Production Systems and Information Engineering
Volume 4 (2006), pp. 115-124

USING REGRESSION TREES IN PREDICTIVE MODELLING

TAMÁS FEHÉR
University of Miskolc, Hungary
Department of Information Engineering
feher@ait.iit.uni-miskolc.hu

[Received November 2005 and accepted May 2006]

Abstract. Tasks where the value of an unknown parameter has to be estimated are typical in data mining projects. The solution of this kind of problems requires the creating of a predictive model. There are several methods for the solution of this type of tasks, e. g. the decision trees and the regression. In the paper an algorithm is demonstrated which combines the properties of the two techniques. Tests on different datasets show the efficiency of the method compared to the results which were provided by the traditional regression and decision tree algorithms.

Keywords: data mining, predicitve modelling, decision tree, regression

1. Introduction

Databases contain a lot of hidden information that can be used by business decisions. During the classification and prediction the value of an unknown parameter is predicted. While in the classification tasks the value of a categorical variable has to be defined, in the prediction tasks the value of a continuous variable is estimated. The model which gives estimation for the value of the target variable by the help of the predictor parameters is named predictive model. The predictive models build up by the help of a training dataset. Their use-value reveals itself when they are able to give estimation from samples which they have never seen. Several techniques are used in the predictive modelling e. g. neural networks, regression, decision trees [8][9]. This paper concentrates on the last two methods.

The traditional decision trees serve basically for the solution of classifications tasks. In each leaf there is a class label that shows, which class the questionable object falls in. However, there are decision tree methods, which make possible the estimating of a continual variable, in such a way that they put a regression model in the leaves. These complex models are called *regression trees*.

The first regression tree method is the CART algorithm [1], which puts a constant function to the leaves. During the years, further algorithms were developed, which put linear functions into the leaves for the more efficient estimation of the continuous value (M5[2], TSIR[3], RETIS[4], SMOTI[5], HTL[6], SECRET[7]).

The difference between the algorithms is in the induction of the tree. The most important factor is how the algorithm finds the split points. The most of the algorithms use a variance-based approach for the measure of the quality of cutting. In the original algorithm of CART the best split in a node is the one that minimizes the excepted impurity I_{exp}, given by the formula:

$$I_{exp} = p_l I_l + p_r I_r, \tag{1}$$

where p_l and p_r denote the probabilities of transition into the left and the right side of a split and I_l and I_r are the corresponding impurities, resting on variance of the children nodes. The variance in a node can be computed by the next formula:

$$\sigma^2(E) = \frac{1}{W(E)} \sum_{e_i \in E} w_i (y_i - \mu(E))^2 \tag{2}$$

where E denotes the set of elements in a node, w_i is the weight of the i^{th} element and $W(E) = \sum_{e_i \in E} w_i$ The y_i is the value of the i^{th} element and $\mu(E)$ denotes mean class value of E.

2. Regression models in the leaves

In [4] it is recognized that in the case of using linear regression in the leaves of the tree, it is not sure, that the goodness of the split has to be judged by the variance. Figure 1 shows, although the variance would be minimal by the cut 'a', nevertheless it seems to be reasonable to bring in a new measure of goodness and to accomplish the split in the point 'b'

According to the above mentioned aspects, the measure of the cut's goodness is the fitting error of the regression model on the left and right side. In the next formula:

$$I(E) = \frac{1}{N_l + N_r} \left[\frac{1}{N_l} \sum_{e_i \in E_l} (y_i - g_l(\vec{x}_i))^2 + \frac{1}{N_r} \sum_{e_j \in E_r} (y_j - g_r(\vec{x}_j))^2 \right] \tag{3}$$

I(E) denotes the measure of a cut's goodness. N_l and N_r are the number of elements in the left and the right side of the cut, E_l and E_r denote the set of elements in the left and the right side. Function g_l and g_r represent the regression plane through elements in the left child and in the right child. The actual values of the elements are denoted by y_i and y_j.

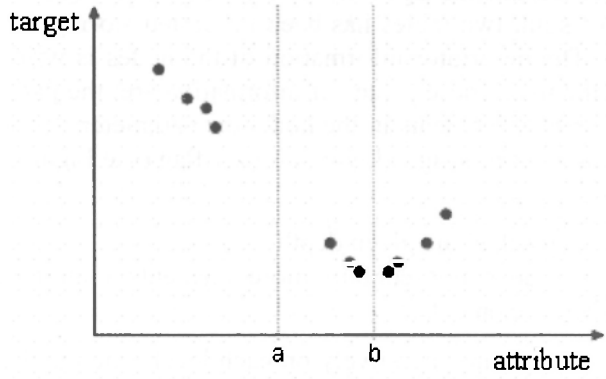


Figure 1. The appropriate split is in the point 'b'

The disadvantage of this method is the huge computing time. In the original paper the author used this algorithm on datasets which contained only few observations. Considering that the measure of cut's goodness is the fitting error of the regression models on the two sets, theoretically all of the splits would have to be executed along all of the dimensions in order to decide, what is the best split on a given node. For example on a dataset, which contains 10^4 observations and which has 10 attributes, about 10^5 splits would have to be performed. This would mean building of $2x10^5$ models (only for one level). In the case of a tree with five levels computing of about 10^6 regression models would be necessary. It is very high number, especially if it is considered that a real-life database can be substantially bigger, having regard of number of elements and dimensionality.

The algorithm presented in this paper extends the idea demonstrated in [4]. In order to building-up the tree finishes in feasible time, in the new algorithm is not accomplished all of the cuts. If the cutting is put in at only every 10^{th}, 100^{th} or 1000^{th} value, and thereby the number of splits along a dimension is reduced to order of magnitude of 10. At price of this compromise may be made not the best, but a good model. This is supported by the running results (see Chapter 4).

3. Regression Tree Algorithm

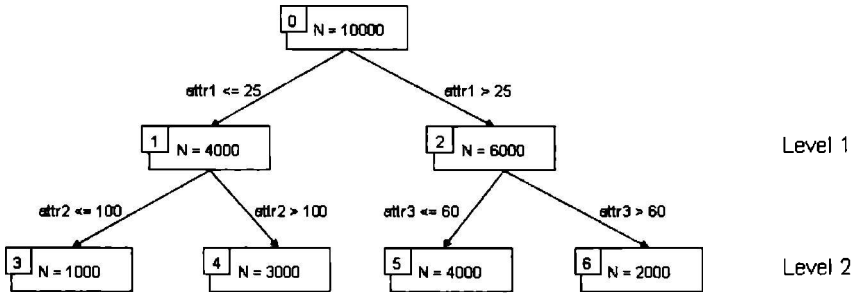
The core of the algorithm is a process, which carries out the given splits along every dimension (the number of splits is determined as an input parameter) and then chooses the split along which cutting the dataset the two regression hyperplanes can be fit with the minimal error (the average error as the measure of the split's goodness is the average of mean square errors weighted by the number

of elements on the left and the right side). So, it makes the regressions to each split. Choosing the best split, two tables has been generated storing the elements of the two child-nodes. The descriptor information of the nodes is written into a special table describing the tree. These pieces of information are: the path from the root to the node, the number of elements in the node, the parameters of the regression and if the node is a leaf then it is signed with a binary flag (see Figure 2). The node is a leaf, if

(i) the tree reached the given depth,
(ii) the leaf does not contain more elements than twice the minimal number of elements.

This process is accomplished iteratively on each level. The number of iterations is equal to the number of nodes on the former level. The input of the procedure is a node from the former level in each iteration step.

The output is a table (see Figure 2), and it contains all information, which is necessary to building-up of the tree.



ID	Leaf	Rule	N	Intercept	Param1	Param2	
1	0	attr1 <= 25	4000	123.456	0.789	2.345	
2	0	attr1 > 25	6000	234.567	1.234	5.432	
3	1	attr1 <= 25 & attr2 <= 100	1000	87.654	0	0.987	
4	1	attr1 <= 25 & attr2 > 60	3000	34.567	0.456	0.023	
5	1	attr1 > 25 & attr3 <= 100	4000	54.321	2.345	0	
6	1	attr1 > 25 & attr3 > 60	2000	12.345	0.456	0.789	

Figure 2. Example illustrating the algorithm

The algorithm achieves a stepwise linear regression in any cases. In the stepwise method, variables are added one by one to the model, and the F statistic for a

variable to be added must be significant at a given level (SLENTRY). After a variable is added, however, the stepwise method looks at all the variables already included in the model and deletes any variable that does not produce an F statistic significant at a determined level (SLSTAY). Only after this check is made and the necessary deletions accomplished can another variable be added to the model. The stepwise process ends, when none of the variables outside the model has an F statistic significant at the given SLENTRY level and every variable in the model is significant at the SLSTAY level, or when the variable to be added to the model is the one just deleted from it.

The algorithm is implemented in SAS Base [10]. The benefit of this environment is the speed (among others): the running time of a stepwise regression on a dataset containing more ten thousands of observations, is a split seconds.

3.1. The assignation of cutting points

The assignation of cutting points works by the undermentioned algorithm.

Input: minimal number of elements in a leaf (MIN), the number of intervals through cuttings, i. e. number of cuts + 1 (N), dimension (DIM)

Step 1: sorting the values along dimension DIM
Step 2: detaching first elements from the list in size of MIN
Step 3: detaching last elements from the list in size of MIN
Step 4: dividing the remainder range to N-2 parts

The algorithm handles exceptions, as follows:

Exception 1: If there are less available values than N, then these values turn split bounds, except when thereby it is not possible to make groups with MIN size in the forepart and in the end of the dimension. In this case the algorithm moves forward to the interior of dimension and tries to assign the next value as lower/upper bound. This property guarantees for example the recognition a binary attribute.

Exception 2: If the number of elements is less than 2*MIN, it does not begin to split dimensions, because in this case it is not possible to create two nodes with MIN size.

Exception 3: If lots of elements belong to a given value along a dimension, exceptions can occur, too. For example there is a database, where half of the customers has 0 dollar on the account and only another half of the customers possesses an account with positive balance. Supposing that after cutting of the first MIN pieces customer, the split point is in the middle of value 0. Since simultaneously only one dimension is in the focus, it cannot be distinguished

between customers with balance 0. The algorithm recognizes this situation and shifts the split points.

Ergo the algorithm stands for working well-balanced, namely it makes all of a size cuts (excluding exceptions, and except the front and the end of dimensions).

4. Experimental Results

There are results from running the algorithm on three different datasets. In case of each dataset the target variable is estimated with regression at first, then with a decision tree algorithm implemented by SAS Enterprise Miner and with the regression tree algorithm at last. The average relative error of the predicted values is chosen for the measure of predictive power of the different methods.

10 percent of the data was separated for testing in case of each dataset. Further 10 percent was separated for the validation in case of SAS decision tree.

The first dataset was artificially generated. In this dataset the number of dimensions is 3 and the number of records is 100. The continuous target variable can be estimated by two predictor variable. One of them is a binary and the other is an interval variable. The dataset can be shown on Figure 3.

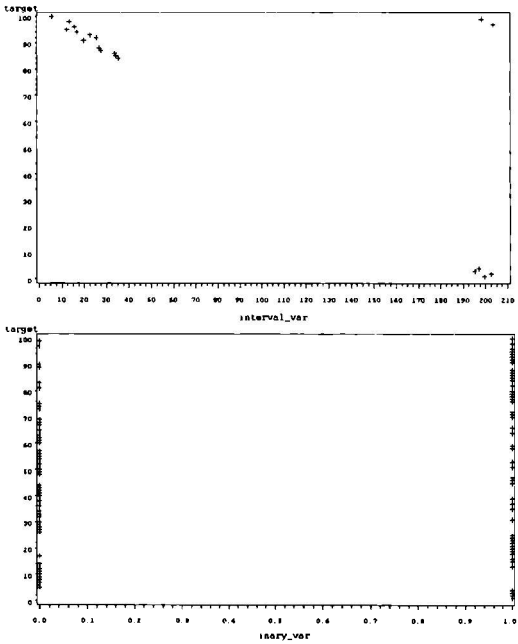


Figure 3. Different views of the dataset1

Investigating the figure it can be seen immediately: in order to the prediction is the most exact, the dataset has to be cut along the binary variable, because the fitting error of the regression models is the smallest in this manner.

Table 1 contains the results of the regression, the decision tree and the regression tree.

Table 1. Model errors on dataset1

Model	Average relative error (%)
Regression	120.92
Decision Tree	104.43
Regression Tree	5.18

As the results show, the behaviour of target variable in dataset1 cannot be modelled acceptable neither with regression nor with decision tree. However the regression tree gives an adequate model. Nevertheless it is not allowed to come to profound conclusions, because this dataset was generated to indicate that there are situations, where the regression tree algorithm performs much better than the traditional methods.

In the followings a real-life database is investigated. The data issue from a medical database. It is not too large, it contains 470 observations and the number of dimensions is 10. The target variable is a continuous quantity named PWV (pulse wave velocity) which means the velocity of the compression wave occurring by the heart-beat [11]. From this parameter can be reasoned the state of the blood-vessels. This value is estimated from other data, such as age, systole and diastole blood pressure, etc.

Table2. Model errors on dataset2

Model	Average relative error (%)
Regression	2.84
Decision Tree	4.59
Regression Tree	0.52

The target variable can be predicted very exactly from the predictor variables. All of the models give little error but it can be seen, that the predictive power of the regression tree is the best.

The third dataset issues from a moneyed corporation environment. It contains 20018 observations and 10 dimensions. The target variable is the income which can be estimated very poorly from the available predictor variables, but the best estimation is given by the regression tree.

Table 3. Model errors on dataset3

Model	Average relative error (%)
Regression	83.10
Decision Tree	53.08
Regression Tree	46.76

4.1. Fine tuning of the models

All of these models can be tuned by different parameters. Since the number of possible model variants is huge, finding the absolute best solution is a difficult task. More models were built to each dataset. The parameters of stepwise regression were constants in the case of every running (slstay = 0.08, slentry = 0.08, see Chapter 3). The SAS Decision Tree can be tuned by several parameters (e. g. splitting criterion: F test / variance reduction, minimum number of observations in a leaf, maximum depth of tree). The regression tree algorithm can be tuned by three parameters: maximum depth of tree, minimum number of observations in a leaf and the number of splits along a dimension (per levels).

The tables contain only the test results of the model variants, which gave the best average relative error on the test data. The documentation of fine tuning's process is not in the paper because of lack of space, but speaking in a general way it can be stated, that the regression tree performed better than conventional regression in any case, and also better than decision tree almost in all cases (the decision tree was able to be better in only such cases, where it was well-tuned and the regression tree was roughly-tuned).

5. Conclusions

Extending the idea presented in [4], a new algorithm has been developed, which combines the properties of regression and decision trees. The result is a decision tree whose leaves contain regression models. By finding the split points the fitting error of the regression model on the left and right side is used as the measure of the cut's goodness. In order to the induction of the tree finishes in an acceptable time, in the algorithm is not carried out all of the cuts. This compromise redound an good predicitve model.

According to the results of the tests the model is much better than the traditional regression (it results from the structure of the model), and it is better than the decision tree used in some commercial software products, as well.

5.1. The speed of the algorithm

The method accomplishes actually the series of splitting and stepwise regression steps. Running the algorithm on the third database (which is the largest) the running time was about 20 minutes on an average computer (PIII, 850 MHz, 384 MB RAM) in the case of a relative big tree (5 levels). Although there are no results referring to this, according to the experiences from the tests, the running time depends minimally on the number of the observations. The most important factor is the number of the splits. The relation between the number of the splits and the running time is linear.

5.2. Further Developments

The algorithm misses the pruning for the present. But with appropriate setting up of the number of the elements in a node the overfitting can be minimalized. Even so completing this algorithm with a pruning method some performance increase could be reached.

From the point of view of the running time it is an important question, how many splits are executed during the build-up of the tree. It would require further researches answering of the question: what kind of algorithm would be optimal to determinate the cut points.

Acknowledgements

The author would like to thank Professor *László Cser*, *Bulcsú Fajszi* and *Márton Zimmer* for the support and the valuable pieces of advice.

REFERENCES

[1] BREIMAN, L., FRIEDMAN, J., OLSHEN, R., STONE, J.: *Classification and regression tree*, Wadsworth & Brooks, 1984.

[2] QUINLAN, J. R.: *Learning with continuous classes*, in Proceedings AI'92, Adams & Sterling (Eds.), World Scientific, pp. 343-348, 1992.

[3] LUBINSKY, D.: *Tree Structured Interpretable Regression*, in Learning from Data, Fisher D. & Lenz H.J. (Eds.), Lecture Notes in Statistics, 112, Springer, pp. 387-398, 1994.

[4] KARALIC, A.: *Linear Regression in Regression Tree Leaves*. In Proceedings of ISSEK'92 (International School for Synthesis of Expert Knowledge) Workshop, Bled, Slovenia, 1992.

[5] MALERBA, D., APPICE, A., CECI, M., MONOPOLI, M.: *Trading-off local versus global effects of regression nodes in model trees.* In H.-S. Hacid, Z.W. Ras, D.A. Zighed & Y. Kodratoff (Eds.), Foundations of Intelligent Systems, 13th International

Symposium, ISMIS'2002, Lecture Notes in Artificial Intelligence, 2366, 393-402, Springer, Berlin, Germany, 2002.

[6] TORGO, L.: *Functional Models for Regression Tree Leaves.* Proc. 14th International Conference on Machine Learning, 1997.

[7] DOBRA, A., GEHRKE, J.: *SECRET - A Scalable Linear Regression Tree Algorithm.* In Proc. of ACM SIGKDD, pp. 481-487, 2002.

[8] HAN, J., KAMBER, M.: *Data Mining – Conceptions and Techniques.* Panem, Budapest, 2004. (in Hungarian)

[9] FAJSZI, B., CSER, L.: *Business Knowledge in the Data.* Budapest, 2004. (in Hungarian)

[10] SAS INSTITUTE INC.: http://www.sas.com/

[11] HAST, J.: *Self-mixing interferometry and its applications in noninvasive pulse detection.* http://herkules.oulu.fi/isbn951426973X/html/x957.html