Decision Trees Using the Minimum Entropy-of-Error Principle

J.P. Marques de Sá[1], João Gama[2], Raquel Sebastião[3], and Luís A. Alexandre[4]

[1] INEB-Instituto de Engenharia Biomédica, Porto, Portugal
[2] LIAAD – INESC Porto, L.A. and Faculty of Economics, Porto, Portugal
[3] LIAAD – INESC Porto, L.A. and Faculty of Science, Porto, Portugal
[4] Informatics Dept., Univ. Beira Interior, Networks and Multim. Group, Covilhã, Portugal
jmsa@fe.up.pt, jgama@fep.up.pt, raquel@liadd.up.pt, lfbaa@di.ubi.pt

Abstract. Binary decision trees based on univariate splits have traditionally employed so-called impurity functions as a means of searching for the best node splits. Such functions use estimates of the class distributions. In the present paper we introduce a new concept to binary tree design: instead of working with the class distributions of the data we work directly with the distribution of the errors originated by the node splits. Concretely, we search for the best splits using a minimum entropy-of-error (MEE) strategy. This strategy has recently been applied in other areas (e.g. regression, clustering, blind source separation, neural network training) with success. We show that MEE trees are capable of producing good results with often simpler trees, have interesting generalization properties and in the many experiments we have performed they could be used without pruning.

Keywords: decision trees, entropy-of-error, node split criteria.

1 Introduction

Decision trees are mathematical devices largely applied to data classification tasks, namely in data mining. The main advantageous features of decision trees are the semantic interpretation that is often possible to assign to decision rules at each tree node (a relevant aspect e.g. in medical applications) and to a certain extent their fast computation (rendering them attractive in data mining applications).

We only consider decision trees for classification tasks (although they may also be used for regression). Formally, in classification tasks one is given a dataset X as an $n \times f$ data (pattern feature) matrix, where n is the number of cases and f is the number of features (predictors) and a target (class) vector T coding in some convenient way the class membership of each case x_i, $\omega_j = \omega(x_i)$, $j = 1,\ldots,c$, where c is the number of classes and ω is the class assignment function of X into $\Omega = \{\omega_j\}$. The tree decision rules also produce class labels, $y(x_i) \in \Omega$.

In automatic design of decision trees one usually attempts to devise a feature-based partition rule of any subset $L \subset X$, associated to a tree node, in order to produce m

X. Jiang and N. Petkov (Eds.): CAIP 2009, LNCS 5702, pp. 799–807, 2009.

subsets $L_i \subset L$ with "minimum disorder" relative to some m-partition of Ω, ideally with cases from a single class only. For that purpose, given a set L with distribution of the partitioned classes $P(\omega_i \mid L)$, $i = 1,\ldots,m$, it is convenient to define a so-called *impurity* (disorder) function, $\phi(L) \equiv \phi\big(P(\omega_1 \mid L),\ldots,P(\omega_m \mid L)\big)$, with the following properties: a) ϕ achieves its maximum at (1/m, 1/m,…, 1/m); b) ϕ achieves its minimum at (1,0,…,0), (0,1,…,0),…,(0,0,…,1); c) ϕ is symmetric.

We only consider univariate decision rules, $y_j(x_i)$ relative to two-class partitions (m=2), also known as Stoller splits (see [3] for a detailed analysis), which may be stated as step functions: $x_{ij} \le \Delta$, $y_j(x_i) = \omega_k$; $\overline{\omega}_k$, otherwise (x_{ij} is one of the x_i features). The corresponding trees are binary trees. For this setting many impurity functions have been proposed with two of them being highly popularized in praised algorithms: the Gini Index (*GI*) applied in the well-known CART algorithm pioneered by Breiman and co-workers [2], and the Information Gain (*IG*) applied in the equally well-known algorithms ID3 and C4.5 developed by Quinlan [7, 8].

The *GI* function for two-class splits of a set L is defined in terms of

$$\phi(L) \doteq g(L) = 1 - \sum_{j=1}^{2} P^2\big(\omega_j \mid L\big) \in [0,0.5];$$

$$\text{namely, } GI_y(L) = g(L) - \sum_{i=1}^{2} P\big(L_i \mid L\big) g_y\big(L_i \mid L\big)$$

In other words, *GI* depends on the average of the impurities $g_y(L_i)$ of the descending nodes L_i of L produced by rule y. Since $g(L)$ doesn't depend on y, the CART rule of choosing the feature which *maximizes* $GI_y(L)$ is equivalent to minimizing the average impurity.

The *IG* function is one of many information theoretic measures that can be applied as impurity functions. Concretely, it is defined in terms of the average of the Shannon entropies (informations) of the descending nodes of node set L:

$$IG_y(L) = info(L) - \sum_{i=1}^{2} P\big(L_i \mid L\big) info_y\big(L_i \mid L\big)$$

$$\text{with } \phi(L) \doteq info(L) = -\sum_{k=1}^{2} P\big(\omega_k \mid L\big) \ln P\big(\omega_k \mid L\big) \in [0,\ ln(2)]$$

Again, maximizing *IG* is the same as minimizing the average Shannon entropy (the average disorder) of the descending nodes. In ID3 and C4.5 log_2 is used instead of *ln* but this is inessential. Also many other definitions of entropy were proposed as alternatives to the classical Shannon definition; their benefits remain unclear.

A fundamental aspect of these impurity measures is that they all are defined in terms of the probability mass functions of the class assignments $P\big(\omega_k \mid L\big)$ and node prevalences $P\big(L_i \mid L\big)$. The algorithms use the corresponding empirical estimates.

The present paper introduces a completely different "impurity" measure. One that does not directly depend on the class distribution of a node, $P\big(\omega_k \mid L\big)$, and the prevalences $P\big(L_i \mid L\big)$, but instead it solely depends on the errors produced by the decision rule:

$$e_i = \omega(x_i) - y(x_i),$$

with convenient numerical coding of $\omega(x_i)$ and $y(x_i)$.

We then apply as "impurity" measure to be minimized at each node the Shannon entropy of the errors e_i. This Minimum Entropy-of-Error (MEE) principle has in

recent years been used with success in many different areas (regression, blind source separation, clustering, etc.); it has also been applied with success in neural network training for data classification (see e.g. [10]).

The present paper describes in section 2 how MEE decision trees can be implemented and how they perform in several real-world datasets in section 3. We also present a comparison of MEE and *IG* behaviors in section 4 and discuss the pruning issue in section 5. Finally we draw some conclusions and present future perspectives in section 6.

2 The MEE Approach

In accordance with [9] we consider $x_{ij} \in \Re$ (i.e. we do not consider categorical predictors), and at each node we assign a code $t \in \{-1, 1\}$ to the each candidate class ω_j. We thus have: $t = 1, \omega(x_i) = \omega_j$; $t = -1, \omega(x_i) = \overline{\omega}_j$ (t meaning $t(\omega(x_i))$). Likewise for $y(x_i)$.

The support of the error random variable E, associated to the errors $e_i = t(\omega(x_i)) - t(y(x_i))$ is therefore $\{-2, 0, 2\}$, with: 0 corresponding to a correct decision; 2 to a misclassification when x_i class is the candidate class and the splitting rule produces the complement; and -2 the other way around.

The splitting criterion is based on the Shannon entropy of E:

$$H_y(E \mid L) = -\left[P_{-1} \ln P_{-1} + P_0 \ln P_0 + P_1 \ln P_1\right] \in \left[0, \ln(3)\right],$$

where $P_{-1} = P_y(E = -2)$, $P_1 = P_y(E = 2)$ and $P_0 = P_y(E = 0) = 1 - P_{-1} - P_1$. Note that contrary to what happened with *GI*, *IG* (and other divulged impurity measures) there is here no room for left and right node impurities and subsequent average. One single function does it all.

Ideally, in the case of a perfect split, the error probability mass function is a Dirac function; i.e., the "errors" are concentrated at zero. Minimizing H_y corresponds to constraining the probability mass function of the errors to be as narrow as possible and usually around zero.

The main algorithmic operations for growing a MEE tree are simple enough and similar to what is done with other impurity measures:

1. At each tree node we are given an $n \times f$ feature matrix X and an $n \times m$ class matrix T (filled with -1, 1).
2. The error probabilities are estimated using:

$$P_2 = p\, n_{1,-1} / n; \quad P_{-2} = (1-p)\, n_{-1,1} / n; \quad P_0 = 1 - P_2 - P_{-2}$$

with $n_{t,t'}$ meaning the number of cases t classified as t' and $p = n_{\omega_j} / n$ the prevalence of the candidate class ω_j.

3. A univariate split y is searched for in the $f \times m$ space minimizing H_y.
4. If a stopping criterion is not satisfied the corresponding left and right node sets are generated and steps 1 through 3 are iterated.

Figure 1 illustrates two entropy-of–error curves relative to the Breast Tissue dataset presented later. In Figure 1a there is a clear class separation: the entropy curve is of

the "convex" type and a global minimum corresponding to the interesting split is found. In Figure 1b the curve is of the "concave" type and the global entropy minimum is useless. As a matter of fact a reasonable split point for this last case would be located near the entropy maximum instead of the minimum. When we say "reasonable" (and later on, optimal) we mean from the probability-of-error point of view.

This phenomenon of the optimal working point for a Stoller split being located near the maximum of the entropy-of-error when there is a large class overlap, had already been studied in detail in [9]. This work also derives for a few class distribution settings the "turning point" when an entropy minimum turns into a maximum as the classes glide and overlap into each other.

In our algorithm we stick to the entropy minimum. This means that we do not consider the possibility of a "reasonable" split when there is considerable class overlap reflected by a "concave"-type entropy curve. This has an impact on the pruning issue as discussed in section 5. For that purpose our algorithm classifies every entropy curve as being of the "concave" or "convex"-type using a very crude rule: set a 100-point grid on the whole feature range and divide it into five equally sized intervals; compare the average of the three central intervals, m_c, with the average of the end intervals, m_e; if $m_c > m_e$ than the curve is classified as "concave", otherwise is classified as "convex". We tried other modifications of this basic scheme but didn't find any clear improvement.

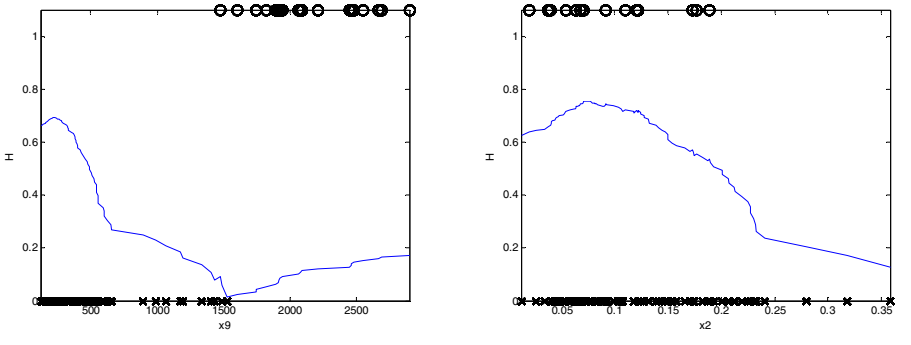

Fig. 1. Entropy-of-error curves for two splits of the Breast-Tissue dataset (splitting the balls from the crosses): a) feature x_9 with class 6; b) feature x_2 with class 2

When there is no valid split for any descendent node of a node L (all entropy curves are concave or the number of cases for any candidate class is very small), the node is considered a leaf.

In a large number of experiments performed with the MEE algorithm we found that one often found better splits (with lower H) when attempting to partition merged classes from the remaining ones. Such "multiclass" splits could even provide good solutions in cases where it was difficult or even impossible to obtain "convex" entropy curves. Figure 2 illustrates an example of a tree with multiclass splits. When evaluating the tree, cases falling into multiclass nodes are assigned to the class with the larger number of cases. The multiclass feature, considering combinations of classes up to $c/2$, is included in the MEE algorithm.

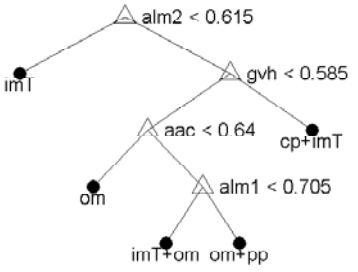

Fig. 2. Tree structure for the Ecoli4 dataset (see below) showing 2-class combinations

3 Application to Real-World Datasets

The MEE algorithm was applied to the datasets presented in Table 1 and its results confronted with those obtained with the CART algorithm implemented by Statistica (StatSoft, Inc.) and the C4.5 implemented by Weka (open source software).

Table 1. Datasets (main characteristics)

	Breast (a)	Breast4 (a)	Olive (b)	Ecoli (c)	Ecoli4 (c)	ImSeg (c)	Glass (c)
No. cases	106	106	572	327	327	2310	214
No. features	9	9	8	5	5	18	9
No. classes	6	4	9	5	4	7	6

(a) "Breast Tissue" dataset described in [6]. Breast4 is Breast reduced to 4 classes: merging {fad,mas,gla}.

(b) "Olive Oils" dataset described in [4].

(c) "E-coli", "Image Segmentation" and "Glass" datasets described in [1]. We removed classes omL, imL and imS from E-coli because they have a low number of cases (resp., 5, 2, 2). Ecoli4 is Ecoli reduced to 4 classes: merging {im, imU}.

All algorithms used unit misclassification costs (i.e., tree costs are misclassification rates). CART and C4.5 used, as is common practice, the so-called midpoint splits: candidate split points lie midway of feature points. In our algorithm we kept the original feature values as split candidate points.

CART was applied with the Gini criterion an cost-complexity pruning [2]. Weka C4.5 applied a postpruning scheme. The MEE algorithm was applied without pruning (justification below).

We applied cross-validation procedures to all datasets, namely leave-one-out with C4.5 and MEE and 25-fold cross-validation to CART (the leave-one-out method wasn't available for CART). Confusion matrices and estimates of the probability of error were computed as well as statistics regarding the tree size (number of nodes).

Table 2 shows the mean error rate and standard deviation (between brackets) for the cross-validation experiments. For the Breast and Ecoli datasets the errors for some classes were always quite high (also found with other classification methods). This led us to merge the poorly classified classes setting up the Breast4 and Ecoli4 datasets (see Table 1).

Table 2. Comparative table of results with mean (std) in cross-validation experiments

	Breast	Breast4	Olive	Ecoli	Ecoli4	Imseg	Glass
CART	0.3679 (0.047)	0.1698 (0.036)	**0.0962** (0.012)	0.2049 (0.022)	**0.1040** (0.017)	0.0675 (0.005)	0.3738 (0.033)
C4.5	**0.3396** (0.046)	0.1226 (0.032)	0.0979 (0.012)	**0.1743** (0.021)	0.1498 (0.020)	**0.0290** (0.003)	0.3224 (0.032)
MEE	0.3679 (0.047)	**0.0943** (0.028)	0.1031 (0.013)	0.2110 (0.023)	**0.1070** (0.017)	0.1182 (0.012)	**0.2664** (0.030)

The three methods were compared using multiple comparison tests based either on the Oneway Anova or the Kruskal-Wallis test according to the p-value of a variance homogeneity test ($p < 0.05$ selects Kruskal-Wallis, otherwise selects Oneway Anova). Multiple comparison was performed at 5% significance level. In Table 2 the significantly best results are printed bold and the significantly worst results are underlined.

4 MEE versus Information Gain

In order to compare both entropy-based criteria, MEE and IG, we generated two-class datasets with an equal number of points, n, represented by 2 features (x_1, x_2) with

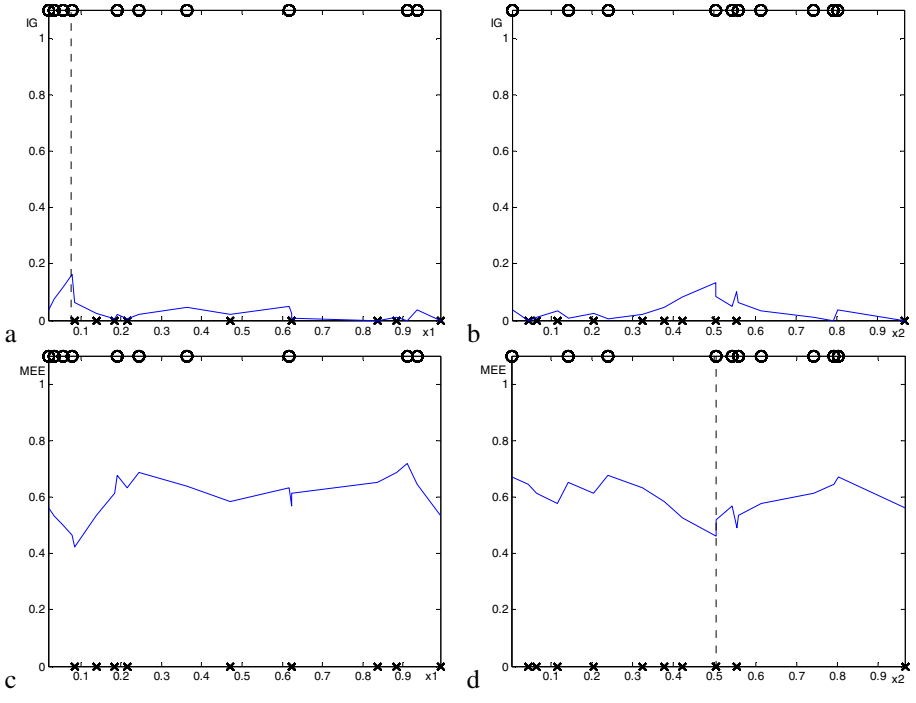

Fig. 3. Comparing IG (top figures) and MEE (bottom figures) in the separation of balls from crosses. IG prefers feature x_1 with IG_{max}=0.1639, whereas for x_2 IG_{max}=0.1325. MEE prefers feature x_2 with MEE=0.4609, whereas no valid minimum is found for x_1.

randomly and uniformly distributed values in [0,1[. One of the features was then selected according to MEE and to IG decision criteria.

For $n = 10$ and several batches of 1000 repetitions of the experiment we found that on average only 1% of the experiments where MEE found a solution that was different from the IG solution. Moreover, we found that all differences between MEE and IG were of the type illustrated in Figure 3. The error probability mass functions for Figure 3a (IG selects x_1) and Figure 3d (MEE selects x_2) are shown in Figure 4. From these figures one concludes that whereas MEE preferred a more "balanced" solution, resembling a Dirac function at zero, IG emphasized the good classification of only one of the two classes, even at the cost of increased errors of the other class.

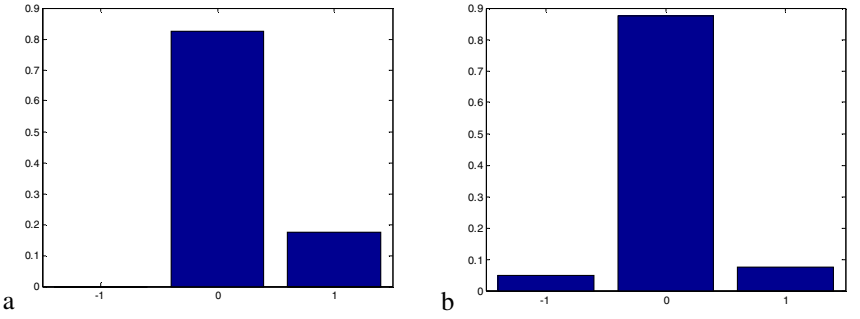

Fig. 4. Probability mass functions of the errors corresponding to: a) Figure 3a (IG selects x_1); b) Figure 3d (MEE selects x_2)

5 The Pruning Issue

Tree pruning is a means of obtaining simpler trees, i.e., simpler models, therefore with better generalization capabilities. CART, C4.5 and other tree design methods employ pruning techniques whenever some evidence of overfitting is found. The MEE method has an important characteristic: it doesn't attempt to find a split whenever the class distributions show a considerable degree of overlap. The quantification on theoretical grounds of what "considerable" means isn't easy. Taking into account the results in [9] one may guess that whenever the distance of the class means is below one pooled standard deviation the entropy-of-error curve will be "concave" and no valid split under the MEE philosophy is found. We believe that this characteristic is one of the reasons why the MEE algorithm always produced smaller trees, on average, than those produced by C4.5 (no tree size statistics were available for CART).

In our experiments MEE trees also showed a tendency to generalize better than those produced by other methods, as measured by the difference between resubstitution estimates of the error rate and the cross-validation estimates with significantly lower $R = |m_R - m_{CV}| / \bar{s}$, where m_R is the mean resubstitution error and m_{CV} the mean cross-validation error.

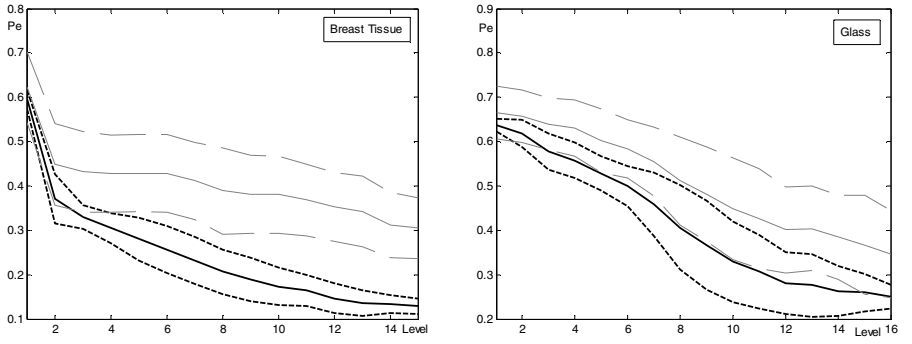

Fig. 5. Mean (solid) and mean±std (dashed) of the training set error (black) and test set error (grey) in 50 experiments on trees designed with 80% of the cases (randomly drawn) and tested in the remaining cases

We have also performed a large number of experiments with the MEE algorithm designing the tree with 80% of randomly chosen cases and testing in the remaining 20% cases, and plotted the mean and mean±standard deviation of the training and test set error estimates along the tree level for 50 repetitions of each tree design. The results of Figure 5 clearly indicate the absence of overfitting. The same conclusion could be drawn in all experiments (over 20 for each dataset) we have carried out.

6 Conclusions

The basic rationale of the MEE approach is that it searches for splits concentrating the error distribution at zero. For the classic approaches what the split is doing in terms of the error distribution is unclear.

From the large number of experiments we carried out we conclude that possible benefits of the MEE trees are the no need of applying a pruning operation and the obtaining of more interesting splits corresponding to errors distributed in a more balanced way as exemplified in section 4. This last aspect could be of interest for some datasets. The results obtained with MEE trees applied to real-world datasets, described in section 3, look quite encouraging especially taking into account that they were obtained with the first version of the algorithm and that there is still much space for improvements.

Besides of introducing obvious improvements in the algorithm (e.g. using midpoint splits) we also intend to study in more detail the following issues: the turning point from “convex” to “concave” behavior of the empirical error distribution; the stopping conditions of the algorithm; Generalization issues such as the evolution of training and test errors with the number of cases. We also intend to study in a comparative way the performance of MEE trees in a larger number of real-world datasets.

References

1. Asuncion, A., Newman, D.J.: UCI Machine Learning Repository. Univ. of California, SICS, Irvine, CA (2007), http://www.ics.uci.edu/~mlearn/MLRepository.html
2. Breiman, L., Friedman, J.H., Olshen, R.A., Stone, C.J.: Classification and Regression Trees. Chapman & Hall/CRC, Boca Raton (1993)
3. Devroye, L., Giörfi, L., Lugosi, G.: A Probabilistic Theory of Pattern Recognition. Springer, Heidelberg (1996)
4. Forina, M., Armanino, C.: Eigenvector Projection and Simplified Nonlinear Mapping of Fatty Acid Content of Italian Olive Oils. Ann. Chim. 72, 127–155 (1981)
5. Loh, W.-Y., Shih, Y.-S.: Split Selection Methods for Classification Trees. Statistica Sinica 7, 815–840 (1997)
6. Marques de Sá, J.P.: Applied Statistics Using SPSS, STATISTICA, MATLAB and R, 2nd edn. Springer, Heidelberg (2007)
7. Quinlan, J.R.: Induction of Decision Trees. Machine Learning 1, 81–106 (1986)
8. Quinlan, J.R.: C4.5 Programs for Machine Learning. Morgan Kaufmann, San Francisco (1993)
9. Silva, L.M., Felgueiras, C.S., Alexandre, L.A., Marques de Sá, J.: Error Entropy in Classification Problems: A Univariate Data Analysis. Neural Comp. 18, 2036–2061 (2006)
10. Silva, L.M., Embrechts, M.J., Santos, J.M., de Sá, J.M.: The influence of the risk functional in data classification with mLPs. In: Kůrková, V., Neruda, R., Koutník, J. (eds.) ICANN 2008, Part I. LNCS, vol. 5163, pp. 185–194. Springer, Heidelberg (2008)