Predicting Outliers

Luis Torgo[1] and Rita Ribeiro[2]

[1] LIACC-FEP, University of Porto, R. Campo Alegre, 823, 4150 Porto, Portugal
ltorgo@liacc.up.pt
http://www.liacc.up.pt/~ltorgo
[2] LIACC, University of Porto, R. Campo Alegre, 823, 4150 Porto, Portugal
rita@liacc.up.pt

Abstract. This paper describes a method designed for data mining applications where the main goal is to predict extreme and rare values of a continuous target variable, as well as to understand under which conditions these values occur. Our objective is to induce models that are accurate at predicting these outliers but are also interpretable from the user perspective. We describe a new splitting criterion for regression trees that enables the induction of trees achieving these goals. We evaluate our proposal on several real world problems and contrast the obtained models with standard regression trees. The results of this evaluation show the clear advantage of our proposal in terms of the evaluation statistics that are relevant for these applications.

1 Introduction

The work described in this paper addresses applications where the main objective is to model rare extreme values, usually known as outliers. Given that the target variable is continuous we are facing regression problems. However, the main difference to standard regression tasks is that our main interest is to predict accurately the occurrences of rare high or low values of the target variable. A typical real world application is the prediction of stock market returns, where small and highly frequent returns are irrelevant for investors, while large movements of the market are the key events where accurate prediction pays off. Our interest is not only to anticipate the occurrence of an extreme value but also to be accurate at predicting its concrete value, because the amplitude of the outlier is relevant for the user of these applications, as it may lead to differentiated actions. Another major requirement of our target applications is the interpretability of the models. This means that discovering the conditions that lead to these extreme values is also a major goal of our models.

Applications where the main modeling objective are rare events abound in recent data mining literature. Nevertheless, existing related work is mostly focused on discrete target variables (i.e. classification tasks). These works include topics like activity monitoring [4], prediction of rare events [17,18], anticipation of surprising patterns [7], novelty detection, anomaly detection, among others. Most of this research is also linked to applications where a data stream is being

N. Lavrač et al. (Eds.): PKDD 2003, LNAI 2838, pp. 447–458, 2003.

monitored with the goal of anticipating rare events, that is time-dependent data. This research is usually focused in the task of distinguish between interesting cases and "normal" occurrences.

The importance and impact of rare cases has been the topic of research on small disjuncts (e.g. [6,19]). This research is again mainly focused on classification tasks and is also strongly related to the study of applications with unbalanced class distributions (e.g. [5]).

A frequent strategy to bias the models towards being accurate in particular types of cases is the use of differentiated misclassification costs (e.g. [16]). This is a common practice in classification tasks and was also used in solving regression problems through a classification approach [15].

All these classification approaches do not solve the problem of being able to accurately predict the specific value of outliers, and are particularly inadequate when these spread over a wide range of values. If the amplitude of the extreme values is relevant for the user, for instance for taking different actions, all these approaches based on classification are not applicable. Obviously, one could further divide the classes representing the extreme values into more specific classes to differentiate their importance but that would mean that we would partition an already low populated class into several classes, thus making our modeling task even more difficult. As such, for this kind of applications only a regression model can handle the problem properly.

Buja and Lee [2] have recently presented a series of new splitting criteria for both classification and regression trees that address related problems. Regarding regression, they propose two different splitting criteria with two objectives: identifying extreme buckets of the data; and identifying pure (low variance) buckets. The first objective is particularly related to ours. The goal of Buja and Lee is to identify areas of the regression surface where the target variable shows a high or low mean value. Although our goal is related to this, we are particularly interested in applications where these extreme values are rare, which demands for specific criteria.

We propose a new splitting criterion for regression trees which enables the induction of models that meet our application requirements. In Section 2 we formalize our target problems and propose evaluation criteria that should guide the search for the best models. Section 3 describes the details of our proposal. The experimental evaluation of this proposal is presented in Section 4. We finish with the conclusions of this work and future research directions.

2 Problem Formulation

In this section we present a general description of our problem. Let D be a data set, consisting of n cases $\{\langle \mathbf{x}_i, y_i \rangle\}_{i=1}^{n}$, where $\mathbf{x}_i$ is a vector of p discrete or continuous variables, and y_i is a continuous target variable value. As we have mentioned before, we are interested in models that are able to predict accurately rare extreme values of Y. To achieve this goal we need to formalize the notion of rare extreme values. We use the statistical notion of outlier with this purpose.

Box plots are visualization tools that are often used to identify outliers. Extreme values are defined in these plots as values above or below the so-called adjacent values [3]. Let r be the interquartile range defined as the difference between the 3rd and 1st quartiles of the target variable. The upper adjacent value, adj_H, is defined as the largest observation that is less or equal to the 3rd quartile plus $1.5r$. Equivalently, the lower adjacent value, adj_L, is defined as the smallest observation that is greater or equal to the 1st quartile minus $1.5r$. Given these two limits we can define our rare extreme values as,

$$\begin{aligned} O &= \{y \in D \mid y > adj_H \vee y < adj_L\} \\ O_H &= \{y \in D \mid y > adj_H\} \\ O_L &= \{y \in D \mid y < adj_L\} \end{aligned} \tag{1}$$

Depending on the application we may have either O_L or O_H empty[1]. Figure 1 shows the box plots of the targets in two applications where we have different types of outliers. These values are drawn with circles in these graphs.

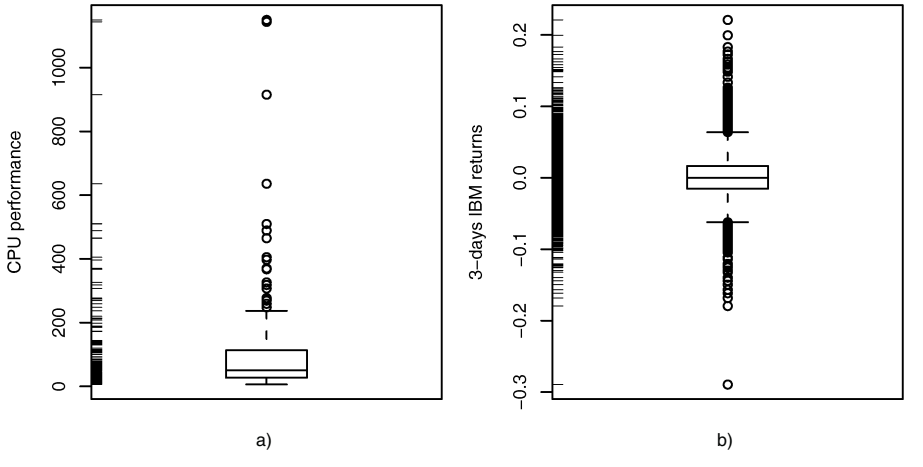

Fig. 1. Two example box plots with different types of extreme values: a) The relative performance of a set of CPUs; b) The 3-days returns of IBM closing prices.

Having described the main features of our target applications we need to define some evaluation criteria to guide the search for the best models. Typical performance measures used in regression settings, such as the mean squared error, are inadequate as they do not stress the fact that we are only interested in the performance in extreme values. This is the same kind of phenomenon as

[1] We will discard applications where both sets are empty as these are not relevant for this study.

the one reported regarding the use of classification accuracy on problems with unbalanced class distributions [8,10].

In the information retrieval literature (e.g. [9]) the notion of relevance seems particularly adequate to our needs. Relevance is defined as the value or utility of a system output as a result of a user search. Relevance is most of the times assessed using two measures: *precision* and *recall*. Precision is defined as the proportion of the cases predicted as target events that really are target events. Recall is defined as the proportion of existing target events that are captured by the model. Our proposal consists of adapting these two measures to our problem setup with the goal of developing a learning tool that maximizes the relevance of the induced model to our application goals.

We define recall in the context of our target applications as the proportion of outliers in our data that are predicted as such (i.e. covered) by our model,

$$recall = \frac{|\ \{\hat{y} \in \hat{Y}_O \mid (y \in O_H \wedge\ \hat{y} > adj_H) \vee (y \in O_L \wedge \hat{y} < adj_L)\}\ |}{|\ O\ |} \tag{2}$$

where $\hat{Y}_O$ is the set of $\hat{y}$ predictions of the model for the outlier cases (i.e. O).

With respect to precision, if we use its standard definition we have,

$$precision_{stand} = \frac{|\ \{\hat{y} \in \hat{Y}_O \mid (y \in O_H \wedge\ \hat{y} > adj_H) \vee (y \in O_L \wedge \hat{y} < adj_L)\}\ |}{|\ \{\hat{y} \in \hat{Y} \mid \hat{y} < adj_L \vee \hat{y} > adj_H\}\ |} \tag{3}$$

where $\hat{Y}$ is the set of $\hat{y}$ predictions of the model.

However, this definition is not adequate to our goals. For instance, with this formulation, assuming $adj_H = 5.6$, a predicted value of 5.8 would have the same value as a prediction of 10.1, for a test case where the true value is 10.5. In our applications this is not acceptable. Otherwise, the best solution would probably be to discretize the target variable and handle the problem as a classification task with differentiated misclassification costs. As we want to distinguish this kind of errors we need to use another definition of precision ($precision_{regr}$) that takes into account the distance between the predicted and true values. At the same time we want to maintain the scale of the measure within the 0..1 interval so that we are able to integrate recall and precision into a single measure using standard approaches. Our proposed definition of $precision_{regr}$ is the following,

$$precision_{regr} = 1 - NMSE_O \tag{4}$$

where $NMSE_O$ is the normalized squared error of the model for the outliers,

$$NMSE_O = \frac{\sum_{y_i \in O} (\hat{y}_i - y_i)^2}{\sum_{y_i \in O} (\bar{Y} - y_i)^2} \tag{5}$$

The value of $NMSE_O$ will usually be between 0 and 1. For the cases where this value goes above 1, which means that the model is performing worse than the naive average model, we consider that the precision of the model is 0.

Obtaining an overall evaluation measure from the values of recall and precision provides a global preference criterion that can be used to guide the search for the models. The F-measure [11] is among the most used measures and is defined as,

$$F = \frac{\left(\beta^2 + 1\right) \cdot precision \cdot recall}{\beta^2 \cdot precision + recall} \tag{6}$$

where β controls the relative importance of recall to precision. This is the definition we use replacing *precision* by our proposed $precision_{regr}$.

3 An Approach Using Regression Trees

Regression trees are known for their computational efficiency, model interpretability and competitive accuracy. For these reasons we have decided to use these models as the base paradigm behind our proposal.

Standard regression trees are obtained using a procedure that minimizes the squared error. This means that the best splits for each tree node are chosen to minimize the weighed squared error between the two branches. As mentioned by Buja and Lee [2] this criterion is not adequate for several data mining applications. That is also the case of our target problems. Moreover, outliers can be a problem for standard regression trees as they may distort the selection of the best splits and may also have a large impact on the average values chosen for the leaves of the trees [14].

The main idea of our proposal to avoid the problems reported above is to use the F-measure presented in Equation (6) to guide the split selection procedure used to grow the trees. As such, the key distinguishing feature of our method is the criterion used to select the best test for each tree node. In our proposal the best split s^*, is chosen using the following criterion,

$$s^*(D_t) = \max_{s \in S} \max\left(F(D_{t_L}), F(D_{t_R})\right) \tag{7}$$

where S is the set of trial splits for the node t [2]; D_{t_L} is the subset of cases in t (D_t) that satisfy the test s (i.e. the left sub-branch of t), while D_{t_R} contains the remaining cases (i.e. $D_{t_R} = D_t - D_{t_L}$); and $F(D)$ is the F-measure for a set of cases.

In order to obtain the F-measure for the branches of a candidate split we need to obtain the values of precision and recall, which we do using the following formulas,

[2] That are the same as in a standard regression tree.

$$precision_{regr_t} = \begin{cases} 1 - \frac{\sum_{y_i \in O_H(D_t)} (\bar{y}_t - y_i)^2}{\sum_{y_i \in O_H(D_t)} (\bar{Y} - y_i)^2} & \text{if } \bar{y}_t > adj_H \ \vee \ \bar{y}_t > \tilde{y}_t \\ 1 - \frac{\sum_{y_i \in O_L(D_t)} (\bar{y}_t - y_i)^2}{\sum_{y_i \in O_L(D_t)} (\bar{Y} - y_i)^2} & \text{if } \bar{y}_t < adj_L \ \vee \ \bar{y}_t \leq \tilde{y}_t \end{cases} \quad (8)$$

where $O_H(D_t)$ ($O_L(D_t)$) is the set of cases of node t that belong to $O_H(O_L)$; $\bar{y}_t$ is the average Y value in the node; $\tilde{y}_t$ is the median Y of the node; and $\bar{Y}$ is the average Y in the training data.

This means that depending on the value of the node average we consider this branch as a tentative to predict high or low outliers, and calculate its precision accordingly. Even if the node average is not in the outlier range of values we still calculate the precision in the node, using the median as a threshold for deciding whether to calculate it with respect to high or low outliers.

Regarding recall we use,

$$recall_t = \begin{cases} 0 & \text{if } \bar{y}_t \geq adj_L \ \wedge \ \bar{y}_t \leq adj_H \\ \frac{| \ y \in D_t \ \wedge \ y \in O_H \ |}{| \ O_H \ |} & \text{if } \bar{y}_t > adj_H \\ \frac{| \ y \in D_t \ \wedge \ y \in O_L \ |}{| \ O_L \ |} & \text{if } \bar{y}_t < adj_L \end{cases} \quad (9)$$

When a trial split leads to a branch having an average target value that is not an outlier, the respective recall is zero. This would lead to an F value of zero according to Equation (6). This is a common situation particularly in top level nodes, where the partitions are still too big, and thus the average Y is seldom an outlier. Moreover, sometimes all trial splits for a node are in these circumstances. This means that we are not able to select the best split for these nodes as all splits have the same score, and thus the tree growth procedure would stop prematurely. These situations occur because in complex applications we seldom find a single split that is able to isolate extreme values in one of the branches so that the branch has an average target that is an outlier. This problem decreases as the tree grows because the number of cases in the nodes gets smaller and thus finding such splits is easier. Although these top level splits have zero recall we should still be able to establish a preference criterion to select one, because we can calculate their precision. In order to overcome this difficulty we have added a small threshold[3] to the value of recall in Equation (6) so that the value of F is not zero even when the recall is null.

Summarizing, our proposal consists of selecting the splits that are able to generate a branch (a subset of cases) with a high value of the F-measure. Notice, that we do not search for a weighted solution between the two branches.

[3] We have used the value of 0.001 in our experiments.

Even if one of the branches as a poor F score, as long as the other achieves a high F-measure we have a good candidate split. This strategy is similar to the one followed by Buja and Lee [2], which also do not search for splits with a good compromise between the left and right branches. These strategies lead to unbalanced trees. Still, we share the opinion of Buja and Lee that consider these trees more interpretable.

Another important question that needs to be addressed when developing a tree-based system, is the tree growth stopping criteria. This is a statistical estimation problem and most systems use a two-stages procedure consisting of growing an overly large tree (possibly overfitting the training data), and then use some statistical estimation procedure (e.g. cross validation) for post-pruning this tree[4]. Given that outliers are insignificant from a statistical perspective, these strategies are difficult to implement in our system because they are based on statistical significance. Because of this we have decided not to post-prune our trees. This is consistent with what is mentioned by Weiss and Hirsh [19] in the context of learning from small disjuncts. These authors mention that pruning is considered questionable when the learning objectives are small subsets of cases.

Our method obtains a tree model in a single stage, stopping the tree growth when one of the following conditions arise:

- The F-measure of the node is above a certain user-definable threshold,
- Or the node does not contain any extreme value (i.e. $D_t \cap O = \phi$).

In order to illustrate the effects of using the proposed splitting criteria as opposed to standard least squares methods, we describe a small example application. Due to space reasons we have chosen a dataset that leads to small trees. We have used the well-known CPU performance dataset. In this domain the task is to predict the relative performance of a set of CPUs given some hardware characteristics of these machines. The dataset has 23 high outlier values (values above 237, c.f. Figure 1a). Using a CART-alike regression tree[5] with a standard 1-SE cross validation pruning algorithm [1], we get the tree on the right-hand side of Figure 2. From the point of view of outliers this tree isolates two classes of outliers, both formed by machines with a maximum main memory size above 28000Kb: One class is less extreme in terms of performance (average performance of 299) and includes machines with cache size below 80Kb; and the other class contains machines with larger cache size that have higher performance (average of 667). According to this tree, all computers with less than 28000Kb memory have low performance. Still, there are three exceptions to this (the numbers between parentheses on each node are the number of outliers in that node) that are neglected by this tree. The solution of our model is given at the left hand-side of Figure 2. Our tree is much more specific in terms of describing the conditions leading to outliers. Moreover, it further distinguishes

[4] See [13] for an overview of pruning methods for regression trees.

[5] In this paper we have used as base implementation of regression trees the package *rpart* [12] of the open source statistical software R (www.r-project.org). This package is a close re-implementation of most of CART's [1] features.

the type of outliers. Namely, we can identify even more extreme performance machines that have a high main memory size (above 48000Kb). Our tree also describes the machines with an outlier performance that have less than 28000Kb memory. This tree is clearly more consistent with the distribution of the outliers (i.e. the type of machines with high performance), as it can be seen from the box plot of the target variable presented in Figure 1a. Although one may think that this tree could be simply overfitting the data, the fact is that as we will see on the results of our experiments for this domain, our models achieve a significantly higher precision, recall and F-value.

In summary, from the perspective of understanding the type of extreme values occurring in this domain, and also under which conditions these appear, we claim that our tree is more informative than a standard regression tree. Moreover, this higher interpretability is accompanied by better accuracy as it will be shown in Section 4.

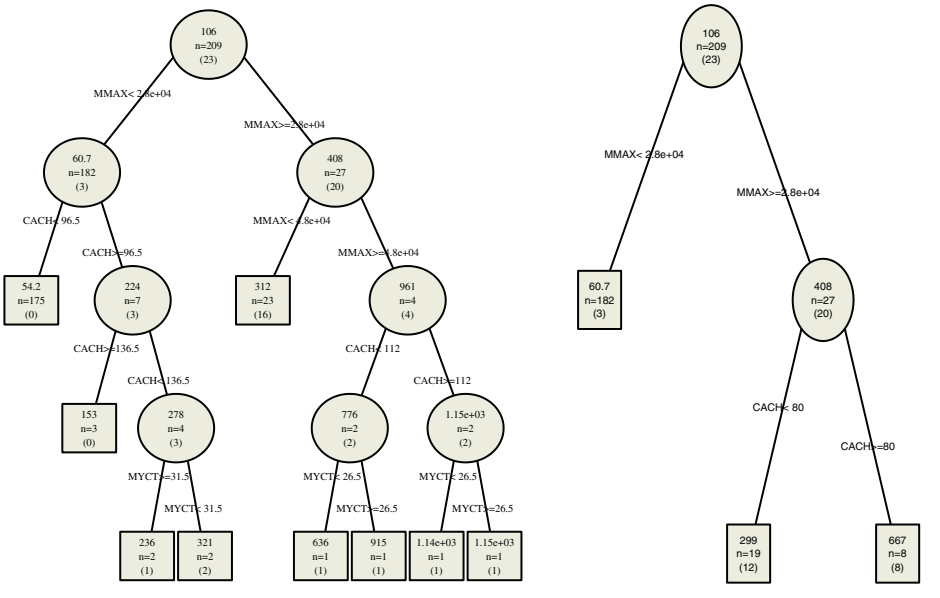

Fig. 2. Our regression tree vs the tree obtained by a CART-alike system, on the machine CPU dataset.

4 Experimental Results

In this section we perform an experimental analysis of the trees obtained with our method. Our analysis compares our proposal to its base paradigm, standard regression trees.

We have carried out a series of experiments using the datasets described in Table 1. These datasets include applications obtained from standard repositories as well as some commercial applications.

Table 1. Datasets description.

Datasets	# cases	continuous attr.	nominal attr.	# outliers	low outliers	high outliers
servo	167	0	4	30	0	30
triazines	186	60	0	9	9	0
algae1	200	8	3	12	0	12
algae2	200	8	3	10	0	10
algae3	200	8	3	22	0	22
algae4	200	8	3	16	0	16
algae5	200	8	3	13	0	13
algae6	200	8	3	19	0	19
algae7	200	8	3	21	0	21
machine_cpu	209	6	0	23	0	23
china	217	9	0	19	0	19
Boston	506	13	0	37	0	37
onekm	710	14	3	8	6	2
cw.drag	1449	12	2	52	1	51
co2.emission	1558	19	8	23	0	23
acceleration	1732	11	3	26	0	26
available.power	1802	7	8	121	0	121
bank8FM	4499	8	0	69	0	69
delta.ailerons	7129	5	0	107	41	66
ibm	8166	10	0	325	140	185
cpu.small	8192	12	0	430	430	0
delta.elevators	9517	6	0	132	60	72
cal.housing	20460	8	0	1071	0	1071
add	30000	10	0	63	0	63
fried.delve	40768	10	0	25	6	19

We have carried out 5 repetitions of 10-fold cross validation experiments using these datasets. These experiments were designed with the goal of estimating the average difference in precision, recall, and F-measure, between a standard regression tree and our proposed method. For the standard method we have used the package *rpart* of R, using cross validation error-complexity pruning with the 1-SE rule according to the method in [1]. Regarding our method we have used a F-value of 0.7 as threshold for deciding when to stop tree growth. The statistical significance of the observed differences was asserted through paired t-tests. Differences that are significant at the 95% level were marked with one sign, while differences significant at 99% have two signs. Plus (+) signs are used to mark differences favorable to standard regression trees, while minus (−) signs are used to indicate the significant wins of our method. Differences that are not significant at these confidence levels have no sign. The F-measure of each method was calculated with $\beta = 1$, meaning that the same weight was given to precision and recall (c.f. Equation (6)).

The results of our experiments are shown on Table 2. This table shows an overwhelming advantage of our method at least from the perspective of the F-measure, which was the criterion used to grow our trees. In effect, in the 25 datasets there were 18 significant wins, 3 insignificant wins and 4 insignificant losses of our proposal. The advantage is even more remarkable in terms of the

proportion of outliers in the domain that are captured by the model (i.e. the recall). However, the results in terms of precision are not so interesting. We have tried to understand the reasons for this lack of precision in some domains. We have varied the F threshold that guides the criterion for stopping tree growth and have observed some variations on these results that seem to indicate that there is some space for improvement of our method by tunning this parameter. Apparently, tree growth may be stopping too soon for these large datasets, where our performance seems to be degrading. Still, if precision was the key objective we could also tune the β parameter of the F-measure that weights the preference between recall and precision when selecting the best splits of the trees[6].

Some of the results given in Table 2 deserve further explanations. Given the definition of precision we use (Equations (4) and (5)) it may seem strange to see some zero values in precision. These occur because some models have a NMSE at predicting the outliers equal or above one. Namely, for several datasets the CART tree is simply a single leaf node, which leads to a NMSE of one, and thus a precision of zero. The values of zero recall are consequence of models that do not predict any of the outliers as such, which occurs when a tree does not have any leaf with an average value that is an outlier.

Summarizing, the results of these experiments clearly show the advantage of our proposal in terms of predicting outliers. Nevertheless, we think some space is left for improvements particularly in terms of tunning the system by changing the stopping criterion as well as the weight between precision and recall. For large datasets, the best solution would probably be to keep a holdout set for proper tuning of these parameters.

5 Conclusions

We have described a new splitting criteria for regression trees with the goal of addressing a specific class of data mining applications. In these domains the main goal of modeling is to predict accurately outlier values in the target variable and also to understand under which conditions these values occur. Our proposal addresses these application goals by leading to regression trees designed to maximize both the number of outliers that are captured by the model and the precision at predicting their values.

The resulting trees were shown to achieve our goals in an extensive experimental comparison using 25 domains. In these experiments we have compared our approach to a standard regression tree and concluded that our proposal clearly outperforms these trees regarding the evaluation criteria that are adequate for this type of applications.

Regarding future work we plan to investigate more deeply the reasons for the failure of our models in terms of precision in some of the domains. Our current explanation lies on the tree growth stopping criteria and we intend to explore other alternatives to the current user settable threshold on the F-measure value.

[6] In our experiments we used equal weight.

Table 2. Regression trees vs our method in terms of Precision, Recall and F measure.

Datasets	$\textbf{Precision}_{regr}$			Recall			F measure		
	CART	Our method	Signif	CART	Our method	Signif	CART	Our method	Signif
servo	0.7616598	0.7829856		0.8713333	0.9800000	−−	0.8053263	0.8668980	−
triazines	0.1474294	0.3113946	−	0.0400000	0.2233333	−	0.0371048	0.2286451	−
algae1	0.2947144	0.4377268	−−	0.0000000	0.3266667	−−	0.0000000	0.3303920	−−
algae2	0.0000000	0.1394068	−−	0.0000000	0.0700000	−	0.0000000	0.0483434	
algae3	0.0000000	0.0994022	−	0.0000000	0.1820000	−−	0.0000000	0.0766102	−
algae4	0.0000000	0.1034622	−−	0.0000000	0.1416667	−−	0.0000000	0.0875694	−
algae5	0.0000000	0.1286299	−−	0.0000000	0.0673333	−	0.0000000	0.0569739	−
algae6	0.0000000	0.0481409	−	0.0000000	0.1600000	−−	0.0000000	0.0383576	−
algae7	0.0127059	0.0983871	−	0.0100000	0.1336667	−	0.0111917	0.0964774	−
machine.cpu	0.5517675	0.6879704	−−	0.8186667	0.8950000	−	0.6266528	0.7596690	−−
china	0.0000000	0.0740538	−	0.0000000	0.0706667	−	0.0000000	0.0621120	−
Boston	0.8243590	0.8225584		0.7580000	0.7595000		0.7675711	0.7361953	
onekm	0.4001506	0.3350779		0.0166667	0.2600000	−−	0.0059831	0.2701772	−−
cw.drag	0.9250750	0.8269861	++	0.8419906	0.9656190	−−	0.8734481	0.8864179	
co2.emission	0.8052664	0.8100812		0.4213333	0.5826667	−	0.4770925	0.6368632	−
acceleration	0.8964751	0.9010154		0.5600000	0.8210000	−−	0.6287861	0.8265599	−−
available.power	0.9668409	0.8567897	++	0.9091224	1.0000000	−−	0.9353684	0.9217586	
bank8FM	0.9781688	0.9529205	+	0.6532389	0.5876751		0.7592038	0.6971588	
delta.ailerons	0.6442916	0.5902405	++	0.1293077	0.1895810	−	0.1659074	0.2671672	−
ibm	0.0000000	0.0100769	−−	0.0000000	0.0055804	−	0.0000000	0.0038138	−
cpu.small	0.9939290	0.9856554	++	0.8519882	0.8620939		0.9165213	0.9189269	
delta.elevators	0.5980803	0.6054747		0.0419848	0.1734476	−−	0.0719734	0.2595248	−−
cal.housing	0.8425739	0.5854252	++	0.3140477	0.4403454	−−	0.4561053	0.5016178	−−
add	0.9320413	0.7925288	++	0.0000000	0.1496825	−−	0.0000000	0.2125705	−−
fried.delve	0.8816233	0.5351138	++	0.0735714	0.0700794		0.0959500	0.0913309	

References

1. L. Breiman, J. Friedman, R. Olshen, and C. Stone. *Classification and Regression Trees*. Statistics/Probability Series. Wadsworth & Brooks/Cole Advanced Books & Software, 1984.
2. A. Buja and Y.-S. Lee. Data mining criteria for tree-based regression and classification. In *Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 27–36, 2001.
3. W. Cleveland. *Visualizing data*. Hobart Press, 1993.
4. T. Fawcett and F. Provost. Activity monitoring: Noticing interesting changes in behavior. In S. Chaudhuri and D. Madigan, editors, *Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 53–62. ACM, 1999.
5. G.Weiss and F. Provost. The effect of class distribution on classifier learning: An empirical study. Technical Report Technical Report ML-TR-44, Department of Computer Science, Rutgers University, 2003.
6. R. Holte, L. Acker, and B. Porter. Concept learning and the problem of small disjuncts. In N. Sridharan, editor, *Proceedings of the 11th International Conference on Artificial Intelligence*. Morgan Kaufmann, 1989.

7. E. Keogh, S. Lonardi, and W. Chiu. Finding surprising patterns in a time series database in linear time and space. In *8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 550–556, 2002.
8. I. Kononenko and I. Bratko. Information-based evaluation criterion for classifier's performance. *Machine Learning*, 6(1):67–80, 1991.
9. C. Meadow, B. Boyce, and D. Kraft. *Text Information Retrieval Systems*. Academic Press, 2nd edition, 2000.
10. F. Provost, T. Fawcett, and R. Kohavi. The case against accuracy estimation for comparing induction algorithms. In *Proc. 15th International Conf. on Machine Learning*, pages 445–453. Morgan Kaufmann, San Francisco, CA, 1998.
11. C. Van Rijsbergen. *Information Retrieval*. Dept. of Computer Science, University of Glasgow, 2nd edition, 1979.
12. T. Therneau and E. Atkinson. An introduction to recursive partitioning using rpart routines. Technical report, Mayo Foundation, 1997.
13. L. Torgo. A comparative study of reliable error estimators for pruning regression trees. In H. Coelho, editor, *Proceedings of the Iberoamericam Conference on AI (IBERAMIA-98)*, 1998.
14. L. Torgo. A study on end-cut preference in least squares regression trees. In P. Brazdil and A. Jorge, editors, *Proceedings of the Portuguese AI Conference (EPIA 2001)*, number 2258 in LNAI, pages 104–115. Springer, 2001.
15. L. Torgo and J. Gama. Regression using classification algorithms. *Intelligent Data Analysis*, 1(4), 1997.
16. P. Turney. Types of cost in inductive learning. In *Proceedings of the Workshop on cost-sensitive learning at the 17th ICML*, pages 15–21, 2000.
17. G. Weiss and H. Hirsh. Learning to predict rare events in event sequences. In R. Agrawal, P. Stolorz, and G. Piatetsky-Shapiro, editors, *Fourth International Conference on Knowledge Discovery and Data Mining (KDD'98)*, pages 359–363, New York, NY, 1998. AAAI Press, Menlo Park, CA.
18. G. Weiss and H. Hirsh. Learning to predict extremely rare events. In *AAAI Workshop on Learning from Imbalanced Data Sets*, pages 64–68. Technical Report WS-00-05, AAAI Press, 2000.
19. G. Weiss and H. Hirsh. A quantitative study of small disjuncts. In *Proceedings of AAAI/IAAI*, pages 665–670, 2000.