible number of rules for a given problem [22]

and large number of rules results in decreased comprehensibility and often prediction accuracy. Hence, application of evolutionary algorithms (EAs) [20] to the problem of decision tree induction [2] become increasingly popular alternative because instead of local search, EAs can perform a global search in the space of candidate solutions.

In addition, simple univariate tests are convenient, however they generated trees may be complicated and inaccurate if the data is more suitably partitioned by not axel-parallel hyper-planes. Therefore in some domains, oblique trees may result in much smaller and more accurate trees.

In our previous works, we applied EAs to univariate regression [16] and model trees [6] and investigated the impact of memetic extensions [7]. In this paper we extend the GMT solution and search also for splitting hyper-planes in internal nodes. We propose a global approach called oblique Global Model Tree ($oGMT$) which finds accurate and less complex solutions to the popular, greedy counterparts. New specialized operators for the oblique split search are designed and a fitness function that penalizes the tree for over-parametrization and reflects not only the number of nodes but also the complexity of the tests is suitably defined. Previously performed research on oblique classification trees [15] showed that the oblique global algorithm managed to find more compact classifiers than the competitors.

1.3 Related Work

Multiple authors have proposed methods to limit negative effects of inducing the decision tree with greedy strategy. One of the first attempts to optimize the overall model tree error was presented in *RETRIS* [13]. The algorithm simultaneously optimizes the split and the models at the terminal nodes to minimize the global error. However *RETRIS* is not scalable and does not support larger datasets because of its huge complexity [22]. *SMOTI* [19] builds regression models not only in leaves but also in the upper parts of the tree. Authors claim that this allows for individual predictors to have both global and local effects on the model tree. In *LLRT* [30] authors search for optimal solution by a near-exhaustive evaluation of all possible splits in a node, based on the quality of fit of linear regression models in the resulting branches.

In the literature, there are attempts of applying evolutionary approach for induction of decision trees (please refer to survey [2]). In the $TARGET$ solution [9], authors propose to evolve a $CART$-like regression tree with simple operators and the Bayesian Information Criterion (BIC) [27] as a fitness function. Later solutions focus on evolving model trees with linear models: $E-Motion$ [1] and non-linear models: $GPMCC$ [23] in the leaves.

To the best of our knowledge, all evolutionary evolved regression and model trees have univariate tests in the splitting nodes. There are however, few oblique regression trees like: $SECRET$ [8] where authors show that the classification approach which bases on splitting two Gaussian mixtures can find better partitions than the $CART$ system and solution proposed by Li et al. [17] where a linear regression tree algorithm finds oblique splits using Principal Hessian Analysis.

The oblique regression and model trees are not as popular as the axis-parallel because the tests require greater computational resources and are much more difficult to interpret than the axis-parallel algorithms.

2 Global Induction of Oblique Model Trees

In this section we would like to present the *GMT* algorithm for global induction of oblique model trees. Structure of the proposed solution follows a typical framework of evolutionary algorithms [20] with an unstructured population and a generational selection.

2.1 Preliminaries

A learning set is composed of M objects: N-dimensional feature vectors $x_j = [x_{j,1}, \ldots, x_{j,N}]$ $(j = 1, \ldots, M)$, each one with defined dependent variable. The feature space could be divided into two regions by a hyper-plane:

$$H(w, \theta) = \{x : \langle w, x \rangle = \theta\}, \tag{1}$$

where $w = [w_1, \ldots, w_N]$ is a weight vector, θ is a threshold and $\langle w, x \rangle$ represents an inner product. If $\langle w, x_i \rangle - \theta > 0$, it can be said that the feature vector x_i is on the positive side of the hyper-plane $H(w, \theta)$.

A *dipole* [15] is a pair (x_i, x_j) of feature vectors. A *dipole* is called *long* if feature vectors constituting it has much different values of the dependent variable. First feature vector that constitutes dipole is randomly selected from the training objects located in the node. Remaining feature vectors are sorted decreasingly according to the differences between dependent variable values to the selected object. To find a second object that constitutes dipole we applied mechanism similar to the ranking linear selection [20] where the selection of the object depends only on its position in the sorted list and not on the actual value. We situate the feature vectors x_i and x_j on the opposite sides of the dividing hyper-plane. The hyper-plane $H(w, \theta)$ splits the dipole (x_i, x_j) if:

$$(\langle w, x_i \rangle - \theta) * (\langle w, x_j \rangle - \theta) < 0. \tag{2}$$

2.2 Representation

The oblique model trees are represented in their actual form as binary trees with splitting hyper-planes in non-terminal nodes and multiple linear models in the leaves. Each hyper-plane in the tree can be represented by a fixed-size $N + 1$-dimensional table of real numbers corresponding to the vector weight w and threshold θ. Each model in the leaf is constructed using standard regression technique [24] with objects and feature vectors associated with that node. The prediction $y_k(x_j)$ calculated for the k-th leaf and j-th object is explained by a linear combination of multiple independent variables $x_{j,1}, x_{j,2}, \ldots, x_{j,N}$:

$$y_k(x_j) = \upsilon_{k,0} + \upsilon_{k,1} * x_{j,1} + \upsilon_{k,2} * x_{j,2} + \ldots + \upsilon_{k,N} * x_{j,N} \tag{3}$$

where $\upsilon_{k,0}, \ldots, \upsilon_{k,N}$ are fixed coefficients that minimizes the sum of squared residuals of the model in the leaf k.

2.3 Initialization

In general, initial population should be generated randomly to cover the entire range of possible solutions. Due to the large solution space, initial individuals are created by applying the classical top-down algorithm, similarly to the typical approaches like *CART* and *M5*. An effective test in non-terminal node is searched based on randomly chosen *long dipole*. The recursive partitioning can finish:

- on a random depth;
- when all the training objects in node are characterized by the same predicted value (or varies only slightly [29]);
- when the number of objects in a node is lower than the predefined value (default value: 5).

Next, a multiple linear model is built at each terminal node. To prevent the EA from being trapped in local optima, initial individuals are induced on a random subsamples of the training data (10% of data, but not more than 500 objects) and the potential tests in internal nodes are searched on random subsets of features (50% of features).

2.4 Fitness Function

The specification of a suitable fitness function is one of the most important and sensitive element in design of the evolutionary algorithm. The direct minimization of the prediction error measured on the learning set usually leads to the overfitting problem. In this work we continue to use Bayesian information criterion *BIC* [27] as a fitness function [6,7]. The *BIC* formula is given by:

$$Fit_{BIC}(T) = -2 * ln(L(T)) + ln(n) * k(T), \tag{4}$$

where $L(T)$ is maximum of likelihood function of the tree T, $k(T)$ is the number of model parameters and n is the number of observations. The log(likelihood) function $L(T)$ is typical for regression models [11] and can be expressed as:

$$ln(L(T)) = -0.5n * [ln(2\pi) + ln(SS_e(T)/n) + 1], \tag{5}$$

where $SS_e(T)$ is the sum of squared residuals on the training data of the tree T.

However, when linear tests are considered, it is necessary to change the penalty for tree over-parametrization. It is rather straightforward that an oblique split based on few features is more complex than a univariate test. As a consequence, the tree complexity should not only reflect the tree size and the number of features that constitute models in the leaves, but also the complexity of the tests in the internal nodes. However, it is not easy to arbitrarily define the balance between different measures because it depends on the problem and user preferences. In such a situation we decided to define the tree complexity $k(T)$ in most flexible way to allow the user to tune its final form:

$$k(T) = \alpha * Q(T) + \beta * M(T) + \gamma * F(T), \tag{6}$$

where $Q(T)$ is the number of nodes in model tree T and $M(T)$ is the sum of the number of all features in the linear models at the leaves. Additional penalty term $F(T)$ works only for oblique trees as it sums up the number of features in multivariate tests. Default values of the parameters are: $\alpha = 2.0$, $\beta = 1.0$ and $\gamma = 1.0$, however further research to determine theirs values should be considered.

2.5 Selection and Termination Condition

Ranking linear selection [20] is applied as a selection mechanism. Additionally, in each iteration a single individual with the highest value of fitness function in current population in copied to the next one *(elitist strategy)*. The evolution terminates when the fitness of the best individual in the population does not improve during the fixed number of generations. In case of a slow convergence, maximum number of generations is also specified, which allows us to limit computation time.

2.6 Genetic Operators

Genetic operators are the two main forces that form the basis of evolutionary systems and provide a necessary diversity and novelty. Tree-based representation requires developing specialized operators corresponding to the classical mutation and cross-over. Previously performed research [15] shows that the recombination of two individuals usually has much higher destructive power and context change than the mutation, therefore the default probability to select cross-over is equal 0.2 and mutation 0.8.

Cross-over solution starts with selecting positions in two affected individuals. We propose three variants of recombination [6] that involve exchanging oblique tests in internal nodes, subtrees and branches between the nodes of individuals.

Mutation solution starts with randomly choosing the type of node (equal probability to select leaf or internal node). Next, the ranked list of nodes of the selected type is created and a mechanism analogous to ranking linear selection is applied to decide which node will be affected. Depending on the type of node, ranking takes into account the location of the internal node (internal nodes in lower parts of the tree are mutated with higher probability) and the absolute error (worse in terms of prediction accuracy leaves and internal nodes are mutated with higher probability). Previously [5,6] we have proposed several variants of mutation for univariate tests in internal nodes, models in the leaves and modifications in the tree structure (pruning the internal nodes and expanding the leaves). In this paper, we present new mutation operators for modifying oblique splits in internal nodes:

- test can be replaced by a new, effective one which is searched based on randomly chosen *long dipole*;
- hyper-plane can be modified by changing one, randomly chosen weight w_i;
- hyper-plane is shifted in such a way that it divides new, randomly chosen *long dipole*.

3 Preliminary Experimental Results

The proposed approach is evaluated on both artificial and real life datasets. It is compared only to the univariate versions of our global inducers: univariate Global Regression Tree (*uGRT*) [16] and univariate Global Model Tree (*uGMT*) [6], since in previous papers [5,6] we compared our solutions with popular counterparts. All presented results correspond to average of 10 runs. The Root mean squared error (*RMSE*) is given as the prediction error measure of the tested systems. Depending on the tree type, complexity measure can cover the tree size (*size*) and average number of features in the leaves (*model*).

The performance of proposed solution is compared on eight artificially generated datasets illustrated in Figure 1. They are variants of the datasets *split plane* and *armchair*. Each dataset can be perfectly predicted with regression trees (univariate a and b; oblique c and d) and model trees (univariate e and f; oblique g and h). In addition, all algorithms are tested on real-life datasets from UCI Machine Learning Repository [3].

Table 1 presents characteristics of investigated datasets and obtained results. Experiments performed on artificial datasets show the importance of tree representation in solving different prediction problems. Proposed solution *oGMT* has the most advanced splits in internal and the models in the leaves, therefore theoretically it is capable to perfectly predict values for each dataset. The major drawback of *oGMT* is that it requires much more generations to find good solution. In specified maximum number of generations (default: 100000) algorithm did not found optimal decisions for each of the 10 runs. Rest of the algorithms managed to find good decisions only when the tree representation was capable to do it.

Figure 2 illustrates *RMSE* and tree size for the best individual so far in evolution, for each tested algorithm on dataset *armchair (b)* and *armchair (h)*. When the data can be perfectly predicted with univariate regression trees (*armchair (b)*), *uGRT* and *uGMT* find optimal decisions almost instantly. Trees with oblique tests in internal nodes need much more iterations to perfectly predict data (over 1000) and even more to find the best tree structure. On the other side, when the *armchair (h)* with non-axis parallel decision border was investigated, application of algorithms with univariate tests *uGRT* and *uGMT* lead to their approximation by a very complicated stair-like structure. Similar situation was for the regression tree *oGRT* when it tried to predict linear regression models.

Presented in Table 1 differences between algorithms on the real-life datasets are not so visible. In general, the main benefit of application oblique tests instead of univariate is that the generated trees are much smaller. This applies to both regression and model trees. Preliminary results suggest also that the application of oblique tests may improve the prediction performance of *GRT* and *GMT* algorithms however the main drawback is the calculation time. In addition, there are two possible reasons why oblique trees have slightly lower than expected *RMSE* results on $Auto - Mpg$ dataset (*uGRT* vs *oGRT*) and *Pyrimidines* dataset (*uGMT* vs *oGMT*). The first plausible reason is the very slow convergence of the oblique regression and model trees. Because of the much

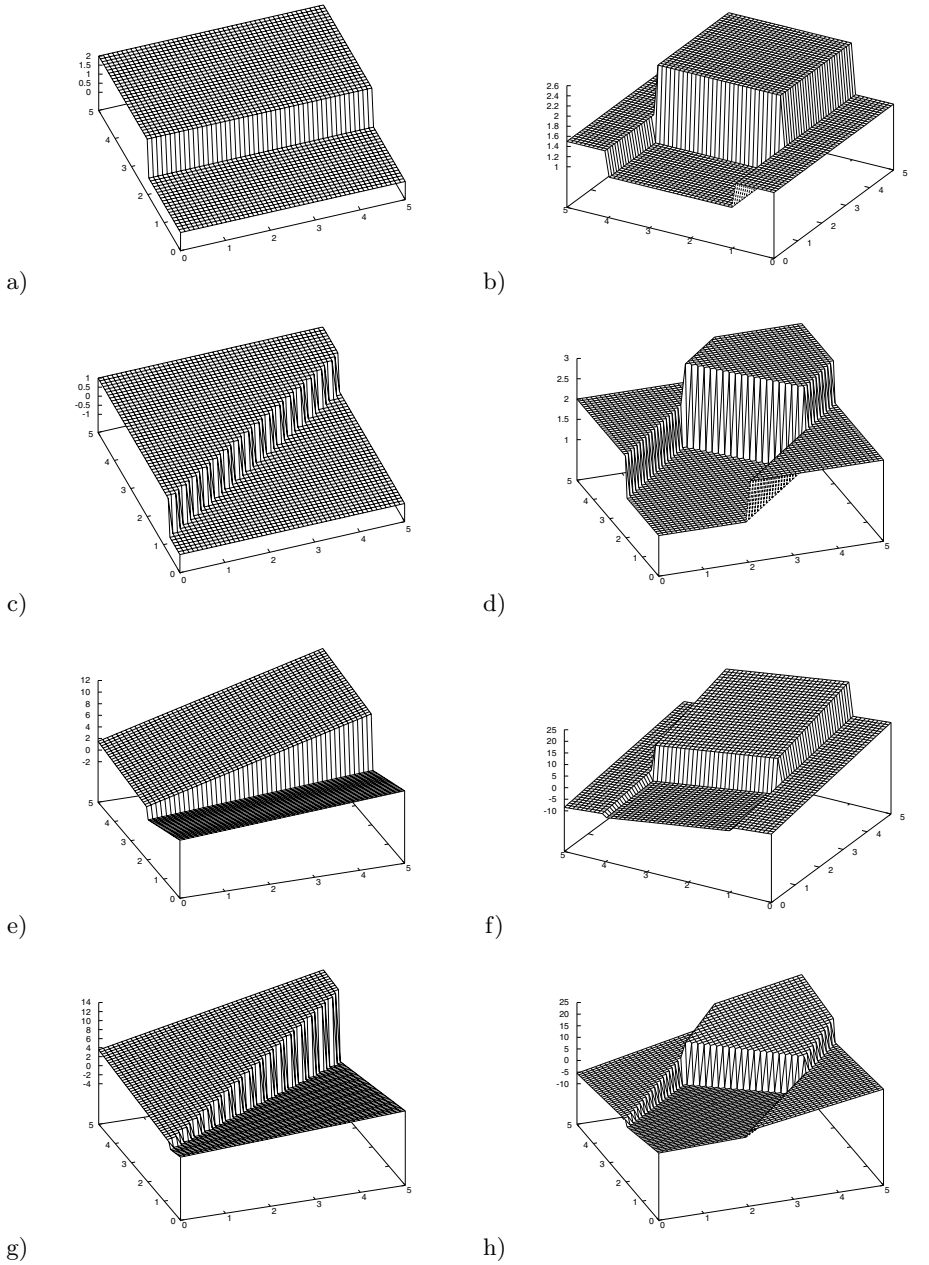

Fig. 1. Examples of artificial datasets (*split plane* - left, *armchair* - right) that can be perfectly predicted with univariate (*a*, *b*) and oblique (*c*, *d*) regression trees and univariate (*e*, *f*) and oblique (*g*, *h*) model trees

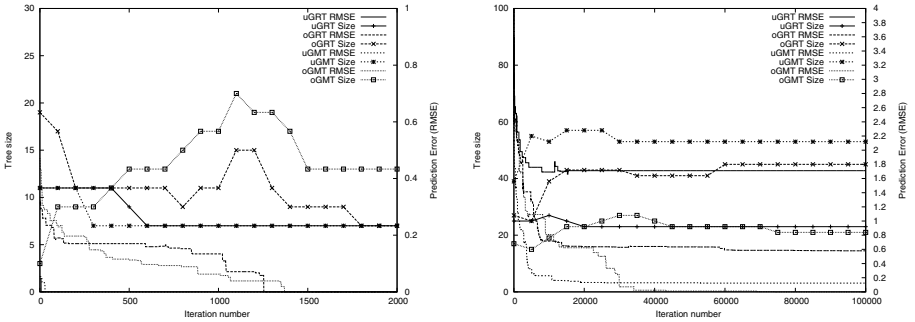

Fig. 2. Influence of the tree representation on performance of the best individual on *armchair b* (left image) and *armchair h* (right image) dataset

larger solution space the evolution could have been terminated too quickly. Second reason may refer to overfitting the oblique trees to the training data. This requires further research to determine the penalty term for over-parametrization in the fitness function.

Table 1. Characteristics of the artificial and real-life datasets (number of objects/number of numeric features/number of nominal features) and obtained results for regression trees: univariate *uGRT* and oblique *oGRT*; and for model trees: univariate *uGMT* and oblique *oGMT*

		uGRT		*oGRT*		*uGMT*			*oGMT*		
Dataset	Properties	RMSE	size	RMSE	size	RMSE	size	model	RMSE	size	model
Split plane (a)	1000/2/0	0.044	2.0	0.080	2.0	0.039	2.0	0.0	0.083	2.0	0.0
Split plane (c)	1000/2/0	0.298	13.2	0.084	2.0	0.304	14.0	0.0	0.065	2.0	0.0
Split plane (e)	1000/2/0	0.369	15.8	1.026	22.2	0.159	2.0	4.0	0.322	2.0	4.0
Split plane (g)	1000/2/0	1.369	22.6	1.093	19.9	1.380	13.4	21.6	0.612	2.2	4.2
Armchair (b)	1000/2/0	0.068	4.0	0.100	4.0	0.068	4.0	0.0	0.100	4.0	0.0
Armchair (d)	1000/2/0	0.262	17.7	0.186	5.2	0.254	18.4	0.0	0.202	6.7	0.0
Armchair (f)	1000/2/0	1.320	20.6	2.897	21.6	0.913	4.0	8.0	0.881	4.1	8.2
Armchair (h)	1000/2/0	2.320	21.5	2.539	27.9	2.019	24.5	29.7	1.300	7.7	13.0
Auto-Mpg	392/4/3	3.212	10.1	3.373	5.4	3.211	3.8	10.2	2.997	2.0	6.8
Pyrimidines	72/27/0	0.108	4.6	0.088	4.5	0.102	2.1	14.1	0.103	2.1	11.8
Triazines	186/60/0	0.146	3.8	0.140	3.1	0.144	2.4	13.7	0.138	2.1	10.7

4 Conclusion and Future Works

In the paper the global induction of oblique model trees was investigated. In contrast to classical top-down inducers, where locally optimal tests are sequentially chosen, in *GMT* the tree structure, oblique tests in internal nodes and models in the leaves are searched in the same time by specialized evolutionary algorithm. Preliminary experimental results show that the globally evolved

oblique regression and model trees are highly competitive compared to univariate counterparts. Extending the representation of tests in internal nodes open new possibilities in finding better predictions.

Proposed approach is constantly improved. Further research to determine more appropriate value of complexity penalty term in the *BIC* criterion is advised and other commonly used measures should be considered. Currently we are working on a *mixed GMT* that will be able to self-adapt structure of the tree: appropriate test in internal node (univariate or oblique) and leaf type (regression or model). On the other hand, we plan to parallelize the evolutionary algorithm and add local optimizations in order to speed-up and focus evolutionary search.

Acknowledgments. This work was supported by the grant S/WI/2/13 from Bialystok University of Technology.

References

1. Barros, R.C., Ruiz, D.D., Basgalupp, M.: Evolutionary model trees for handling continuous classes in machine learning. Information Sciences 181, 954–971 (2011)
2. Barros, R.C., Basgalupp, M.P., Carvalho, A.C., Freitas, A.A.: A Survey of Evolutionary Algorithms for Decision-Tree Induction. IEEE Transactions on Systems Man and Cybernetics, Part C 42(3), 291–312 (2012)
3. Blake, C., Keogh, E., Merz, C.: UCI Repository of Machine Learning Databases (1998), `http://www.ics.uci.edu/~mlearn/MLRepository.html`
4. Breiman, L., Friedman, J., Olshen, R., Stone, C.: Classification and Regression Trees. Wadsworth Int. Group (1984)
5. Czajkowski, M., Kretowski, M.: Globally Induced Model Trees: An Evolutionary Approach. In: Schaefer, R., Cotta, C., Kołodziej, J., Rudolph, G. (eds.) PPSN XI. LNCS, vol. 6238, pp. 324–333. Springer, Heidelberg (2010)
6. Czajkowski, M., Kretowski, M.: An Evolutionary Algorithm for Global Induction of Regression Trees with Multivariate Linear Models. In: Kryszkiewicz, M., Rybinski, H., Skowron, A., Raś, Z.W. (eds.) ISMIS 2011. LNCS, vol. 6804, pp. 230–239. Springer, Heidelberg (2011)
7. Czajkowski, M., Kretowski, M.: Does Memetic Approach Improve Global Induction of Regression and Model Trees? In: Rutkowski, L., Korytkowski, M., Scherer, R., Tadeusiewicz, R., Zadeh, L.A., Zurada, J.M. (eds.) EC 2012 and SIDE 2012. LNCS, vol. 7269, pp. 174–181. Springer, Heidelberg (2012)
8. Dobra, A., Gehrke, J.: SECRET: A Scalable Linear Regression Tree Algorithm. In: Proc. of KDD (2002)
9. Fan, G., Gray, J.B.: Regression tree analysis using target. Journal of Computational and Graphical Statistics 14(1), 206–218 (2005)
10. Fayyad, U., Piatetsky-Shapiro, G., Smyth, P., Uthurusamy, R. (eds.): Advances in Knowledge Discovery and Data Mining. AAAI Press (1996)
11. Gagne, P., Dayton, C.M.: Best Regression Model Using Information Criteria. Journal of Modern Applied Statistical Methods 1, 479–488 (2002)
12. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning. Data Mining, Inference, and Prediction, 2nd edn. Springer (2009)
13. Karalic, A.: Linear Regression in Regression Tree Leaves. International School for Synthesis of Expert Knowledge, Bled, Slovenia (1992)

14. Kotsiantis, S.B.: Decision trees: a recent overview. Artificial Intelligence Review, 1–23 (2011)
15. Kretowski, M., Grzes, M.: Global induction of oblique decision trees: An evolutionary approach, Intelligent Information Processing and Web Mining. In: Proc. of the IIS. Advances in Soft Computing, pp. 309–318 (2005)
16. Kretowski, M., Czajkowski, M.: An Evolutionary Algorithm for Global Induction of Regression Trees. In: Rutkowski, L., Scherer, R., Tadeusiewicz, R., Zadeh, L.A., Zurada, J.M. (eds.) ICAISC 2010, Part II. LNCS (LNAI), vol. 6114, pp. 157–164. Springer, Heidelberg (2010)
17. Li, K.C., Lue, H.H., Chen, C.H.: Interactive Tree-Structured Regression via Principal Hessian Directions. Journal of the American Statistical Association 95, 547–560 (2000)
18. Llorà, X., Wilson, S.W.: Mixed decision trees: Minimizing knowledge representation bias in LCS. In: Deb, K., Tari, Z. (eds.) GECCO 2004. LNCS, vol. 3103, pp. 797–809. Springer, Heidelberg (2004)
19. Malerba, D., Esposito, F., Ceci, M., Appice, A.: Top-down Induction of Model Trees with Regression and Splitting Nodes. IEEE Trans. on PAMI 26(5), 612–625 (2004)
20. Michalewicz, Z.: Genetic Algorithms + Data Structures = Evolution Programs, 3rd edn. Springer (1996)
21. Naumov, G.E.: NP-completeness of problems of construction of optimal decision trees. Soviet Physics Doklady 36(4), 270–271 (1991)
22. Potts, D., Sammut, C.: Incremental Learning of Linear Model Trees. Machine Learning 62, 5–48 (2005)
23. Potgieter, G., Engelbrecht, A.: Evolving model trees for mining data sets with continuous-valued classes. Expert Systems with Applications 35, 1513–1532 (2008)
24. Press, W.H., Flannery, B.P., Teukolsky, S.A., Vetterling, W.T.: Numerical Recipes in C. Cambridge University Press (1988)
25. Rokach, L., Maimon, O.: Top-down induction of decision trees classifiers - A survey. IEEE Transactions on Systems, Man, and Cybernetics - Part C 35(4), 476–487 (2005)
26. Rokach, L., Maimon, O.Z.: Data mining with decision trees: theory and application. Machine Perception Arfitical Intelligence 69 (2008)
27. Schwarz, G.: Estimating the Dimension of a Model. The Annals of Statistics 6, 461–464 (1978)
28. Torgo, L.: Functional Models for Regression Tree Leaves. In: Proc. of ICML, pp. 385–393. Morgan Kaufmann (1997)
29. Quinlan, J.: Learning with Continuous Classes. In: Proc. of AI 1992, pp. 343–348. World Scientific (1992)
30. Vogel, D., Asparouhov, O., Scheffer, T.: Scalable look-ahead linear regression trees. In: Proc. of 13th ACM SIGKDD, pp. 757–764. ACM Press, New York (2007)