Linear Regression in Regression Tree Leaves

Aram Karalič
Jožef Stefan Institute
Jamova 39
61111 Ljubljana
Slovenia
aram.karalic@ijs.si

June 1992

Abstract

The advantage of using linear regression in the leaves of a regression tree is analysed in the paper. It is carried out how this modification affects the construction, pruning and interpretation of a regression tree. The modification is tested on artificial and real-life domains where its impact on classification error and stability of the induced trees is considered. The results show that the modification is beneficial, as it leads to smaller classification errors of induced regression trees. The Bayesian approach to estimation of class distributions is used in all experiments.

1 Introduction

In the context of inductive learning, regression trees are used when one wants to learn a relation between (discrete or continuous) attributes and a *continuous* class. Regression trees are similar to binary classification trees. Their inner nodes are labelled with a test on the value of an attribute, and their leaves are labelled with a function, that prescribes a value to the class. A part of a regression tree is shown on Figure 1. The trees are constructed by algorithms from the TDIDT family of algorithms. Two well known representatives of this family are ID3 [13] and ASSISTANT [11, 4] algorithms. Once built, regression tree is interpreted similarly as a classification tree. When classifying an example by the regression tree, the interpretation begins at the root of the tree. According to the result of the test in the node, the example "travels" to the left or to the right subtree of the node until a leaf is reached. At that point, a class value is computed according to

the function labelling the leaf. This class value represents an answer of the tree and is assigned to the example. In the basic CART algorithm [1] the class value in the leaves of the induced regression tree is estimated as a constant function. This definition of the leaves is extended in the paper. The extension allows function in the leaf, which predicts value of the class, to be linear function of continuous attributes.

Regression trees are briefly described in Section 2. Section 3 contains the description of Bayesian approach to the tree-structured regression. The main topic of the article — use of linear regression in the leaves of the regression tree — is introduced in Section 4. The developed concepts were implemented in RETIS — a system for regression tree construction. RETIS was used to test the behaviour of local linear regression in six different domains. The experiments are described in Section 5 and their results are discussed in Section 6, where also conclusions are drawn.

2 Regression Trees

The most important distinction between regression and classification trees is the following: while classification trees are used to classify objects into *discrete* classes, regression trees are used when the class is *continuous*. A regression tree actually implements a function $y(x_1, x_2, \ldots, x_n)$ of n continuous or discrete attributes $x_1, x_2, \ldots, x_n$. More about regression trees can be found in [1].

The algorithm for regression tree construction belongs to TDIDT family of algorithms. These algorithms split example set, representing the node of a tree, into two subsets, from which they recursively form subtrees. The vital part of the algorithm is measure of goodness of split, which is derived from the measure of the impurity of an example set. Since the class is continuous, an estimate of variance of the class values is used as an impurity measure. The best split of examples in a node is taken to be the one that minimises the expected impurity I_{exp}, given by

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

where p_l, p_r denote probabilities of transitions into the left and the right son of the node, and I_l, I_r are the corresponding impurities. The variance of the class values of example set E, computed by the formula:

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

is used as an impurity measure in original CART algorithm. In the formula, w_i is the weight of the i-th example, $W(E)$ denotes the sum of example weights of the example set E and $\mu(E)$ denotes mean class value of example set E.

The quality of the constructed tree is measured by the *mean squared error* R of a tree T, defined with

$$R(T) = \frac{1}{N}\sum_{i=1}^{N}(y_i - y(x_{i1}, x_{i2}, \ldots, x_{in}))^2$$

where N is the number of examples used for testing the tree, y_i is the actual value of the class of the i-th example, $x_{i1}, x_{i2}, \ldots, x_{in}$ are values of its attributes, and $y(x_{i1}, x_{i2}, \ldots, x_{in})$ represents the value of the class estimated by a regression tree. The square root of the mean squared error ($\sqrt{R}$) facilitates better interpretation of the tree quality, since it uses the same unit of measure as does the predicted quantity, namely class. To compare the quality of several trees, possibly from different domains, the *relative mean squared error*, defined as

$$RE(T) = \frac{R(T)}{R(\mu)}$$

is used. Here, the mean squared error of the tree is normalised by the mean squared error of the predictor which always predicts the mean value of the training example set.

3 Bayesian Approach to Tree-Structured Regression

Bayesian approach to estimating class distribution in the tree-structured regression is based on Good's notion of Bayesian analysis [8] and was introduced in [10]. Here, let us briefly summarise main characteristics of Bayesian approach. Basic idea is that one initially assumes a prior value of estimated quantity. Some experiments are then performed. According to the results of experiments, prior value, or prior hypothesis, is modified as to obtain posterior value of the estimated quantity. This process can be incrementally repeated to improve the estimation. Bayesian approach to estimating probabilities was introduced in [2].

Bayesian estimate of class distribution of example set E is the combination of initial (prior) class distribution and class distribution of E. Weight, and thus influence, of the initial class distribution is regulated with parameter m. The prior distribution is obtained by taking the whole learning example set and assigning weight m/N to each example (N is the number of examples in the whole example set). The obtained example set is called E_m. The distribution of example set E if effectively obtained by taking distribution of $E \cup E_m$. The same principle is applied for computing various distribution momenta, as for example mean value and variance. For example, Bayesian estimate for variance of class values of given example set E is no longer given by (2) but is computed by the following formula:

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

4 Linear Regression in Regression Tree Leaves

In the basic CART algorithm, class value is estimated as a constant function in the leaves of the induced regression tree. The definition of a leaf of a regression tree will be extended. The extension will allow function in the leaf, which predicts the value of the class, to be linear function of the continuous attributes. This enables the leaves to be of the form shown in Figure 1. The use of linear regression in the leaves of a regression tree will be named *local linear regression.*

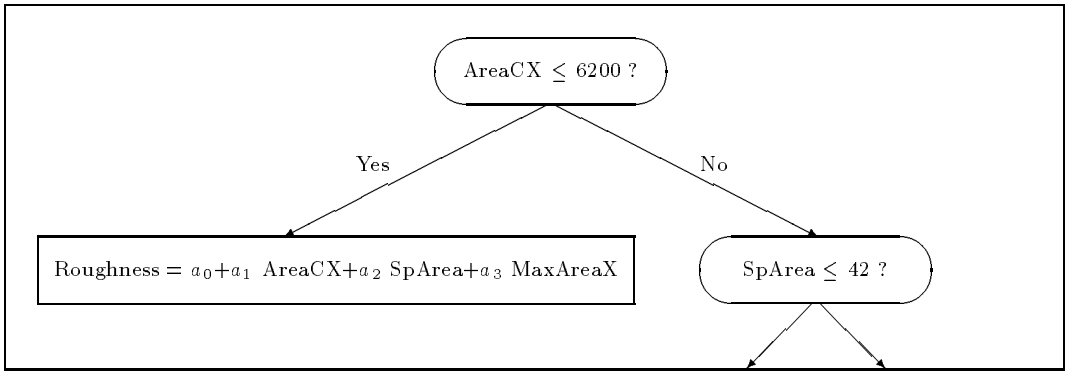

Figure 1: Part of a regression tree using local linear regression.

In this section the influence of local linear regression on the construction, pruning and interpretation of a regression tree is examined. The modification is tested in artificial and real-life domains and its impact on classification error and stability of the induced trees is analysed in following sections. The Bayesian approach to the estimation of class distributions is used.

4.1 Local Linear Regression During the Construction of a Regression Tree

In the process of tree construction, the basic CART algorithm chooses subtrees so as to minimise the expected variance, defined by formulae (1) and (2). However, when using local linear regression, variance is not an appropriate measure for impurity of an example set, as can be illustrated by the following example. Suppose we have an example set containing four examples, described with only one one attribute, and let attribute and class values be as depicted in Figure 2. Although in this case the variance is large, the

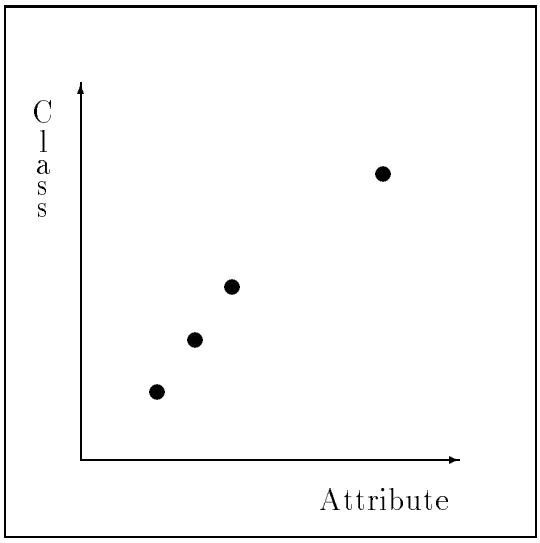

Figure 2: Example of set with large variance and low impurity.

observed set of examples is very pure: the error committed when using linear regression is almost zero. For that reason, the expression

$$I(E) = \frac{1}{W(E)} \sum_{e_i \in E} (y_i - g(\vec{x}_i))^2 \tag{3}$$

is used as an alternative impurity measure of an example set E. Function g represents the regression plane through the example set. Expected impurity of a split is then estimated as

$$I_{exp} = p_l I_l + p_r I_r$$

where p_l, p_r denote probabilities of transitions into the left and the right son of the node, and I_l, I_r are corresponding impurities (estimated by (3)). We have thus shown, that the main modification necessary during the construction of a regression tree with enabled local linear regression is use of impurity measure (3) instead of impurity measure (2) .

4.2 Local Linear Regression During Post-Pruning of a Regression Tree

For post-pruning of the generated regression trees an algorithm based on the Niblett-Bratko post-pruning method [12] was used. Pruning is based on the idea that for every node an estimation of its classification error on test examples is made. In each node the algorithm makes an estimate of static error (e_s) and backed-up error (e_b). *Static error* represents the expected error on unknown examples if this node was a leaf (that is: if the

tree was pruned at this node). *Backed-up error* is an estimation of expected error in the case the tree was not pruned at that node. If the static error is less than or equal to the backed-up error, the subtree is pruned and the node is converted to a leaf.

The effect of employing local linear regression is that the impurity (and thus the classification error) is not estimated with the expression (2), but with the expression (3). So, the static error is estimated by (3) and backed-up error as

$$e_b = p_l e_l + p_r e_r$$

where p_l, p_r are probabilities of going to the left or to the right subtree, and e_l, e_r represent error estimates for those subtrees. Level of pruning is regulated by the value of parameter m.

4.3 Local Linear Regression During Interpretation of a Regression Tree

When classifying a new example with a regression tree, the class assigned to the example is determined by the value of the linear function in the leaf to which the example came during the process of classification.

5 Experiments

The developed concepts were implemented in RETIS — a system for regression tree construction. RETIS was used to test the behaviour of local linear regression in six different domains. Experiment for each domain consisted of 10 repetitions of the sequence:

1. Split the set of all examples into learning example set and testing example set in proportion 70 : 30.
2. Construct a tree using learning examples and $m = 0$.
3. For each value of m (from a given set of values) do:
   - prune the tree using that value of m,
   - test (on testing examples) the pruned tree's classification error.
4. Select the best tree in the sequence of pruned trees.

We used $m = 0$ during the tree-construction phase for the similar reasons as described in [3]; namely: we consider the tree merely as a different form of learning examples and want to preserve all the information present in the learning example set. The values of parameter m, used in step 2, were: $0, {}^1/_{1024}, {}^1/_{512}, {}^1/_{256}, {}^1/_{128}, {}^1/_{64}, {}^1/_{48}, {}^1/_{32}, {}^1/_{24}, {}^1/_{16},$

$^1/_{14}$, $^1/_{12}$, $^1/_{10}$, $^1/_8$, $^1/_7$, $^1/_6$, $^1/_5$, $^1/_4$, $^1/_3$, $^1/_2$, 1, 2, 3, 4, 5, 6, 7, 8, 10, 12, 14, 16, 24, 32, 48, 64, 128, 256, 512 and 1024. The classification error of the best tree in the obtained sequence of pruned trees was analysed. It should be stressed, that at this point we do not discuss the issue of selecting the best tree from the sequence of pruned trees. We just assume that we know how to do it and observe how the local linear regression affects the results.

In our experiments six domains were used; three of them were artificially generated while three of them are real-life domains. Short description of the domains follows:

- **LINE**: An artificial domain, one discrete attribute x_1 (uniformly distributed values v_{11} and v_{12}), and two continuous attributes x_2 and x_3, with values uniformly distributed over interval $[0, 1)$. Class value is given by simple part-wise linear relation:
  if $x_1 = v_{11}$
  then $y := 1 + 2x_2 + x_3$
  else $y := -4 - 2x_2 - x_3$
  300 examples were available for experiments.

- **LEXP**: An artificial domain, one discrete attribute x_1 (uniformly distributed values v_{11} and v_{12}), and four continuous attributes x_2, x_3, x_4 and x_5, with values uniformly distributed over interval $[0, 1)$. Class value is given by relation:
  if $x_1 = v_{11}$
  then $y := 1 + 2x_2 + 3x_3 - e^{-2(x_4+x_5)}$
  else $y := 1 - 1.2x_2 - 3.1x_3 + e^{-3(x_4-x_5)}$
  300 examples were available for experiments.

- **LOSC**: An artificial domain, one discrete attribute x_1 (uniformly distributed values v_{11} and v_{12}), and four continuous attributes x_2, x_3, x_4 and x_5, with values uniformly distributed over interval $[0, 1)$. Class value is given by relation:
  if $x_1 = v_{11}$
  then $y := 1 + 1.5x_2 + x_3 + \sin(2(x_4 + x_5))e^{-2(x_2+x_4)}$
  else $y := -1 - 2x_2 - x_3 + \sin(3(x_4 + x_5))e^{-3(x_3-x_4))}$
  300 examples were available for experiments.

- **GRV3**: The domain of steel grinding. Roughness of the workpiece is to be determined from the properties of the sound produced during the process of steel grinding. 123 examples were available, described in terms of three attributes. Detailed description of the domain together with the previous work can be found in [7, 9].

- **SPIN**: This domain deals with the prediction of the cotton yarn strength from the properties of spinning material mixture. The mixture is described in terms of 10 attributes. There were only 18 learning examples available for the experiments. Details of the domain and the previous work can are described in [14].

- **ZRMK**: In this domain properties of fresh concrete are predicted from the data describing input materials and mix proportions. 254 examples described in terms of

ten continuous and four discrete attributes were available. The domain is described in more detail in [5].

The results of the experiments are summarised in six graphs, depicted in Figure 3, each containing the results obtained in one domain.

6 Discussion and Conclusions

We wanted to find out whether local linear regression was beneficial. The answer can be (partially) found by looking at graphs on Figure 3. The results of the t-test of significance for correlated samples [6] for the hypothesis: "The classification error when using local linear regression is smaller than the error without using local linear regression." are presented in the Table 1. It can be seen that in most cases local linear regression helped to achieve better results: the classification error was smaller when local linear regression was used. The best results were achieved in domains LINE, LEXP, GRV3. The improvement is also very significant in domains LOSC and ZRMK. However, in the SPIN domain, the use of local linear regression even quite significantly decreased the performance. This can

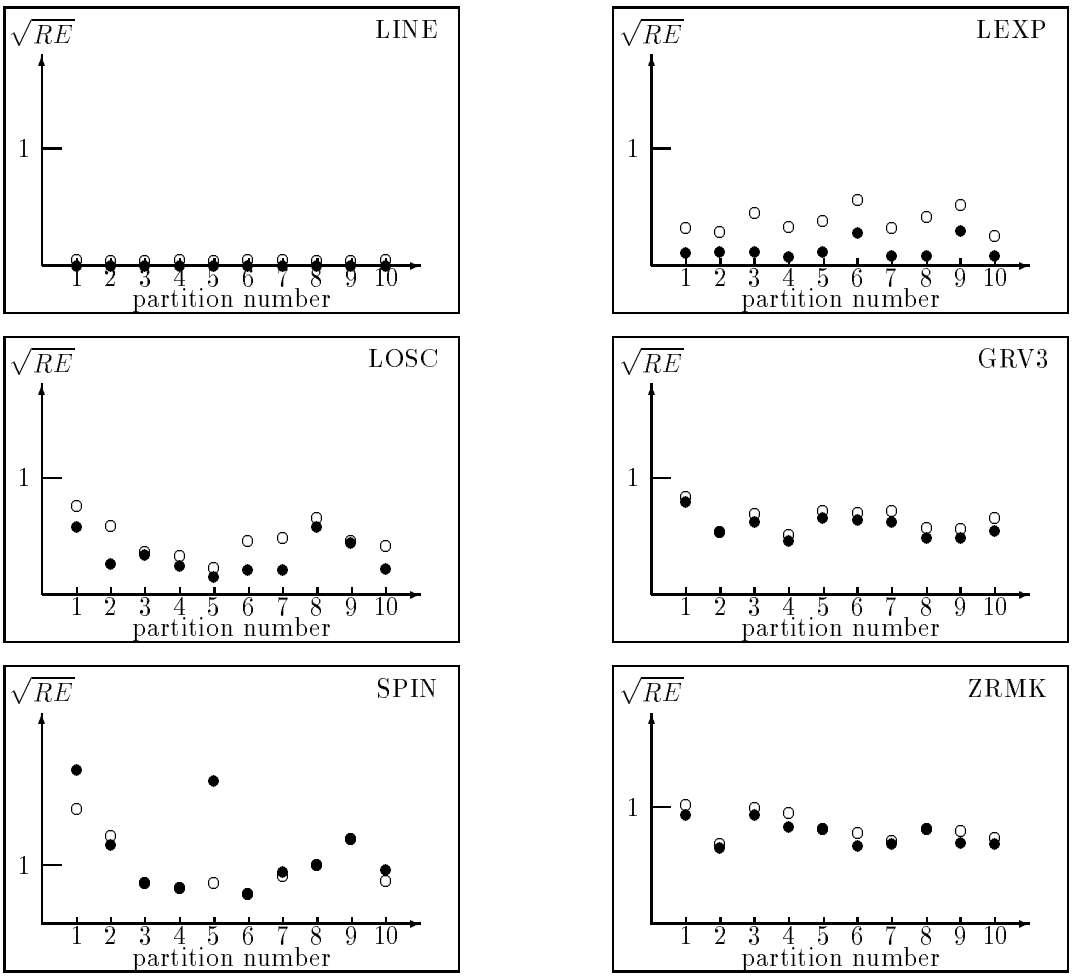

Figure 3: Comparison between regression trees created without the possibility of local linear regression (○) and regression trees built with local linear regression enabled (●). Relative error on test examples for all ten splits into learning end testing examples is shown.

be explained by the fact, that only 18 examples were available in this domain. Therefore, any generalisation must be made very carefully. Since we feel that generalisation by linear regression is "stronger" than the one with constant function, it is intuitively understandable that it can produce more unstable behaviour. Excellent results in domains LINE and LEXP can be explained by their inherent linearity or "quasi-linearity" of the parts of the domains. Results in GRV3 and ZRMK domains indicate some linear relationship between continuous attributes and classes in these domains. However, the question remains whether these regularities are really linear, or some other regression methods (fitting exponential curve, for example) could perform better. This is the impression obtained from looking at the final results significance tests. But if we examine each of the ten repetitions of the experiment separately, we will find cases where local linear regression decreased performance. Therefore, local linear regression should be used wherever there are indications of some linear relationships between attributes and class, but it should

not be used blindly, without careful examination of the background knowledge available for the domain.

Domain	hypothesis H_0	significance of H_0
LINE	$\sqrt{RE}_{m_+} < \sqrt{RE}_{m_-}$	>99.9
LEXP	$\sqrt{RE}_{m_+} < \sqrt{RE}_{m_-}$	>99.9
LOSC	$\sqrt{RE}_{m_+} < \sqrt{RE}_{m_-}$	99.9
GRV3	$\sqrt{RE}_{m_+} < \sqrt{RE}_{m_-}$	>99.9
SPIN	$\sqrt{RE}_{m_+} > \sqrt{RE}_{m_-}$	90.1
ZRMK	$\sqrt{RE}_{m_+} < \sqrt{RE}_{m_-}$	99.9

Table 1: Results of t-tests for hypotheses comparing error committed using local linear regression ($\sqrt{RE}_{m_+}$) and error committed without the use of local linear regression ($\sqrt{RE}_{m_-}$).

The overall conclusion is that local linear regression is beneficial in the sense of the quality of the induced regression tree. We think that it would be interesting to try also some other forms of regression. As a future work, we intend to try using exponential and quadratic regression and also to establish criteria for choosing the right form of regression.

References

[1] Breiman, L., Friedman, J.H., Olshen, R.A., & Stone, C.J.: *Classification and Regression Trees*, Wadsworth Int. Group, Belmont, California, USA, 1984.

[2] Cestnik, B.: Estimating probabilities: A Crucial Task in Machine Learning, *Proceedings of ECAI-90*, Stockholm, Sweden, 1990.

[3] Cestnik, B., & Bratko, I.: On Estimating Probabilities in Tree Pruning, *Proceedings of EWSL-91*, Porto, Portugal, 1991.

[4] Cestnik, B., Kononenko, I., & Bratko, I.: ASSISTANT 86: A Knowledge-Elicitation Tool for Sophisticated Users, *Progress in Machine Learning*, ed. by Bratko, I. and Lavrač, N., Sigma Press, Wilmslow, 1987.

[5] Cestnik, B., & Urbančič, T.: *Prediction of Fresh Concrete Properties with Artificial Intelligence Methods*, Technical Report, IJS DP-5963, Jožef Stefan Institute, Ljubljana, Slovenia, 1990 (in Slovene).

[6] Ferguson, G.A.: *Statistical Analysis in Psychology and Education*, McGraw-Hill, London, United Kingdom, 1966.

[7] Filipič, B., Junkar, M., Bratko, I., & Karalič, A.: An Application of Machine Learning to a Metal-Working Process, *Proceedings of ITI-91*, Cavtat, Croatia, 1991 (in press).

[8] Good, I.J.: *The Estimation of Probabilities*, M.I.T. Press, Cambridge, Massachusetts, USA, 1965.

[9] Junkar, M., Filipič, B., & Bratko, I.: Identifying the grinding process by means of inductive machine learning, *Preprints of the first CIRP Workshop of the Intelligent Manufacturing Systems*, Budapest, Hungary, 1991.

[10] Karalič, A., & Cestnik, B.: The Bayesian Approach to Tree-Structured Regression, *Proceedings of ITI-91*, Cavtat, Croatia, 1991 (in press).

[11] Kononenko, I.: *The Development of the Inductive Learning System Assistant*, M.Sc. Thesis, Edvard Kardelj University, Ljubljana, Slovenia, 1985 (in Slovene).

[12] Niblett, T. & Bratko, I.: Learning Decision Rules in Noisy Domains, *Development in Expert Systems*, (ed. Bramer, M.), Cambridge University Press, 1986.

[13] Quinlan, J.R.: Induction of Decision Trees, *Machine Learning*, Vol. 1, 1986.

[14] Stjepanovič, Z., & Jezernik, A.: A Contribution to the Prediction of Cotton Yarn Properties Using Artificial Intelligence, *Preprints of the first CIRP Workshop of the Intelligent Manufacturing Systems*, Budapest, Hungary, 1991.