Fundamenta Informaticae XXI (2001) 1001–1013
IOS Press

Prediction of Ordinal Classes Using Regression Trees[*]

Stefan Kramer [C]
Institute for Computer Science
Albert-Ludwigs-University Freiburg
Georges-Köhler-Allee Geb. 79
D-79110 Freiburg i. Br., Germany
skramer@informatik.uni-freiburg.de

Gerhard Widmer
Department of Medical Cybernetics & AI
University of Vienna, and
Austrian Research Institute for Artificial Intelligence
Schotteng. 3, A-1010 Vienna, Austria
gerhard@ai.univie.ac.at

Bernhard Pfahringer
Department of Computer Science
University of Waikato
Hamilton, New Zealand
bernhard@cs.waikato.ac.nz

Michael De Groeve
Department of Computer Science
Katholieke Universiteit Leuven
Leuven, Belgium

Abstract. This paper is devoted to the problem of learning to predict ordinal (i.e., ordered discrete) classes using classification and regression trees. We start with S-CART, a tree induction algorithm, and study various ways of transforming it into a learner for ordinal classification tasks. These algorithm variants are compared on a number of benchmark data sets to verify the relative strengths and weaknesses of the strategies and to study the trade-off between optimal categorical classification accuracy (hit rate) and minimum distance-based error. Preliminary results indicate that this is a promising avenue towards algorithms that combine aspects of classification and regression.

Keywords: Machine Learning, Ordinal Classes, Regression Trees, Decision Trees, Classification, Regression, Inductive Logic Programming

[*]Various parts of this research were supported by the following projects: "Carcinogenicity Detection by Machine Learning" (Austrian Federal Ministry of Science and Transport); project no. 6930 (Jubiläumsfonds der Österreichischen Nationalbank); and P12645-INF (Fonds zur Förderung der Wissenschaftlichen Forschung (FWF)). Michael de Groeve was supported by a SOCRATES (ERASMUS) grant.
[C]Corresponding author

1. Introduction

Learning to predict discrete classes or numerical values from preclassified examples has long been, and continues to be, a central research topic in Machine Learning (e.g., [1, 16, 17]). A class of problems between classification and regression, learning to predict *ordinal classes*, i.e., discrete classes with a linear ordering, has not received much attention so far, which seems somewhat surprising, as there are many classification problems in the real world that fall into that category.

Given ordered classes, one is not only interested in maximizing the classification accuracy, but also in minimizing the distances between the actual and the predicted classes. However, the optimization of the classification accuracy does not imply the optimization of the numerical, distance-based error, and vice versa. Rather, there seems to be a trade-off between these two goals. One of the aims of this study is to investigate the trade-off between optimizing one or the other quantity.

In this paper, we study ways of learning to predict ordinal classes using regression trees. We will start with an algorithm for the induction of regression trees and turn it into an ordinal learner by some simple modifications. This seems a natural strategy because regression algorithms by definition have a notion of relative distance of target values, while classification algorithms usually do not. More precisely, we start with the algorithm S-CART (*Structural Classification and Regression Trees*) [8, 9, 10] and study several modifications of the basic algorithm that turn it into a distance-sensitive classification learner. Several variants of this algorithm are compared on a number of data sets to verify the relative strengths and weaknesses of the strategies and to study the trade-off between optimal categorical classification accuracy (hit rate) and minimum distance-based error.

This paper is organized as follows: The next section discusses related work. In section 3, we describe S-CART, the algorithm for first-order classification and regression trees that we build upon. In Section 4, we present the modifications that turn S-CART into an algorithm for predicting ordinal classes. In the subsequent section, we report the performance of several S-CART variants in five application domains. In the final section of the paper we discuss further work and come to our conclusions.

2. Related Work

Though the problem of learning to predict ordinal variables seems ubiquitous, especially in the social sciences, in information retrieval, and in other domains involving the prediction of human preferences, there has been rather little work in the machine learning and data mining area that specifically targets this problem. The field of statistics has developed several approaches to the problem of predicting ordinal variables, such as *Ordinal Logistic Regression* (e.g., [12, 13]). Some of these have also been studied in the field of neural networks (e.g,. [11]). Machine Learning, on the other hand, has started to look at the problem only recently.

[15] present a tree-based algorithm for the prediction of ordinal classes. They assume that the independent variables are ordered as well, which implies that the predictions made should be consistent with the order of the attribute values in the decision nodes. The authors present “repair strategies” for correcting inconsistent trees in case these consistency constraints are violated, as well as an algorithm for constructing consistent trees in the first place.

[6, 7] describe an algorithm based on the large margin idea known from data-dependent Structural Risk Minimization [19]. The algorithm is similar to Support Vector Machines [2]. They demonstrate

good results on artificial data and on a (very small) “real-world” information retrieval dataset. Unfortunately, the induced models are not readily interpretable, as they do not provide an intensional description of the learned concepts.

Other machine learning research that seems relevant to the problem of predicting ordinal classes is work on *cost-sensitive learning*. In the domain of propositional learning, some induction algorithms have been proposed that can take into account matrices of misclassification costs (e.g., [18, 22]). Such cost matrices might be used to express relative distances between classes.

Our goal was to provide a general algorithm that induces interpretable, symbolic models. Our algorithm makes no ordering assumptions regarding the independent variables, as in [15]. In relation to neural network and support vector machine approaches, its big advantage is the interpretability of the learned models (trees). Moreover, every regression tree algorithm (e.g., M5’ [23]) can easily be turned into an ordinal learner by our method. And finally, it should be noted that the tree learning algorithm S-CART, which forms the basis of our ordinal learners, can be applied to both *propositional* and *relational* domains – it is a full-fledged *Inductive Logic Programming (ILP)* algorithm. Indeed, both the Biodegradability and the Mesh datasets used in our experiments are of a relational nature. We have thus also provided a natural solution to ordinal prediction learning in ILP.

3. The Basic Learning Algorithm: S-CART (Structural Classification and Regression Trees)

Structural Classification and Regression Trees (S-CART) [8, 9, 10] is an algorithm that learns a first-order theory for the prediction of either discrete classes or numerical values from examples and relational background knowledge. The algorithm constructs a tree containing a positive literal or a conjunction of literals in each node, and assigns a discrete class or a numeric value to each leaf. S-CART is a full-fledged relational version of CART [1]. After the tree growing phase, the tree is pruned using so-called error-complexity pruning for regression or cost-complexity pruning for classification [1]. These types of pruning are based on a separate “prune set” of examples or on cross-validation.

For the construction of a tree, S-CART follows the general procedure of top-down decision tree induction algorithms [17]. It recursively builds a binary tree, selecting a positive literal or a conjunction of literals (as defined by user-defined schemata [20]) in each node of the tree until a stopping criterion is fulfilled. The algorithm keeps track of the examples in each node and the positive literals or conjunctions of literals in each path leading to the respective nodes. This information can be turned into a clausal theory (i.e., a set of first-order classification or regression rules).

As a regression algorithm, S-CART is designed to predict a numeric (real) value in each node and, in particular, in each leaf. In the original version of the algorithm the target value predicted in a node (let us call this the *center value* from now on) is simply the mean of the numeric class values of the instances covered by the node. A natural choice for the *evaluation measure* for rating candidate splits during tree construction is then the *Mean Squared Error (MSE)* of the example values relative to the means in the two new nodes created by the split:

$$MSE = \frac{1}{n_1 + n_2} \sum_{i=1}^{2} \sum_{j=1}^{n_i} (y_{ij} - \bar{y_i})^2$$

where n_i is the number of instances covered by branch i, y_{ij} is the value of the dependent variable of training instance e_j in branch i, and $\bar{y_i}$ is the mean of the target values of all training instances in branch i.

In constructing a single tree, the simplest possible stopping criterion is used to decide whether the tree should be further refined: S-CART stops extending the tree given some node when no literal(s) can be found that produce(s) two partitions of the training instances in the node with a required minimum cardinality. The post-pruning strategy then takes care of reducing the tree to an appropriate size.

S-CART has been shown to be competitive with other regression algorithms. Its main advantages are that it offers the full power and flexibility of first-order (Horn clause) logic, provides a rich vocabulary for the user to explicitly represent a suitable language bias (e.g. through the provision of schemata), and produces trees that are interpretable as well as good predictors.

As our goal is to predict discrete ordered classes, S-CART cannot be used directly for this task. We will, however, include results with standard S-CART in the experimental section to find out how paying attention to ordinal classes influences the mean squared error achievable by a learner.

4. Inducing Trees for the Prediction of Ordinal Classes

In the following, we describe a few simple modifications that turn S-CART into a learning algorithm for ordinal classification problems. In section 4.2, we consider some pre-processing methods that also might improve the results.

4.1. Adapting S-CART to Ordinal Class Prediction

The most straightforward way of adapting a regression algorithm like S-CART to classification tasks is to simply run the algorithm on the given data as if the ordinal classes (represented by integers) were real values, and then to apply some sort of *post-processing* to the resulting rules or regression tree that translates real-valued predictions into discrete class labels.

An obvious post-processing method is *rounding*. S-CART is run on the training data, producing a regular regression tree. The real values predicted in the leaves of the tree are then simply rounded to the nearest of the ordinal classes (not to the nearest integer, as the classes may be discontiguous; after pre-processing, they might indeed be non-integers — see section 4.2 below).

More complex methods for mapping predicted real values to symbolic (ordinal) class labels are conceivable. In fact, we did perform experiments with an algorithm that greedily searches for a mapping, within a defined class of functions, that minimizes the mean squared error of the resulting (mapped) predictions on the training set. Initial experiments were rather inconclusive; in fact, there were indications of the algorithm overfitting the training data. However, more sophisticated methods might turn out to be useful. This is one of the goals of our future research.

An alternative to post-processing is to modify the way S-CART computes the target values in the nodes of the tree *during tree construction*. We can force S-CART to always predict integer values (or more generally: a valid class from the given set of ordinal classes) in any node of the tree. The leaf values will thus automatically be valid classes, and no post-processing is necessary.

It is a simple matter to modify S-CART so that instead of the *mean* of the class values of instances covered by a node (which will in general not be a valid class value), it chooses one of the class values represented in the examples covered by the node as the center value that is predicted by the node, and

Table 1. Variants of S-CART for learning ordinal classes.

Name	Formula
POSTPROC. ROUND	$\hat{c}_i$ = mean of the $c_{ij} \in C_i$; real values in leaves of learned tree are rounded to nearest class in C_i
MEDIAN	$\hat{c}_i$ = median of class labels in multiset C_i
ROUNDEDMEANTOCLASS	$\overline{c}$ = mean of the $c_{ij} \in C_i$, $\hat{c}_i = \overline{c}$ rounded to nearest class $c_{ij} \in C_i$
MODE	$\hat{c}_i$ = most frequent class in C_i

relative to which the node evaluation measure (e.g., the mean squared error, see Section 3 above) is computed. Note that in this way, we modify S-CART's *evaluation heuristic* and thus its *bias*.

There are many possible ways of choosing a center value; we have implemented three: the *median*, the *rounded mean*, and the *mode*, i.e., the most frequent class. Let E_i be the set of training examples covered by node N_i during tree construction and C_i the multiset of the class labels of the examples in E_i, with $|E_i| = |C_i| = n$. In the MEDIAN strategy, S-CART selects the class $\hat{c}_i$ as center value that is the median of the class labels in C_i; in other words, if we assume that the example set E_i is sorted with respect to the class values of the examples, MEDIAN chooses the class of the $(n/2)^{th}$ example.[1] In contrast, the ROUNDEDMEANTOCLASS strategy chooses the class closest to the (real-valued) mean $\overline{c}$ of the class values in C_i. Finally, in the MODE strategy the center value $\hat{c}_i$ for node N_i is chosen to be the class with the highest frequency in C_i.

Table 1 summarizes the variants of S-CART that will be put to the test in Section 5 below.

4.2. Pre-processing

The results of regression algorithms can often be improved by applying various transformations to the raw input data before learning. The basic idea underlying different data transformations is that numbers may represent fundamentally different types of measurements. [14] distinguish, among others, the broad classes of *amounts and counts* (which cannot be negative), *ranks* (e.g., 1 = smallest, 2 = next-to-smallest, ...), and *grades* (ordered labels, as in A, B, C, D, E). They suggest the following types of pre-processing transformations: for amounts and counts, translate value v to $tv = \log(v + c)$; for ranks, $tv = \log((v - 1/3)/(N - v + 2/3))$, where N is the maximum rank; and for grades, $tv = (\phi(P) - \phi(p))/(P - p)$, where P is the fraction of observed values that are at least as big as v, p is the fraction of values $> v$, and $\phi(x) = x \log x + (1 - x) \log(1 - x)$. We have tentatively implemented these three pre-processing methods in our experimental system and applied the appropriate transformation to the respective learning problem in our experiments. Table 2 summarizes them in succinct form, in the notational frame of our learning problem.

[1]In this case the *Mean Absolute Deviation (MAD)* is used as distance metric instead of the Mean Squared Error, because the former measure is the one that is known to be minimized by the median.

Table 2. Pre-processing types (c = original class value; tc = transformed class value)

Name	Formula
RAW	No pre-processing ($tc = c$)
COUNTS	$tc = \log(c + 1 - \min(Classes))$
RANKS	$tc = \log((c - 1/3)/(N - c + 2/3))$, where $N = \max(Classes)$
GRADES	$tc = (\phi(P) - \phi(p))/(P - p)$, where $\phi(x) = x \log x + (1 - x)\log(1 - x)$, P = fraction of observed class values $\geq c$, p = fraction of observed class values $> c$

Note that these transformations do not by themselves contribute to the goal of learning rules for ordinal classes. Rather, they should be viewed as possible enhancements to the methods described above. In fact, pre-processing usually transforms the original ordinal classes into real numbers. That is no problem as the number of distinct values remains unchanged. Thus, the transformed values can still be treated as discrete class values without changing the learning algorithms.

In principle, one could combine any pre-processing technique with any method for predicting ordinal classes described above. For practical reasons, however, we only tested one type of pre-processing together with one of the methods in the experiments. Firstly, we applied only the one type of pre-processing that we considered suitable for the dependent variable of the given learning problem. As it turned out, the dependent variable was a "grade" in four of the five application domains, and a "count" in the remaining domain. So, due to the nature of the data, we actually used only two of the transformations in the experiments. Secondly, we actually applied this type of preprocessing together with POSTPROC. ROUND. So, in effect, we rounded to the next class in the "transformed space" and mapped this prediction back.

5. Experiments

5.1. Algorithms compared

In the following, we experimentally compare the S-CART variants and preprocessing methods on several benchmark datasets. Three quantities will be measured:

1. *Classification Accuracy* as the percentage of exact class hits,
2. the *Root Mean Squared Error (RMSE)* $\sqrt{1/n \sum_{i=1}^{n}(c_i - \hat{c})^2}$ of the predictions on the test set, as a measure of the average distance of the algorithms' predictions from the true class, and
3. the *Spearman rank correlation coefficient* with a correction for ties, which is a measure for the concordance of actual and predicted ranks.

As ordinal class prediction is somewhere "between" classification and regression, we additionally include two "extreme" algorithms in the experimental comparison. One, S-CART_CLASS, is a variant of S-CART designed for categorical classification. S-CART_CLASS chooses the most frequent class in a node as center value and uses the *Gini index of diversity* [1] as evaluation measure; it does not pay attention to the distance between classes. The other extreme, called S-CART_REGRESS, is simply the original S-CART as a regression algorithm that acts as if the task were to predict real values; we are interested in finding out how much paying attention to the discreteness of the classes costs in terms of achievable RMSE. (Of course, the percentage of exact class hits achieved by S-CART_REGRESS cannot be expected to be high.) Finally, we will also list the *Default* or *Baseline Accuracy* for each algorithm on each data set and the corresponding *Baseline RMSE*.

5.2. Data sets

The algorithms were compared on five datasets that are characterized by a clear linear ordering among the classes. Three of the data sets were taken from the UCI repository: Balance, Cars and Nursery.

The fourth dataset, the Biodegradability dataset [5], describes 328 chemical substances in the familiar "atoms and bonds" representation [21]. The task is to predict the half-rate of surface water aerobic aqueous biodegradation in hours. For previous experiments, we had already discretized this quantity and mapped it to the four classes *fast*, *moderate*, *slow*, and *resistant*, represented as 1, 2, 3, and 4.

The fifth dataset is the Mesh dataset [4], one of the by now classical benchmark problems in the area of ILP. The problem consists in predicting the optimal granularity of a finite element (FE) model of a given physical structure. More precisely, the task is to predict the appropriate number of FEs along a given edge. In the widely used data set first described in [3], there are 13 classes (1, . . . , 12 and 17 FEs). It seems obvious that in this domain, classification error should be regarded as a gradual phenomenon. Prescribing 1 FE for an edge that should have 12 is a worse mistake than predicting class 4 when the correct class is 5.

5.3. Results

In Tables 3 to 7, we summarize the results (classification accuracy, RMSE and Spearman rank correlation coefficient) on these datasets. For Balace, Car and Biodegradability, these are the results of 10-fold stratified cross-validation. For the Nursery dataset, we used 2/3 of the data for training and 1/3 for testing due to the sufficient size of the dataset. In the Mesh domain, we performed 5-fold cross-validation, where each of the 5 Mesh structures was held-out in turn. The reason for this is that we wanted to avoid dependencies between the training and the test sets. In the experiments, we used default settings of S-CART and did not make any attempts to optimize the learning parameters.

Figures 1 and 2 illustrate the results in the tables graphically. These plots depict the trade-off between the predictive (classification) error and the RMSE. Note that in the figures we present the misclassification rate rather than the accuracy (as in the tables).

The first and most fundamental obervation we make is that the learners improve upon the baseline values in almost all cases, both in terms of RMSE and in terms of classification accuracy. In other words, they really learn something.

As expected, there seems to be a fundamental tradeoff between the two goals of error minimization and accuracy maximization. This tradeoff shows most clearly in the results of the "extreme" algorithms

Table 3. Results from 10-fold cross-validation for Balance (625 examples, 4 attributes)

Approach	Accuracy	RMSE	Spearman
BASELINE	46.1%	1.39	-
S-CART_CLASS	79.8%	0.75	0.707
S-CART_REGRESS	4.3%	0.68	0.697
PREPROC. GRADES	77.8%	0.69	0.736
POSTPROC. ROUND	76.0%	0.71	0.723
MEDIAN	77.9%	0.75	0.707
ROUNDEDMEANTOCLASS	72.8%	0.68	0.731
MODE	79.8%	0.73	0.725

Table 4. Results from 10-fold cross-validation for Cars (1278 examples, 6 attributes)

Approach	Accuracy	RMSE	Spearman
BASELINE	70.0%	0.84	-
S-CART_CLASS	95.4%	0.27	0.943
S-CART_REGRESS	78.9%	0.23	0.945
PREPROC. GRADES	95.2%	0.25	0.952
POSTPROC. ROUND	94.7%	0.26	0.939
MEDIAN	92.0%	0.32	0.875
ROUNDEDMEANTOCLASS	92.1%	0.30	0.892
MODE	88.7%	0.41	0.810

S-CART_REGRESS and S-CART_CLASS: S-CART_CLASS, which solely seeks to optimize the hit rate during tree construction but has no notion of class distance, is among the best class predictors in all five domains, but among the worst in terms of RMSE. S-CART_REGRESS, on the other hand, is rather successful as a minimizer of the RMSE, but unusable as a classifier.

Interestingly, neither of the two solves its particular problem optimally: some ordinal learners beat S-CART_CLASS in terms of accuracy, and some beat the regression "specialist" S-CART_REGRESS in terms of the RMSE.

For Balance and for Cars, both the pre-processing and the simple post-processing method are able to achieve good predictive accuracy while at the same time keeping an eye on the class-distance-weighted error. The pre-processing method seems to perform particularly well in this trade-off. Both the pre-processing and the post-processing methods also perform favorably in terms of the Spearman rank correlation coefficient. Note that in the Balance domain, predicting the mode improves over pure

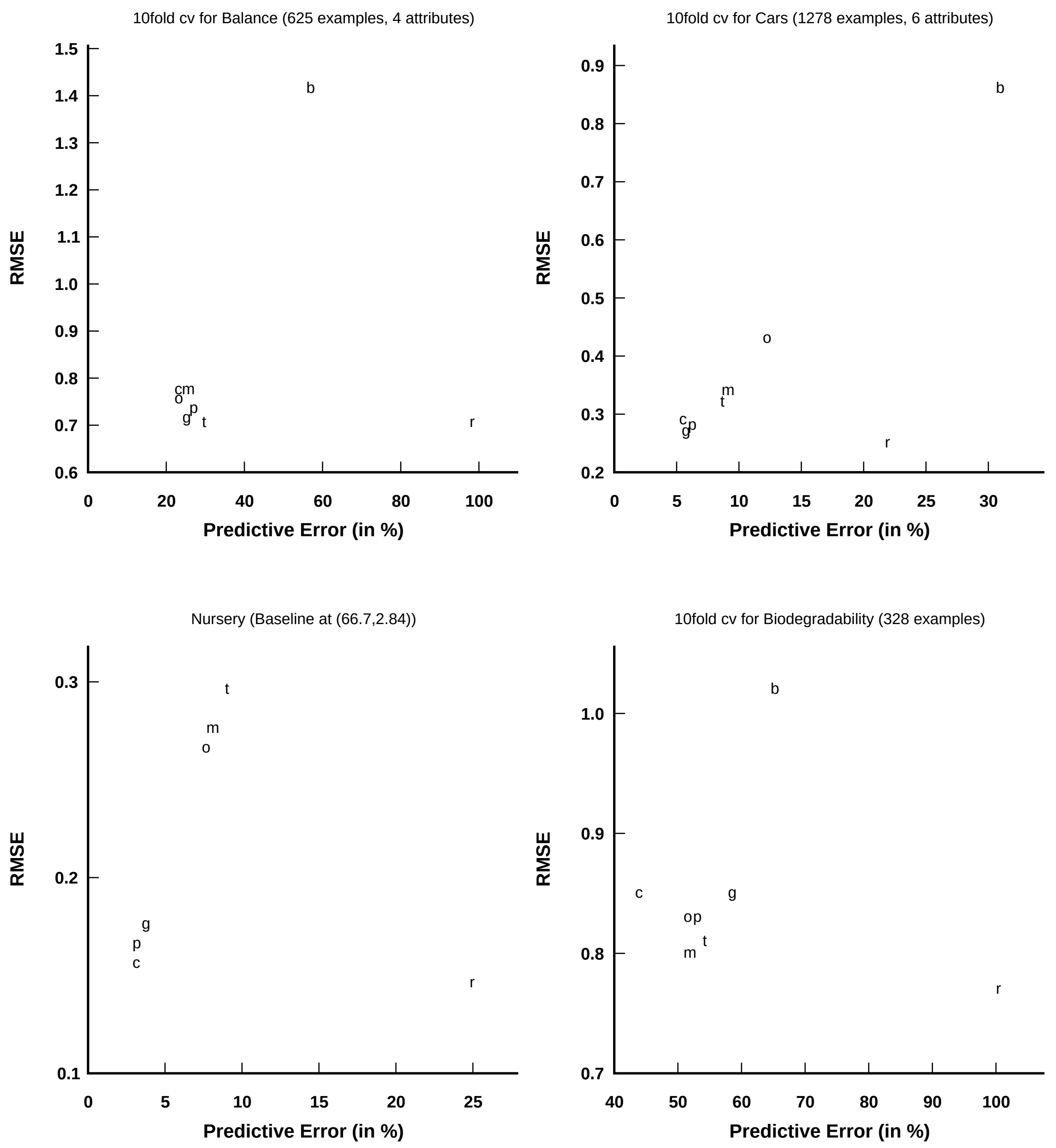

Figure 1. Illustration of the trade-off between predictive (classification) error and RMSE; b ... BASELINE, c ... S-CART_CLASS, r ... S-CART_REGRESS, g ... PREPROC. GRADES, p ... POST-PROC. ROUND, m ... MEDIAN, t ... ROUNDEDMEANTOCLASS, o ... MODE

Table 5. Results for Nursery (12961 examples, 8 attributes); 2/3 of the examples were used for training, 1/3 for testing

Approach	Accuracy	RMSE	Spearman
BASELINE	33.3%	2.84	-
S-CART_CLASS	97.6%	0.15	0.985
S-CART_REGRESS	75.7%	0.14	0.975
PREPROC. GRADES	97.0%	0.17	0.985
POSTPROC. ROUND	97.6%	0.16	0.985
MEDIAN	92.8%	0.27	0.962
ROUNDEDMEANTOCLASS	91.6%	0.29	0.955
MODE	93.1%	0.26	0.965

Table 6. Results from 10-fold cross-validation for Biodegradability (328 examples)

Approach	Accuracy	RMSE	Spearman
BASELINE	36.6%	1.01	-
S-CART_CLASS	57.9%	0.84	0.561
S-CART_REGRESS	1.2%	0.76	0.537
PREPROC. GRADES	43.3%	0.84	0.436
POSTPROC. ROUND	48.8%	0.82	0.489
MEDIAN	50.3%	0.79	0.538
ROUNDEDMEANTOCLASS	47.3%	0.80	0.510
MODE	50.3%	0.82	0.506

classification in terms of the RMSE and the Spearman rank correlation coefficient.

In the Nursery domain, all the variants fail to show a favorable performance compared to the classification and regression "specialists". This is the only domain in our experiments, where none of the variants are able to achieve an interesting result in terms of the trade-off between classification accuracy and distance-based error.

For Balance, Cars and Nursery, methods modifying the center value during tree construction (MEDIAN, ROUNDEDMEANTOCLASS and MODE) do not seem to perform well. An exception is the performance of MODE in the Balance domain, which represents an improvement over S-CART_CLASS.

Results for Biodegradability are different from the other results. The biodegradability domain is different from other domains in this study in several respects: It has fewer examples, it is known to have class noise and it is essentially relational. Here, methods modifying the center value during tree

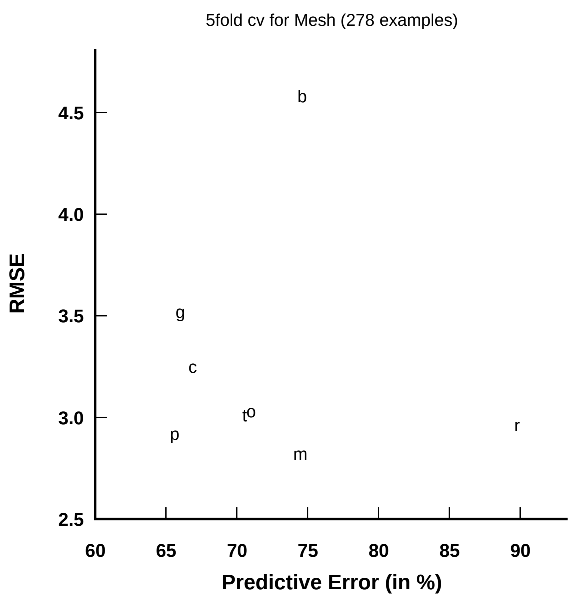

Figure 2. Illustration of the trade-off between predictive (classification) error and RMSE (ctd.); b ... BASELINE, c ... S-CART_CLASS, r ... S-CART_REGRESS, g ... PREPROC. COUNTS, p ... POSTPROC. ROUND, m ... MEDIAN, t ... ROUNDEDMEANTOCLASS, o ... MODE

Table 7. Results from 5-fold cross-validation for Mesh (278 examples)

Approach	Accuracy	RMSE	Spearman
BASELINE	26.3%	4.51	-
S-CART_CLASS	34.0%	3.18	0.683
S-CART_REGRESS	11.0%	2.89	0.731
PREPROC. COUNTS	34.9%	3.45	0.675
POSTPROC. ROUND	35.3%	2.85	0.795
MEDIAN	26.6%	2.75	0.780
ROUNDEDMEANTOCLASS	30.2%	2.94	0.779
MODE	29.9%	2.96	0.687

construction perform better, but not good enough to be competitive with either the classification or the regression method. Still, it should be noted that the RMSE of these methods is between the RMSE of the classification "specialist" and the one of the regression "specialist".

Finally, the results in the Mesh domain suggest it is possible for ordinal learners to beat both the classification and the regression "specialists" at the same time. The clear "winner" in this domain is POSTPROC. ROUND, which achieves a higher classification accuracy than S-CART_CLASS and at the same time a lower RMSE than S-CART_REGRESS. This result is most motivating; it shows that it is possible to develop learners that achieve good predictive accuracy while at the same time keeping the numeric (class-distance-weighted) error low.

Summing up, the results of the experiments are quite encouraging: In three of the five domains, S-CART variants predicting ordinal classes positioned themselves favorably in the trade-off between optimizing classification accuracy and optimizing distance-based error. In one domain (Nursery), all variants apparently failed to do so, and in one domain (Mesh), one of the ordinal learning algorithms was able to "beat" both the classification and the regression variants of S-CART. Also, it seems like pre-processing or post-processing methods are to be preferred over methods enforcing "legal" values during tree construction.

Drawing more general conclusions from these limited experimental data seems unwarranted. Our results so far show that tree learning algorithms for predicting ordinal classes can be naturally derived from regression tree algorithms, but more extensive experiments with larger data sets from diverse areas will be needed to establish the precise capabilities and relative advantages of these algorithms.

6. Further Work and Conclusion

Further work will be to perform experiments including the other, third transformation (for ranks) suggested by Mosteller and Tukey. Another direction of further work could be to combine the pre-processing methods with the other methods presented in this paper: In fact, almost all combinations make sense and could be tested.

It also would be interesting to build tree induction algorithms that do not enforce the prediction of "legal" classes during tree construction, but deal with this problem in the pruning phase. However, it is not yet clear which measure should be optimized in the pruning phase. An initial attempt optimizing the Spearman rank correlation coefficient in the pruning phase failed to produce interesting results in terms of the trade-off addressed in this paper.

In summary, we have taken first steps towards effective methods for learning to predict ordinal classes using regression trees. We have shown how algorithms for learning ordered discrete classes can be derived by simple modifications to a basic regression tree algorithm. Experiments in five benchmark domains have shown that, in some cases, the resulting algorithms are able to achieve good predictive accuracy while at the same time keeping the class-distance-weighted error low.

References

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

[2] Cortes, C., Vapnik, V.: Support Vector Networks. *Machine Learning*, **20**, 1995, 273–297

[3] Dolšak, B., Muggleton, S.: . The Application of Inductive Logic Programming to Finite-Element Mesh Design. In S. Muggleton (ed.), *Inductive Logic Programming*. Academic Press, 1992.

[4] Dolšak, B., Bratko, I., Jezernik, A.: Application of Machine Learning in Finite Element Computation. In R.S. Michalski, I. Bratko & M. Kubat (eds.), *Machine Learning and Data Mining: Methods and Applications*. Wiley, Chichester, UK, 1998.

[5] Dzeroski, S., Blockeel, H., Kompare, B., Kramer, S., Pfahringer, B., Van Laer, W.: Experiments in Predicting Biodegradability, in: *ILP-99: Proceedings Ninth International Workshop on Inductive Logic Programming*, Springer, Berlin, 1999.

[6] Herbrich, R., Graepel, T., Obermayer, K.: Support Vector Learning for Ordinal Regression. In *Proceedings of the Ninth International Conference on Artificial Neural Networks*, 1999, 97–102.

[7] Herbrich, R., Graepel, T., Obermayer, K.: *Regression Models for Ordinal Data: A Machine Learning Approach*. Report TR 99-3, Dept. of Computer Science, Technical University of Berlin, 1999.

[8] Kramer, S.: Structural Regression Trees. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence (AAAI-96)*. AAAI Press/MIT Press, Cambridge, MA, 1996.

[9] Kramer, S.: Relational Learning vs. Propositionalization: Investigations in Inductive Logic Programming and Propositional Machine Learning. *AI Communications*, **13**(4), 2000, 275–276.

[10] Kramer S., Widmer G.: Inducing Classification and Regression Trees in First-Order Logic, in: S. Dzeroski, N. Lavrac (eds.): *Relational Data Mining*, Springer, Berlin, 2001.

[11] Mathieson, M.: Ordered Classes and Incomplete Examples in Classification. In M. Mozer et al. (eds.), *Advances in Neural Information Processing Systems 9*. MIT Press, Cambridge, MA, 1996.

[12] McCullagh, P.: Regression Models for Ordinal Data. *Journal of the Royal Statistical Society Series B*, **42**, 1980, 109–142.

[13] McCullagh, P., Nelder, J.A.: *Generalized Linear Models*. Chapman & Hall, London, 1983.

[14] Mosteller, F., Tukey, J.W.: *Data Analysis and Regression - A Second Course in Statistics*. Addison-Wesley, Reading, MA, 1977.

[15] Potharst, R., Bioch, J.C.: Decision Trees for Ordinal Classification. *Intelligent Data Analysis*, **4**(2), 2000.

[16] Quinlan, J.R. (1992). Learning with Continuous Classes. In *Proceedings AI'92*. World Scientific, Singapore, 1992.

[17] Quinlan, J.R. (1993). *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.

[18] Schiffers, J. (1997). A Classification Approach Incorporating Misclassification Costs. *Intelligent Data Analysis*, **1**(1).

[19] Shawe-Taylor, J., Bartlett, P., Williamson, R.C., Anthony, M.: *Structural Risk Minimization over Data-dependent Hierarchies*. Technical Report NC-TR-1996-053, Royal Holloway, University of London, 1996.

[20] Silverstein, G., Pazzani, M.J.: Relational Clichés: Constraining Constructive Induction During Relational Learning. In *Proceedings of the 8th International Workshop on Machine Learning (ML-91)*. Morgan Kaufmann, San Mateo, CA, 1991.

[21] Srinivasan, A., Muggleton, S., King, R.D.: Comparing the use of background knowledge by Inductive Logic Programming systems. In *Proceedings ILP-95*, Katholieke Universiteit Leuven, Belgium, 1995.

[22] Turney, P.D.: Cost-Sensitive Classification: Empirical Evaluation of a Hybrid Genetic Decision Tree Induction Algorithm. *Journal of Artificial Intelligence Research*, **2**, 1995, 369–409.

[23] Wang, Y., Witten, I.: Inducing Model Trees for Continuous Classes. In *Poster Papers – 9th European Conference on Machine Learning*, 1997, 128–137.