CORE*- a system that predicts continuous variables

Marko Robnik-Šikonja
Faculty of Computer and Information Science
University of Ljubljana
Tržaška 25, 1001 Ljubljana, Slovenia
Marko.Robnik@fri.uni-lj.si

We describe CORE, an inductive system for learning regression trees. CORE incorporates some novel techniques for attribute estimation and constructive induction and can form various types of models in the leaves of the tree. Experimental results and its use in practice show that it is powerful tool for the analysis of the regression problems. It provides the user not only with the better predictive accuracy but also with the models that better capture the human understanding of the underlying phenomena.

1 Introduction

The majority of current propositional inductive learning systems predict discrete class. They can also solve the regression (also called continuous class) problems by discretizing the prediction (class) in advance. This approach is often inappropriate. Regression learning systems (also called function learning systems), e.g., CART (Breiman et al., 1984), Retis (Karalič, 1992), M5 (Quinlan, 1993), directly predict continuous value. We describe a regressional learning system called CORE which builds regression trees and incorporates novel techniques for attribute estimation, constructive induction and pruning and can form various types of models in the leaves of the tree.

The problem of estimating the quality of attributes seems to be an important issue in machine learning (e.g., feature selection, constructive induction). Heuristic measures for estimating the attribute's quality mostly assume the independence of attributes. They are therefore less appropriate in domains with strong dependencies between attributes. RReliefF (Robnik Šikonja and Kononenko, 1997) is aware of the contextual information and can correctly estimate the quality of attributes in problems with strong dependencies between attributes.

With constructive induction we change the description language of the learning algorithm in a way that makes certain dependencies and subconcepts easier to express. This can result in more compact and comprehensible hypotheses and better prediction accuracy.

When predicting the value of a new example with regression tree we follow decisions in the branches and predict the value with the model in the leaf. To induce the right model is therefore very important task for the learning algorithm.

In the next Section we provide some basics about regression and regression trees. In Section 3 we describe the most important ingredients of our learning system. We analyze the attribute estimation problem in regression, constructive induction, use of various models in the leaves of the tree and the pruning algorithms. In Section 4 we describe some experiments. The last Section summarizes and gives guidelines for further work.

2 The basics

The problem of regression consists of obtaining a functional model that relates the value of the target (dependent) continuous variable F with the values of predicting (independent) variables $A_1, A_2, ...A_v$. In the machine learning community the target variable is also called continuous class, while the predicting variables are mostly referred to as attributes.

CORE is an inductive learning system that builds models in form of a regression tree. As its input it uses the set of examples, each described with the vector of prediction and attribute values $(f_i, a_1, a_2, ..., a_v)$. Figure 1 illustrates the regression tree. We can see that it is similar to decision tree, except that instead of predicting discrete variable (class) it predicts continuous variable. Each interior node contains a decision. If the values of the examples agree with the decision we follow left branch otherwise we follow the right branch. When we reach the leaf of the tree we predict the value of the example with the model contained in that leaf.

Similar to other systems that learn regression trees e.g., CART (Breiman et al., 1984), Retis (Karalič, 1992) and M5 (Quinlan, 1993), CORE performs the induction by means of a recursive partitioning. In each node we estimate the attributes using the examples in that node and either select a single attribute for splitting the examples (as

*The CORE learning system was developed as a result of author's work on MSc. thesis and is freely available from author for all academic purposes (otherwise the author should be contacted).

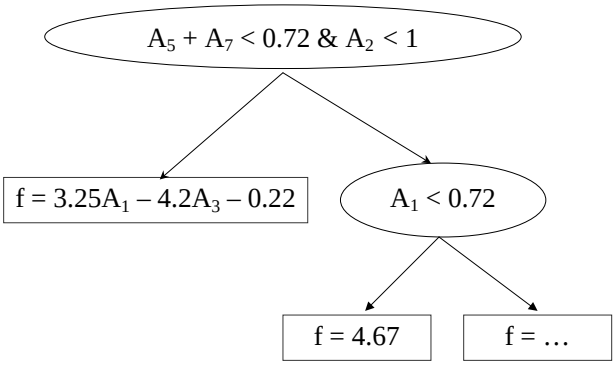

Figure 1: An illustration of the regression tree. Interior nodes contain decision and leaves contain prediction models.

CART, Retis and M5 do), or try to form a construct that would capture some subconcept. According to the values of the examples at selected attribute/construct we split examples into two groups and than recursively repeat the process on each group. In the terminal node (called leaf) we build the model to predict the outcome. We use three stopping criteria which can all be in effect at the same time: the minimal number of examples in the node, the minimal proportion of all examples in a node and proportion of the standard deviation in a root of the tree. While the rationale for the first two is to provide enough examples for the reliable model, the third criterion estimates how much variance is there left in the data after we apply the model.

After this initial building phase most of the learning systems use some form of pruning to reduce the tree size and to increase its generalization performance.

As we see from the description above there are four important tasks for the learning system: estimation of the quality of the attributes in each interior node, construction of the splitting decision, building the predictive models in the leaves of the tree and pruning the tree. We describe each of these crucial ingredients below.

3 Ingredients

3.1 Attribute estimation

The problem of estimating the quality of attributes seems to be an important issue in both classification and regression and in machine learning in general (e.g., feature selection, constructive induction). Heuristic measures for estimating the attribute's quality mostly assume the independence of attributes, e.g., information gain (Hunt et al., 1966), Gini index (Breiman et al., 1984), distance measure (Mantaras, 1989), and j-measure (Smyth and Goodman, 1990) for discrete class and the mean squared and the mean absolute error (Breiman et al., 1984) for regression. They are therefore less appropriate in domains with strong dependencies between attributes. Relief (Kira and Rendell, 1992) and its extension ReliefF (Kononenko, 1994) in classification are aware of the contextual information and can correctly estimate the quality of attributes in problems with strong dependencies between attributes.

RReliefF (Robnik Šikonja and Kononenko, 1997) is an adaptation of ReliefF for regression problems and preserves all its favorable features: it is robust to noise and additional random attributes and can handle missing values. RReliefF's estimate $W[A]$ of the quality of attribute A is an approximation of the following difference of probabilities (Kononenko, 1994):

$$W[A] =$$

$$P(\text{diff. value of A}|\text{nearest inst. with diff. predict.}) -$$

$$P(\text{diff. value of A}|\text{nearest inst. with similar predict.}) \quad (1)$$

If we intuitively think of the quality of attributes, we expect that good attributes would partition examples with quite different prediction and not separate between examples with similar prediction values. This is exactly what RReliefF does. The first term in Equation (1) rewards the attributes for partitioning between the different prediction values, and the second term punishes the attributes which separate between the similar values. We have to take this into account into the local context, of course. The Equation (1) provides us with a different perspective on attribute estimation and our use on some problems from medicine, ecology and economy has shown that humans quite like these estimates because they are in agreement with our intuitive understanding of the underlying phenomena.

3.2 Constructive induction

With constructive induction we change the description language of the learning algorithm in a way that makes certain dependencies and subconcepts easier to express. This can result in more compact and comprehensible hypothesis and better prediction accuracy. We do not know of any other regression system using constructive induction, so we have decided to use operator based construction in our regression tree context. We have implemented a set of operators (e.g. conjunction, addition, multiplication) and a user can choose which ones the learning algorithm will try to apply.

The search space of the constructive induction is extremely large (exponential in the number of attributes) and time consuming therefore it is essential to find useful heuristics to bound it. We have used RReliefF and combined it with the minimum description length (MDL)

principle (Li and Vitányi, 1993). We have derived the coding schemes for constructs and estimations of their quality (Robnik Šikonja, 1997). The experiments have shown that with the use of constructive induction we can gain a lot in terms of prediction accuracy, compactness and comprehensiveness of the tree, but we have to take good care not to overfit the data.

3.3 Models in the leaves

The first regression tree learning system CART (Breiman et al., 1984) has used the average prediction (class) value of the examples in each leaf node as the predictor. Retis (Karalič, 1992) and M5 (Quinlan, 1993) use linear models for prediction. Retis uses all the attributes to form the model, while M5 prunes the models with heuristics which tries to capture the trade off between the complexity of the model, number of available examples and the predictive accuracy. Our system also prunes the models, but we prefer the pruning based on the MDL principle.

We can have other types of models in the leaves. Nearest neighbour prediction and kernel regression models have been tested (Torgo, 1997) and encouraging results were obtained. We intend to employ these two type of models in further work.

3.4 Pruning

There are three well known techniques for pruning regression trees:

- CART (Breiman et al., 1984) uses cost complexity pruning which uses parameter α to control the trade off between the size and accuracy of the tree. To set the value of this parameter we have to provide a separate pruning set of examples or perform cross validation. For real world problems typically there is not enough data for a separate pruning set, while the cross-validation is computationally expensive.

- Retis (Karalič, 1991) uses the bottom-up algorithm for pruning which compares the error estimates of a node with the error estimate of its subtree. The errors are estimated using Bayesian approach to probability estimation (Cestnik, 1991).

- M5 (Quinlan, 1993) uses the same algorithm as Retis but uses different error estimation.

In CORE we have implemented the pruning techniques of Retis and M5. Additionally we have implemented a pruning based on the MDL principle. Besides the advantage of being theoretically sound it was also empirically shown (Robnik Šikonja, 1997) that using the new pruning we get approximately the same prediction error but statistically significantly smaller trees.

4 Results

We do not have enough space available to systematically test all the novel ingredients of our system and compare them with the ones previously used. Some of these experiments and analyses can be found in (Robnik Šikonja, 1997). Here we present just a single experiment which compares the performance of CORE with RReliefF and with mean squared error as measures of the attribute's quality. All other parameters of the learning system are the same (stopping criteria, linear models in the leaves, M5 pruning).

We ran our system on the artificial data sets and on domains with continuous prediction value from UCI (Murphy and Aha, 1995). Artificial data sets used were Fraction (floating point generalization of the parity concept), Modulo-8, (integer generalization of the parity), Parity (randomized continuous parity), Linear (simple linear dependency with 4 important attributes) and Cosinus (nonlinear dependency using cosinus on one attribute multiplied by the linear combination of two attributes). Altogether there were 11 artificial data sets, each consisting of 10 attributes - 2, 3 or 4 important, the rest are random, and containing 1000 examples. For each domain we collected the results as the average of 10 fold cross-validation. We present results in Table 1.

Table 1: Relative error and complexity of the regression and model trees with RReliefF and MSE as the estimators of the quality of the attributes.

	RReliefF		MSE		
domain	RE	C	RE	C	S
Fraction-2	.34	112	.86	268	+
Fraction-3	.73	285	1.08	440	+
Fraction-4	1.05	387	1.10	392	0
Modulo-8-2	.22	98	.77	329	+
Modulo-8-3	.58	345	1.08	436	+
Modulo-8-4	1.05	380	1.07	439	0
Parity-2	.28	125	.55	208	+
Parity-3	.31	94	.82	236	+
Parity-4	.35	138	.96	283	+
Linear	.02	4	.02	4	0
Cosinus	.27	334	.41	364	+
Auto-mpg	.13	97	.14	102	0
Auto-price	.14	48	.12	53	0
CPU	.12	33	.15	47	0
Housing	.17	129	.15	177	0
Servo	.25	53	.28	55	0

We compare the relative mean squared error (RE) of the predictors and the complexity (C) of the trees. C is the number of all the occurrences of all the attributes anywhere in the tree plus the constant term in the leaves. The column labeled S presents the significance of the differ-

ences in RE between RReliefF and MSE computed with paired t-test at 0.05 level of significance. 0 indicates that the differences are not significant at 0.05 level, '+' means that predictor with RReliefF is significantly better, and '-' implies the significantly better score by MSE.

On artificial data the predictor generated with RReliefF is mostly significantly better than the predictor generated with MSE. On UCI databases the predictors are comparable, RReliefF being better on 3 data sets, and MSE on 2, but with insignificant differences. The complexity of the models induced by RReliefF is considerably smaller in most cases on both artificial and UCI data sets which indicates that RReliefF was more successful detecting the dependencies. With RReliefF the strong dependencies were mostly detected and expressed with the selection of the appropriate attributes in the upper part of the tree; the remaining dependencies were incorporated in the linear models used in the leaves of the tree. MSE was blind for strong dependencies and was splitting the examples solely to minimize their impurity (mean squared error) which prevented it to successfully model weaker dependencies with linear formulas in the leaves of the tree.

5 Conclusions

Our experiments on artificial and real world problems as well as its use in practice showed that CORE is a powerful tool for analysis of the regression problems. It provides the user not only with the better predictive accuracy but also with the models that better capture the human understanding of the underlying phenomena.

As further work we are planning to extend CORE with new constructive operators and new types of models in the leaves. The use in practice and feedback from the users will guide us in our research and development.

References

Breiman, L., Friedman, L., Olshen, R., and Stone, C. (1984). *Classification and regression trees*. Wadsworth Inc., Belmont, California.

Cestnik, B. (1991). *Ocenjevanje verjetnosti v avtomatskem učenju*. PhD thesis, Univerza v Ljubljani, Fakulteta za elektrotehniko in računalništvo.

Hunt, E., Martin, J., and Stone, P. (1966). *Experiments in Induction*. Academic Press, New York.

Karalič, A. (1991). Avtomatsko učenje regresijskih dreves iz nepopolnih podatkov. Master's thesis, Univerza v Ljubljani, Fakulteta za elektrotehniko in računalništvo.

Karalič, A. (1992). Employing linear regression in regression tree leaves. In Neumann, B., editor, *Proceedings of ECAI'92*, pages 440–441. John Wiley & Sons.

Kira, K. and Rendell, L. A. (1992). A practical approach to feature selection. In D.Sleeman and P.Edwards, editors, *Proceedings of International Conference on Machine Learning*, pages 249–256. Morgan Kaufmann.

Kononenko, I. (1994). Estimating attributes: analysis and extensions of Relief. In De Raedt, L. and Bergadano, F., editors, *Machine Learning: ECML-94*, pages 171–182. Springer Verlag.

Li, M. and Vitányi, P. (1993). *An Introduction to Kolmogorov Complexity and Its Applications*. Springer-Verlag, New York.

Mantaras, R. (1989). ID3 revisited: A distance based criterion for attribute selection. In *Proceedings of Int. Symp. Methodologies for Intelligent Systems*, Charlotte, North Carolina, USA.

Murphy, P. and Aha, D. (1995). UCI repository of machine learning databases. (http://www.ics.uci.edu/ mlearn/MLRepository.html).

Quinlan, J. R. (1993). Combining instance-based and model-based learning. In *Proceedings of the X. International Conference on Machine Learning*, pages 236–243. Morgan Kaufmann.

Robnik Šikonja, M. and Kononenko, I. (1997). An adaptation of Relief for attribute estimation in regression. In Fisher, D., editor, *Machine Learning: Proceedings of the Fourteenth International Conference (ICML'97)*, pages 296–304. Morgan Kaufmann Publishers.

Robnik Šikonja, M. (1997). The developement of the heuristics for guiding the learning of the regression trees. University of Ljubljana, Faculty of Computer and Information Science, MSc thesis. (in Slovene).

Smyth, P. and Goodman, R. (1990). Rule induction using information theory. In Piatetsky-Shapiro, G. and Frawley, W., editors, *Knowledge Discovery in Databases*. MIT Press.

Torgo, L. (1997). Functional models for regression tree leaves. In Fisher, D., editor, *Machine Learning: Proceedings of the Fourteenth International Conference (ICML'97)*, pages 385–393. Morgan Kaufmann Publishers.