Globally Induced Model Trees: An Evolutionary Approach

Marcin Czajkowski and Marek Krętowski

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

Abstract. In the paper we propose a new evolutionary algorithm for induction of univariate regression trees that associate leaves with simple linear regression models. In contrast to typical top-down approaches it globally searches for the best tree structure, tests in internal nodes and models in leaves. The population of initial trees is created with diverse top-down methods on randomly chosen subsamples of the training data. Specialized genetic operators allow the algorithm to efficiently evolve regression trees. Akaike's information criterion (*AIC*) as the fitness function helps to mitigate the overfitting problem. The preliminary experimental validation is promising as the resulting trees can be significantly less complex with at least comparable performance to the classical top-down counterparts.

Keywords: Model trees, evolutionary algorithms, regression trees, *AIC*, simple linear regression.

1 Introduction

Data mining [7] is a process of extracting useful information, relationships and hidden patterns in large databases. One of data mining techniques is predictive modeling also known as supervised prediction or supervised learning. The most common predictive task in data mining applications besides classification is regression. Regression and model trees are now popular alternatives to classical statistical techniques like standard regression or logistic regression [10]. The tree-based approaches are gaining in popularity because of their ease of application, fast operation and effectiveness. Additionally, the hierarchical tree structure closely resembles a human way of decision making which makes regression trees natural and easy to understand even for inexperienced analysts.

Recently many regression and model tree systems have been proposed. One of the first solutions was presented in the seminal book describing the *CART* system [4]. *CART* algorithm finds a split that minimizes the sum of squared residuals of the model when predicting and builds a piecewise constant model with each terminal node fitted by the training sample mean. In the next years multiple authors improve upon the accuracy of regression trees by replacing the

R. Schaefer et al. (Eds.): PPSN XI, Part I, LNCS 6238, pp. 324–333, 2010.

single predicted values in the leaves by more advanced models (e.g. linear). *M5* [20], *SECRET* [6], *SMOTI* [15] or *RT* [21] are some of the model tree algorithms that have been proposed.

All aforementioned systems induce regression and model trees in a top-down approach. Starting from the root node they search for the locally optimal split (test) according to the given optimality measure and then the training data is redirected to newly created nodes. This procedure is recursively repeated until the stopping criteria are met. Finally, the post-pruning is applied to improve the generalization power of the predictive model. Such a greedy technique is fast and generally efficient in many practical problem, but obviously does not guarantee the globally optimal solution. It can be expected that a more global induction could be more adequate in certain situations.

In this paper we want to investigate a global approach to model tree induction based on a specialized evolutionary algorithm. Our work covers the induction of univariate regression tree with simple linear models in leaves. The proposed solution may be applied to the problems that are primarily concerned with the regression of an outcome onto a single predictor. As an example the original genetic epidemiology problem required only consideration of simple linear regression models like [19] to locate genes associated with a quantitative trait of interests. There are other systems that associate leaves with simple linear regression like the one described by Alexander and Grimshaw [1] called *Treed Regression*.

Previously performed research showed that evolutionary inducers are capable to efficiently induce various types of classification trees: univariate [11], oblique [12] and mixed [13]. In our last paper we applied a similar approach to obtain accurate and compact regression trees [14]. In this work we would like to extend standard regression trees by replacing single predicted values (means) in leaves by simple linear models. Additionally, the search for an optimal structure was modified and now is driven by the Akaike's information criterion (*AIC*) [2].

The rest of the paper is organized as follows. In the next section a new evolutionary algorithm for global induction of univariate model trees is described. Experimental validation of the proposed approach on artificial and real-life data is presented in section 3. In the last section, the paper is concluded and possible future works are sketched.

2 Evolutionary Induction of Model Trees

In the proposed approach the general structure of the system follows a typical framework of evolutionary algorithms [16] with an unstructured population and a generational selection.

2.1 Representation

Model trees are represented in their actual form as classical univariate trees. Each test in a non-terminal node concerns only one attribute (nominal or continuous valued). Additionally, in every node information about learning vectors

associated with the node is stored. This enables the algorithm to perform more efficiently local structure and tests modifications during applications of genetic operators.

In a case of a continuous-valued feature typical inequality tests are applied. As potential splits only precalculated candidate thresholds are considered. A candidate threshold for the given attribute is defined as a midpoint between such a successive pair of examples in the sequence sorted by the increasing value of the attribute, in which the examples are characterized by different predicted values. Such a solution significantly limits the number of possible splits and focuses the search process. For a nominal attribute at least one value is associated with each branch. It means that an inner disjunction is built into the induction algorithm.

A simple linear model is calculated at each terminal node of the model tree using standard regression technique [17]. A dependent variable Y is modeled as a linear function of single variable X:

$$Y = \beta_0 + \beta_1 * X \tag{1}$$

where X is one of the independent variables, β_0 is the intercept and β_1 is the slope of the regression line that minimizes the sum of squared residuals of the model.

2.2 Initialization

Like in *M5* approach [20] we first learn a standard regression tree (with constants in the leaves) and only afterwards we turn it into a model tree. Initial individuals are created by applying the classical top-down algorithm to randomly chosen subsamples of the original training data (10% of data, but not more than 500 examples). Additionally, for every initial tree one of three test search strategies in non-terminal nodes is applied. Two strategies come from the very well-known regression tree systems i.e. *CART* [4] and *M5* [20] and they are based on *Least Squares* or *Least Absolute Deviation*. The last strategy is called *dipolar*, where a pair of feature vectors (dipole) is selected and then a test is constructed which splits this dipole. Selection of the dipole is randomized but longer (with bigger difference between dependent variable values) dipoles are preferred and mechanism similar to the ranking linear selection [16] is applied. The recursive partitioning is finished when all training objects in a node are characterized by the same predicted value (or it vary only slightly [20]), the number of objects in a node is lower than the predefined value (default value: 5) or the maximum tree depth is reached (default value: 10).

One of two search strategies of predicted variable used in linear model at leaves is applied. First one calculates simple linear regression model for each attribute and applies the one that minimizes the sum of squared residuals of the linear regression model. In second strategy the simple linear model is built from training objects in this leaf on the randomly chosen independent variable.

2.3 Termination Condition

When the fitness of the best individual in the population does not improve during the fixed number of generations (default value is equal 1000) the evolution terminates. Additionally maximum number of generations is specified, which allows limiting the computation time in case of a slow convergence (default value: 5000).

2.4 Genetic Operators

In our previous paper [14] we have presented two specialized genetic operators corresponding to the classical mutation and cross-over. We have extended them due to simple linear model in leaves.

Application of both operators can result in changes of the tree structure, tests in non-terminal and models in terminal nodes. After applying any operator it is usually necessary to relocate learning vectors between parts of the tree rooted in the altered node. This can cause that certain parts of the tree does not contain any learning vectors and has to be pruned.

Mutation operator. A mutation-like operator is applied with a given probability to a tree (default value is 0.8) and it guarantees that at least one node of the selected individual is mutated. Firstly, the type of the node (leaf or internal node) is randomly chosen with equal probability and if a mutation of a node of this type is not possible, the other node type is chosen. A 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. While concerning internal nodes, the location (the level) of the node in the tree and the quality of the subtree starting in the considered node are taken into account. It is evident that modification of the test in the root node affects whole tree and has a great impact, whereas mutation of an internal node in lower parts of the tree has only a local impact. In the proposed method, nodes on higher levels of the tree are mutated with lower probability and among nodes on the same level the absolute error calculated on the learning vectors located in the subtree is used to sort them. As for leaves, only absolute error is used to put them in order, but homogenous leaves are not included. As a result, leaves which are worse in terms of accuracy are mutated with higher probability.

Modifications performed by mutation operator depend on the node type (i.e. if the considered node is a leaf node or an internal node). For a non-terminal node a few possibilities exist:

- A completely new test can be found by means of the dipolar method used for the initialization;
- The existing test can be altered by shifting the splitting threshold (continuous-valued feature) or re-grouping feature values (nominal features);
- A test can be replaced by another test or tests can be interchanged;
- One sub-tree can be replaced by another sub-tree from the same node;
- A node can be transformed (pruned) into a leaf.

After performed mutation in internal nodes the models in corresponding leaves are not recalculated for performance reasons. However, adequate linear models can be found while performing the leaves mutations. Modifying a leaf makes sense only if it contains objects with different dependent variable values. For a terminal node two possibilities exists:

- The leaf is transformed into an internal node and a new test is chosen in the aforementioned way;
- Simple linear model is replaced by other one that is calculated on different predictor variable.

Cross-over operator. In the proposed solution there are three variants of recombination. All of them start with selecting of cross-over positions in two affected individuals. One node is chosen randomly in each of two trees. In the most straightforward variant, the subtrees starting in the selected nodes are exchanged. This corresponds to the classical cross-over from genetic programming. In the second variant, which can be applied only when non-internal nodes are randomly chosen and the numbers of outcomes are equal, only tests associated with the nodes are exchanged. The third variant is also applicable only when non-internal nodes are drawn and the numbers of descendants are equal. Branches which start from the selected nodes are exchanged in random order.

2.5 Selection

As a selection mechanism the ranking linear selection is applied. Additionally, the individual with the highest value of the fitness function in the iteration is copied to the next population *(elitist strategy)*.

2.6 Fitness Function

A fitness function drives the evolutionary search process and is very important and sensitive component of the algorithm. When concerning any prediction task it is well-known that the direct minimization of the prediction error measured on the learning set leads to an overfitting problem. In a typical top-down induction of decision trees, the over-specialization problem is partially mitigated by defining a stopping condition and by applying a post-pruning. In our previous work [14] the search for an optimal structure was embedded into the evolutionary algorithm by incorporating a complexity term into the fitness. This term worked as a penalty for increasing the tree size, however, there were no optimal value of it for all possible datasets.

In presented approach, we decided to use Akaike's information criterion (AIC) [2] as the fitness in the search for an optimal structure. This measure of the goodness of fit of an estimated statistical model works as a penalty for increasing the tree size.

The fitness function is minimized and for binary regression tree models has the following form:

$$Fitness_{AIC}(T) = -2 * ln(L(T)) + 2 * k(T) \tag{2}$$

where $L(T)$ is the maximum of the likelihood function of the tree T and $k(T)$ is the number of model parameters in the tree. Log(likelihood) function $L(T)$ is typical for regression models [9] and can be expressed as

$$ln(L(T)) = -0.5n * [ln(2\pi) + ln(SS_e(T)/n) + 1] \tag{3}$$

where $SS_e(T)$ is the sum of squared residuals of the tree T and n is the number of observations. The term, $2 * k(T)$ can also be viewed as a penalty for over-parametrization. In our approach we set $k(T) = Q(T) + 1$ in AIC criterion where $Q(T)$ is equal a number of terminal nodes in model tree T. Then, the fitness function is:

$$Fitness_{AIC}(T) = n(ln(2\pi) + ln(SS_e(T)/n(T)) + 1) + 2(Q(T) + 1) \tag{4}$$

In [10] authors suggested that the effective number of parameters estimated is actually much higher than $Q(T) + 1$ due to the split rule selections that were made during the T tree construction process. However, higher $k(T)$ value in AIC criterion leads to the smaller trees with less predictive accuracy. Further research to determine the appropriate value of complexity penalty term in the AIC criterion for proposed solution is required and other commonly used measures such as Bayesian information criterion (*BIC*) [18] or structural risk minimization (*SRM*) [5] should be considered.

3 Experimental Validation

Validation of the global approach to induction of model trees (denoted in tables as *GMT*) was performed on synthetical and real-life datasets. Obtained results are compared with two model trees and two regression trees. For the purpose of comparison, we have implemented the top-down regression model with simple linear regression in each leaf alike to *Treed Regression* [1] algorithm (denoted as *TR*). However, for better performance we have improved pruning to the *AIC* cost-complexity pruning proposed in [22]. We also present the results for more advanced model tree *M5* [20] proposed by Quinlan.

For real-life datasets results obtained by the classical top-down inducer *REP-Tree*, which is publicly available in the *Weka* system [8], are also presented. *REP-Tree* builds a regression tree using variance and prunes it using reduced-error pruning (with backfitting). Finally, we enclose the results of global induction of regression tree approach (denoted as *GRT*) from our previous work [14].

Each system run with default values of parameters. All results presented in the table correspond to averages of 10 runs and were obtained by using test sets (when available) or by 10-fold cross-validation. The average number of nodes is given as a complexity measure of regression and model trees.

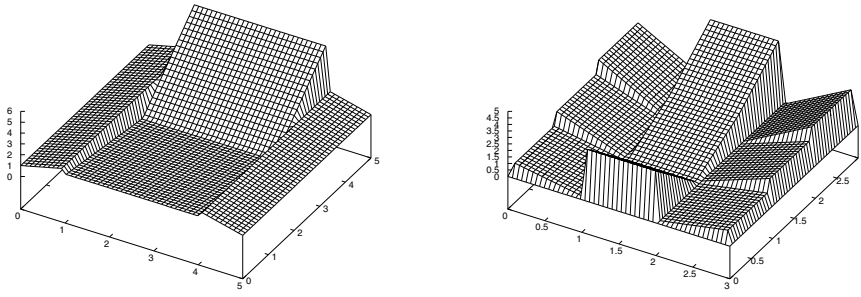

Fig. 1. Examples of artificial datasets (*armchair2* - left, *ski jump* - right)

Synthetical datasets. First group of experiments was performed on two simple artificially generated datasets with analytically defined decision borders. Both datasets contain a dependent feature that is linearly dependent with one of two independent features. One thousand observations for each dataset were divided into a training set (33.3% of observations) and testing set (66.7%).

Illustrated in figure 1 *armchair2* function is defined as:

$$g(x,y) = \begin{cases} x+1 & x \in [0,1] \\ -x-6 & x \in [4,5] \\ -0.5y+1.5 & x \in [1,4],\ y \in [0,3] \\ 3y-9 & x \in [1,4],\ y \in [3,5] \end{cases} \tag{5}$$

and the *ski jump* function:

$$g(x,y) = \begin{cases} -x+1 & x \in [0,1],\ y \in [0,1] \\ -2x+2 & x \in [0,1],\ y \in [1,2] \\ -3x+3 & x \in [0,1],\ y \in [2,3] \\ -4y+4 & x \in [1,2],\ y \in [0,1] \\ 2y-2 & x \in [1,2],\ y \in [1,2] \\ 3y-4 & x \in [1,2],\ y \in [2,3] \\ x-2 & x \in [2,3],\ y \in [0,1] \\ 2x-4 & x \in [2,3],\ y \in [1,2] \\ 3x-6 & x \in [2,3],\ y \in [2,3] \end{cases} \tag{6}$$

Table 1. Results obtained on the synthetical datasets *armchair2* and *ski jump*. Root mean squared error (RMSE) is given as the error measure and number of nodes as the tree size.

	GMT		*M5*		*TR*	
Dataset	RMSE	Tree size	RMSE	Tree size	RMSE	Tree size
armchair2	0.12	6.5	0.37	11.0	0.35	24.0
ski jump	0.30	10.9	0.61	26.0	0.54	25.0

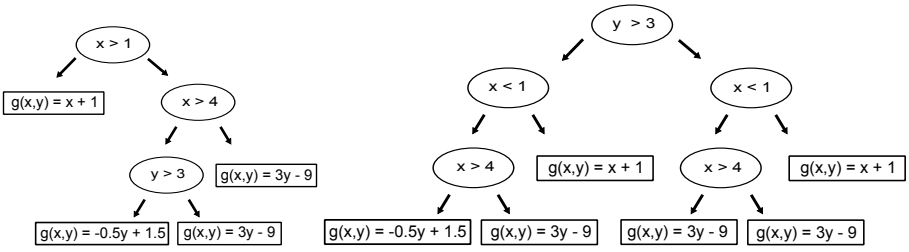

Fig. 2. Examples of model trees for *armchair2* (*GMT* - left, *TR* - right) from the experiment

Table 1 presents obtained results only for model trees as the regression trees on those synthetic datasets are not competitive due to the training sample mean in terminal nodes.

It should be noticed that in both cases trees obtained by *GMT* have optimal or almost optimal structure and gain very small error. For the top-down inducers both problems were too difficult. *M5* and *TR* generated overgrown trees and as a result the testing error is higher. Additionally, the reason why *M5* model tree performed lower than *TR* is that the *M5* tried to use both independent features in linear model at leaves.

The advantage of the global approach can be observed in the in figure 2 where the optimal model trees for *GMT* and *TR* on the first dataset *armchair2* are illustrated. We can see that the first split which minimizes the sum of squared residuals ($y < 3$) is not optimal as it leads to overgrown tree.

Real-life datasets. In the second series of experiments, several datasets taken from UCI Machine Learning Repository [3] or provided by L. Torgo on his website are analyzed to assess the performance of the proposed system in solving real-life problems. Table 2 presents characteristics of investigated datasets and obtained results. More complex model trees like *M5* or *SMOTI* were not included in these experiments as in this paper we are focussing on comparing the improvement of global induction for regression trees and model trees that associate leaves with simple linear regression. We plan to extend evolving model trees with multivariate linear regression so it will be possible to compare with more advanced modeling trees.

It can be observed that the prediction accuracy of model trees that associate leaves with simple linear regression models is comparable to regression trees that have training sample mean in terminal nodes. It is not surprising that size of the trees are smaller in favor to model trees however, it should be noticed that globally induced trees are less complex. Proposed solution *GMT* performed better on 8 out of 10 datasets comparing to *TR* and 9 out of 10 comparing to *REPTree* in term of accuracy. Tree size for *GMT* was lower on almost all datasets.

Table 2. Characteristics of the real-life datasets (number of objects/number of numeric features/number of nominal features) and obtained results. Root mean squared error (RMSE) is given as the error measure and number of nodes as the tree size.

		GMT		*TR*		*GRT*		*REPTree*	
Dataset	Properties	RMSE	size	RMSE	size	RMSE	size	RMSE	size
Abalone	4177/7/1	2.297	7.7	2.636	10.9	2.314	51.8	2.358	201
Auto-Mpg	392/4/3	3.434	9.9	3.670	73.5	3.572	45.4	3.646	94
Auto-Price	159/17/10	2507.6	3.7	2433.9	9.9	2618.9	13.8	2760.5	32
Delta Ailerons	7129/6/0	0.000178	11.1	0.000185	7.2	0.000179	82.6	0.000175	291
Delta Elevators	9517/6/0	0.00150	9.3	0.00157	16.2	0.00148	78.3	0.00150	319
Housing	506/14/0	4.322	9.1	4.495	11.8	4.126	32.3	4.84	41
Machine CPU	209/7/0	67.53	3.8	78.18	11.3	63.99	14.8	92.34	15
Pyrimidines	74/28/0	0.1090	4.52	0.0987	5.8	0.1011	10.7	0.1355	1.0
Triazines	186/61/0	0.1405	4.7	0.1564	3.9	0.1387	13.7	0.1517	7.0
Wisconsin Cancer	194/32/0	34.33	3.1	35.13	1.9	39.22	16.3	35.88	9.0

4 Conclusion

In the paper a new global approach to model tree learning is presented. In contrast to classical top-down inducers, where locally optimal tests are sequentially chosen, both the tree structure, tests in internal nodes and models in leaves are searched in the same time by specialized evolutionary algorithm. This way the inducer is able to avoid local optima and to generate better predictive model. Even preliminary experimental results show that the globally evolved regression models could be competitive compared to the top-down based counterparts, especially in term of tree size.

The presented approach is constantly improved and currently we are working on introducing oblique tests in the non-terminal nodes. On the other hand, we plan to extend the knowledge representation by evolving model trees with multivariate linear regression. However, the proposed solution may be applied to the problems that are primarily concerned with the regression of an outcome onto a single predictor.

Acknowledgments. This work was supported by the grant W/WI/3/2010 from Bialystok University of Technology.

References

1. Alexander, W.P., Grimshaw, S.D.: Treed Regression. Journal of Computational and Graphical Statistics 5, 156–175 (1996)
2. Akaike, H.: A New Look at Statistical Model Identification. IEEE Transactions on Automatic Control 19, 716–723 (1974)
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. Cherkassky, V., Mulier, F.: Learning from Data: Concepts, Theory and Methods. Wiley, New York (1998)
6. Dobra, A., Gehrke, J.: SECRET: A Scalable Linear Regression Tree Algorithm. In: Proc. KDD 2002 (2002)
7. Fayyad, U., Piatetsky-Shapiro, G., Smyth, P., Uthurusamy, R. (eds.): Advances in Knowledge Discovery and Data Mining. AAAI Press, Menlo Park (1996)
8. Frank, E., et al.: Weka 3 - Data Mining with Open Source Machine Learning Software in Java. University of Waikato (2000), `http://www.cs.waikato.ac.nz/~ml/weka`
9. Gagne, P., Dayton, C.M.: Best Regression Model Using Information Criteria. Journal of Modern Applied Statistical Methods 1, 479–488 (2002)
10. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning. In: Data Mining, Inference, and Prediction, 2nd edn. Springer, Heidelberg (2009)
11. Krętowski, M., Grześ, M.: Global Learning of Decision Trees by an Evolutionary Algorithm. In: Information Processing and Security Systems, pp. 401–410. Springer, Heidelberg (2005)
12. Krętowski, M., Grześ, M.: Evolutionary Learning of Linear Trees with Embedded Feature Selection. In: Rutkowski, L., Tadeusiewicz, R., Zadeh, L.A., Żurada, J.M. (eds.) ICAISC 2006. LNCS (LNAI), vol. 4029, pp. 400–409. Springer, Heidelberg (2006)
13. Krętowski, M., Grześ, M.: Evolutionary Induction of Mixed Decision Trees. International Journal of Data Warehousing and Mining 3(4), 68–82 (2007)
14. Krętowski, 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. LNCS (LNAI), vol. 6114, pp. 157–164. Springer, Heidelberg (2010)
15. 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)
16. Michalewicz, Z.: Genetic Algorithms + Data Structures = Evolution Programs, 3rd edn. Springer, Heidelberg (1996)
17. Press, W.H., Flannery, B.P., Teukolsky, S.A., Vetterling, W.T.: Numerical Recipes in C. Cambridge University Press, Cambridge (1988)
18. Schwarz, G.: Estimating the Dimension of a Model. The Annals of Statistics 6, 461–464 (1978)
19. Shannon, W.D., Province, M.A., Rao, D.C.: Tree-Based Models for Fiting Stratified Linear Regression Models. Journal of Classification 19, 113–130 (2002)
20. Quinlan, J.: Learning with Continuous Classes. In: Proc. AI 1992, pp. 343–348. World Scientific, Singapore (1992)
21. Torgo, L.: Inductive Learning of Tree-based Regression Models. Ph.D. Thesis, University of Porto (1999)
22. Xiaogang, S., Morgan, W., Juanjuan, F.: Maximum Likelihood Regression Trees. Journal of Computational and Graphical Statistics 13(3), 586–598 (2004)