Hindawi Publishing Corporation
Advances in Artificial Intelligence
Volume 2009, Article ID 134807, 13 pages
doi:10.1155/2009/134807

Research Article

A New Information Measure Based on Example-Dependent Misclassification Costs and Its Application in Decision Tree Learning

Fritz Wysotzki and Peter Geibel

Faculty of Electrical Engineering and Computer Science, University of Technology Berlin, Sekr. FR 5-8, Franklinstraße 28/29, D-10587 Berlin, Germany

Correspondence should be addressed to Fritz Wysotzki, wysotzki@cs.tu-berlin.de

Received 29 December 2008; Revised 2 June 2009; Accepted 21 July 2009

Recommended by Rattikorn Hewett

This article describes how the costs of misclassification given with the individual training objects for classification learning can be used in the construction of decision trees for minimal cost instead of minimal error class decisions. This is demonstrated by defining modified, cost-dependent probabilities, a new, cost-dependent information measure, and using a cost-sensitive extension of the CAL5 algorithm for learning decision trees. The cost-dependent information measure ensures the selection of the (local) next best, that is, cost-minimizing, discriminating attribute in the sequential construction of the classification trees. This is shown to be a cost-dependent generalization of the classical information measure introduced by Shannon, which only depends on classical probabilities. It is therefore of general importance and extends classic information theory, knowledge processing, and cognitive science, since subjective evaluations of decision alternatives can be included in entropy and the transferred information. Decision trees can then be viewed as cost-minimizing decoders for class symbols emitted by a source and coded by feature vectors. Experiments with two artificial datasets and one application example show that this approach is more accurate than a method which uses class dependent costs given by experts a priori.

1. Introduction

The inductive construction of classifiers from training sets is one of the most common research areas in machine learning (ML) and therefore in human-computer interaction. The traditional task is to find a hypothesis (a classifier) that minimizes the mean classification error (see, e.g., [1] for an overview). However, as already stated in the technological roadmap of the MLnetII project (Network of Excellence in Machine Learning [2]), the consideration of costs in learning and classification is one of the most relevant fields in machine learning research and many applications. Turney [3] gives an overview of the possible types of costs which may appear in inductive concept learning in general, and Ling and Sheng [4] present an overview of cost-sensitive learning methods. In this paper, we will consider the case of classification learning through the construction of decision trees, which minimize the mean cost of false classifications (with error minimization as a special case). Ling et al. [5], for example, describe the construction of decision trees guided by a minimization of total costs (including costs for misclassification and attribute measurement). Drummond and Holte [6] investigate the cost-sensitivity of four commonly used attribute selection criteria in tree learning.

One way to incorporate costs in classification learning is to use a cost function that specifies the mean misclassification costs in a class-dependent manner a priori [1, 7–10]. In general, the class-dependent costs c_k for not recognizing class k have to be provided by an expert. Using this type of cost implies that the misclassification costs are assumed to be the same for each example of the respective class k. A more natural approach is to let the cost depend on the individual training example for which it is objectively measured or determined, and not just on its class. The mean cost for a

class or subclass can then be computed ("learned") on an objective basis. We hold that directly using the individual costs may also produce more accurate classifiers as compared with using cost estimates given by experts.

One example of application is the classification of a bank's credit applicants as either "good customers" (who will pay back their credit) or "bad customers" (who are not likely to pay back their loans in full). A classifier for this task can be constructed from the bank's records of past credit applications containing personal information on customers, information on the actual loans (amount, duration etc.), back payments on loans, and the bank's actual profit or loss. The loss occurring in a single case can be seen as the misclassification costs for that example in a natural way. In the case of a good customer, the cost is the bank's loss if that customer has been rejected. Where bad customers are concerned, the cost is simply the actual loss if the loan is not paid back in full. There are a lot of other possible applications where major costs resulting from false decisions have to be avoided. For example, a medical diagnosis may not overlook a dangerous disease like cancer, something that might not be very likely; yet not detecting it could lead to high costs (e.g., the death of the patient in an extreme case). Another example would be searching for texts in a text database such as on the Internet, the importance of which depends on the goals of specific user groups. Yet another application is the modeling of cognitive and general behavioral processes where, for example, emotional evaluation plays a major role.

One approach for using example-dependent costs has already been discussed and applied in the context of rough classifiers [11], the perceptron, piecewise linear classifiers, support vector machines [12–14], in the examination of concept drift [15], and in reinforcement learning and process control [16, 17]. In this paper it is applied to the decision tree learning (see also [18, 19]). From the perspective of information theory, which will be the dominating aspect of this paper, this kind of classifier can be considered a decoder which tries to detect (reconstruct) a class symbol coded by a feature vector, which in turn is transmitted through a (generally noisy) channel. We will also introduce a new, cost-dependent information measure, discuss its properties, and use it in tree construction. We feel that it might be of general importance for information theory.

On the other hand, decision trees have the advantage of being able to be broken up into a set of rules which can be interpreted in a human-like fashion. Therefore, decision tree learning can be used as a tool for automatic knowledge acquisition in human-machine interaction, for example, in expert systems. In the cost-dependent case introduced here, this process can also be controlled by a factor of subjective importance defined by the cost of false decisions.

This article is structured as follows. Section 2 introduces the new, cost-dependent information measure. In Section 3, the CAL5 algorithm for the induction of decision trees [20–22] is introduced with a short overview. Section 4 describes the modification of CAL5 in the case of object dependent costs. Experiments with two artificial domains and the above-mentioned credit problem can be found in Section 5. In Section 6, a comparison with the results of three other algorithms is presented and Section 7 concludes the article.

2. Computation of a Cost-Dependent Information Measure

We start with the introduction of cost-dependent probabilities and a cost-dependent information measure as a generalization of the information measure introduced by Shannon, which only depends on classical probabilities [23]. The term "information measure" refers to the quantity of information produced by a source, which is coded and then transferred through a (generally noisy) channel to a receiver where it is decoded. This is sometimes (especially in ML) called "transinformation," a term we also use in the following. As an example of application, we describe its use in the construction of decision trees as classifiers from training sets consisting of objects/situations, each of which is described by a feature vector, its class, and observed or measured cost for misclassification. This is a relevant task in machine learning, decision, and information theory.

In this construction (learning) of decision trees, which can be viewed as sequential decoders of class symbols coded by feature vectors, the next attribute for branching has to be selected if no unique class decision is possible yet. The attribute x giving the (local) maximum information, that is, the maximum reduction of uncertainty about the classification, is usually used. In the case of attributes with continuous (real) values, an appropriate discretization has to be performed (see Section 3 for details). The information about the classes carried (coded) by a single attribute x is measured by the transinformation I_x [23], which is defined as the difference of the entropy H_k of the classes before measuring x and $H_{k|x}$, the expectation (mean) of entropy defined for the set of measured values of x. H_k measures the uncertainty of a class decision before measuring x, and $H_{k|x}$ the remaining mean uncertainty (equivocation) after measuring and decoding x. In this context, we regard x and k as stochastic variables with the values x_i $(i = 1,\ldots,m)$ and k_j $(j = 1,\ldots,n)$, respectively. Transinformation is then defined by

$$\begin{aligned} I_x &= H_k - H_{k|x} \\ &= -\sum_{j=1}^{n} p\left(k_j\right) \log p\left(k_j\right) \\ &\quad + \sum_{i=1}^{m} p(x_i) \sum_{j=1}^{n} p\left(k_j \mid x_i\right) \log p\left(k_j \mid x_i\right), \end{aligned} \tag{1}$$

$p(k_j)$ is the a priori probability of class k_j, $p(x_i)$ is the probability of observing the value x_i of attribute x, and $p(k_j \mid x_i)$ denotes the probability of class k_j when an attribute value x_i has been observed.

Now we introduce a cost $c_{k_j}(x_i)$ for not deciding the true class k_j if value x_i is measured. We therefore assume that the costs are independent of the incorrectly predicted class and that the costs for correct classification are zero.

To define a cost-dependent transinformation measure as a generalization of the Shannon information, we introduce the cost-dependent probabilities $p^c(k_j), p^c(k_j \mid x_i), p^c(x_i \mid k_j), p^c(x_i)$ to replace $p(k_j), p(k_j \mid x_i), p(x_i \mid k_j), p(x_i)$ in I_x [12, 18] with the definitions:

$$p^c\left(k_j\right) =_{\text{Def.}} \frac{B_{k_j} p\left(k_j\right)}{\sum_{r=1}^{n} B_{k_r}\ p(k_r)} = \frac{B_{k_j} p\left(k_j\right)}{b}, \tag{2}$$

where

$$\begin{aligned} B_{k_j} &=_{\text{Def.}} \sum_{i=1}^{m} c_{k_j}(x_i)\, p\left(x_i \mid k_j\right), \\ b &=_{\text{Def.}} \sum_{r=1}^{n} B_{k_r} p(k_r), \end{aligned} \tag{3}$$

B_{k_j} denotes the mean cost arrived at by averaging on all objects of class k_j. The value $p^c(k_j)$ is the new, cost-sensitive probability of class k_j arrived at by multiplying the "classical" probability $p(k_j)$ with the mean cost B_{k_j} for not recognizing class k_j normalized by the mean cost b that takes all classes into account.

Furthermore, we define the cost-transformed conditional probabilities

$$p^c\left(k_j \mid x_i\right) =_{\text{Def.}} \frac{c_{k_j}(x_i)\, p\left(k_j \mid x_i\right)}{B_{x_i}}, \tag{4}$$

where

$$\begin{aligned} B_{x_i} &=_{\text{Def.}} \sum_{j=1}^{n} c_{k_j}(x_i)\, p\left(k_j \mid x_i\right), \\ p^c(x_i) &=_{\text{Def.}} \frac{B_{x_i} p(x_i)}{\sum_{r=1}^{m} B_{x_r} p(x_r)} = \frac{B_{x_i} p(x_i)}{b}. \end{aligned} \tag{5}$$

The probabilities $p^c(x_i)$ and the conditional probabilities $p^c(k_j \mid x_i), p^c(x_j \mid k_i)$ are defined according to the same principle used above, that is, by multiplying the classical probabilities with the mean normalized costs. B_{x_i} is the mean cost for misclassification if feature x_i emerges. It can be proved that for the normalization constant b holds

$$b = \sum_{r=1}^{n} B_{k_r} p(k_r) = \sum_{i=1}^{m} B_{x_i}\ p(x_i). \tag{6}$$

It can also be proved that $p^c(k_j), p^c(x_i), p^c(k_j \mid x_i), p^c(x_j \mid k_i)$ are probabilities which do indeed satisfy the axioms and rules that hold for classical probabilities.

The well-known Bayes decision rule decides for the class, which has the highest probability in the cost-free case [24]. Defining new probabilities by multiplying the original ones with the normalized costs is consistent with the cost-sensitive Bayes decision rule including costs for misclassification:

$$\text{Decide for class } k_j \text{ if } B_{k_j} p\left(k_j\right) > B_{k_i} p(k_i) \quad \forall i \neq j, \tag{7}$$

which gives identical decision results if both sides are divided by the normalization constant b. Then the rule can be written in the form:

$$\text{Decide for class } k_j \text{ if } p^c\left(k_j\right) > p^c(k_i) \quad \forall i \neq j, \tag{8}$$

that is, with probabilities again, but cost-dependent ones.

The class decision is usually based on the observed attribute values, not only on the prior class probabilities and costs. In the following, we will restrict our considerations to a single attribute x, but this can be generalized for arbitrary sequences of test attributes. Using conditional probabilities, we have the corresponding decision rule restricted to objects with feature x_k:

$$\begin{aligned} &\text{Decide for class } k_j \text{ if } p^c\left(k_j \mid x_k\right) > p^c(k_i \mid x_k) \\ &\qquad\qquad \text{holds } \forall i \neq j. \end{aligned} \tag{9}$$

When using joint probabilities, this is equivalent to $p^c(k_j, x_k) > p^c(k_i, x_k)$. A special case is $B_{k_j} = c_{k_j}$, that is, if only class dependent costs are considered.

Note that the cost of misclassification also measures the (perhaps subjective or problem-dependent) importance of the class k_j in question: the cost $c_{k_j}(x_i)$ corresponds to the relevance of k_j in a situation (or for an object) characterized by the feature x_i. Similarly, the cost $c_{k_j}(\mathbf{x}^{(p)})$ depends on a feature *vector* $\mathbf{x}^{(p)}$ describing a situation or object in more detail. This is the type of cost associated with training vectors ("example-dependent costs") taken into consideration in this paper.

We have simplified the definitions of cost-dependent probabilities here by regarding the attributes x with discrete values only. For tree construction in the case of attributes with continuous values, these must be discretized to transform them into attributes with discrete values. The discrete values constructed represent intervals I, whose cost function is computed as the mean of $c_{k_j}(\mathbf{x}^{(p)})$ for objects $\mathbf{x}^{(p)}$ falling in that interval (see Section 4).

Now we can define a cost-dependent entropy H_k^c and a cost-dependent transinformation I_x^c by simply replacing in the classical expressions appropriate probabilities with the cost-dependent ones defined above:

$$\begin{aligned} I_x^c &= H_k^c - H_{k|x}^c \\ &= -\sum_{j=1}^{n} p^c\left(k_j\right) \log p^c\left(k_j\right) \\ &\quad + \sum_{i=1}^{m} p^c(x_i) \sum_{j=1}^{n} p^c\left(k_j \mid x_i\right) \log p^c\left(k_j \mid x_i\right), \end{aligned} \tag{10}$$

I_x^c is the cost-dependent transinformation which can be regarded as a generalization of Shannon information and will be used in the construction of cost-dependent decision trees. These decision trees are (locally) optimized during sequential construction to minimize the mean cost of misclassification instead of the mean classification error.

The following proposition states two important properties of the cost-sensitive transinformation and relates it to its classical, cost-independent counterpart.

Proposition 1. (a) *If the costs $c_{k_j}(x_i)$ for all classes k_j and values x_i are equal, one arrives at a cost-independent classical definition of H_k (Shannon entropy) and I_x (Shannon information) as accordingly special cases.*

(b) *If the mean cost B_k for one class k exceeds the mean cost for all other classes, then the cost-dependent probability $p^c(k)$ will be greater than the classical probability $p(k)$. If, in the special case of two classes, there are different mean costs B_k for those two classes, then there will be a larger mean difference of the cost-dependent probabilities compared to the classical probabilities, and the entropy will thus become smaller; that is, it holds that $H_k^c < H_k$. This means that there is a reduction of (subjective) uncertainty. $H_k^c < H_k$ is also reached in the general case of more than two classes if the cost for a class exceeds some threshold depending on the (classical) probability of that class.*

Proof. (a) If $c_{k_j}(x_i) = c$ for all values x_i and all classes k_j, then $p^c(x_i) = p(x_i)$ and $p^c(k_j) = p(k_j)$ because $B_{k_j} = c, B_{x_i} = c$ and $\sum_{r=1}^{n} p(k_r) = 1$; that is, the classical case will emerge.

(b) We consider the case of two classes and set $B_{k_1} = c$ and $B_{k_2} = \varepsilon$. If $c > \varepsilon$, that is, $\varepsilon/c < 1$, then $p^c(k_1) > p(k_1)$ and, subsequently, $p^c(k_2) < p(k_2)$ because

$$p^c(k_1) = \frac{cp(k_1)}{cp(k_1) + \varepsilon p(k_2)} = \frac{p(k_1)}{p(k_1) + (\varepsilon/c)p(k_2)}, \qquad (11)$$

$$p(k_1) + p(k_2) = 1, \qquad p^c(k_1) + p^c(k_2) = 1, \text{ too.}$$

Then we have

$$\begin{aligned} H_k^c &= -p^c(k_1)\log p^c(k_1) - p^c(k_2)\log p^c(k_2) \\ &= -p^c(k_1)\log p^c(k_1) - \left(1 - p^c(k_1)\right)\log\left(1 - p^c(k_1)\right). \end{aligned} \qquad (12)$$

The function $f(p) = -p \log p - (1 - p)\log(1 - p)$ monotonously decreases in the interval $0.5 \le p \le 1.0$ with its maximum at $p = 0.5 = 1 - p$. From $p^c(k_1) > p(k_1) \ge 0.5$, it then follows that $f(p^c(k_1)) < f(p(k_1))$. Because of the symmetry of the entropy function $f(p)$, there is also $H_k^c < H_k$ if $p(k_1) < 0.5$ and $p^c(k_2) > (1 - p(k_1)) = p(k_2)$ since then $p(k_2) > 0.5$ holds. The generalization to more than two classes is shown in the following. If the relative cost ε/c is very small (e.g., c is very large as compared with ε), then in the limit

$$\frac{\varepsilon}{c} \longrightarrow 0, \qquad (13)$$

that is, if $c \to \infty$ or $\varepsilon \to 0$, $p^c(k_1) \to 1$ and $H_k^c \to 0$, and we get a total overgeneralization of class k_1. This also holds for more than two classes with $c > \varepsilon_2, \ldots, c > \varepsilon_n$. In this extreme situation there is no uncertainty; it is then subjectively optimal to decide for class k_1 every time despite the high error rate. It also follows that $H_k^c < H_k$ will hold if the maximum cost c exceeds some threshold. This is a generalization of the case of two classes treated above and means that the decision uncertainty given by the entropy is reduced if there are high costs for not recognizing special classes. □

Remark 1. An attribute x which is irrelevant, that is, does not discriminate between the classes (gives no information about the classes), can become relevant in cost-dependent classification because its values might be associated with different costs. An example of this will be presented in Section 5.2.

Remark 2. As mentioned in the introduction, the results obtained here are applied in cognitive science for a theoretical foundation of cost-controlled human behavior. One application of the explanation of the generation and possible control of psychopathological behavior is described in a paper written together with a well-known German psychotherapist and researcher in psychoanalysis [25].

3. Decision Tree Learning with CAL5

The following section provides an overview of how a decision tree is constructed with CAL5. A comparison with other decision tree algorithms can be found in Section 3.2.

3.1. Overview of CAL5. The CAL5 algorithm [1, 20–22, 26] for learning decision trees for classification and prediction converts real-valued attributes into discrete-valued ones by defining intervals on the real dimension through the use of statistical considerations. The intervals (corresponding to discrete or "linguistic" values) are automatically constructed and adapted to establish an optimal discrimination between the classes through axis-parallel hyperplanes in the feature space. The trees are built top-down in the usual manner through stepwise branching with new attributes to improve the discrimination. An interval in a new dimension x is formed if the hypothesis "one class dominates in the interval" or the alternative hypothesis "no class dominates" can be decided on using a user-defined confidence level by means of the estimated conditional class probabilities. "Dominates" means that the class probability exceeds some threshold given by the user.

In the following, we will give a more detailed-description of CAL5. If, during tree construction, a terminal node (leaf) representing an interval *in which no class dominates* is reached, it will be refined by using the next attribute x for branching. The attribute is selected using the transinformation measure (see Section 2) to decide which is the (locally) best discriminating attribute, that is, which one transfers maximum information on the classes at this node. To determine this locally best discriminating attribute, the following procedure of discretization and computation of transinformation is applied to all attributes occurring in the feature vector (apart from the attribute immediately preceding the actual node on the path leading to it).

We assume that one (real-valued) attribute x is selected. In order to use it to construct a branching and compute its transinformation, intervals defining the discrete values must be constructed first.

In the first step of the interval construction, all values of the training objects reaching the terminal node are ordered along the new dimension x. In the second step, values are collected one by one from left to right, tentatively forming intervals. Now a confidence interval $[\delta_1, \delta_2]$ for each class probability is defined for a user-specified confidence level $1 - \alpha$ to estimate the class probabilities in each current tentative interval I on x. This depends on the frequency n_k of class k, the total number n of objects in I and on α. (Note that I defines a tentative discrete value of the originally numerical attribute x.) A Bernoulli distribution is assumed for the occurrence of class symbols in I. The confidence interval $[\delta_1, \delta_2]$ is derived using the Chebyshev inequality which gives

$$\delta_{1,2}(n_k, n, \alpha) = \frac{2\alpha\, n_k + 1}{2\alpha\, n + 2} \mp \frac{\sqrt{4n_k\alpha(1 - n_k/n) + 1}}{2\alpha n + 2}. \quad (14)$$

Applying a confidence interval for probabilities in the statistical sense makes it easier for us to introduce the confidence of class decisions in the cost-sensitive setting as compared with the use of a geometrical method such as the consideration of the distance of class boundaries as used by, for example, Tóth and Pataki [27, 28].

With the confidence interval, the following "metadecision" is made for each tentative x-interval I.

(1) If, for a class k in I, it holds that

$$\delta_1(n_k, n, \alpha) \geq S, \quad (15)$$

that is, the entire confidence interval is above S, then decide that "class k dominates" in interval I" and close I. This means that $p(k \mid I) \geq S$ holds for the probability of the occurrence of class k in I with a probability (confidence) of $1 - \alpha$. S with $0.5 \leq S \leq 1.0$ is a user-given threshold, defining the maximum admissible error 1-S in a class decision using the tree. If class k dominates in I, the path is closed and k is attached as a final class label to this newly created terminal node when x becomes the attribute with maximum transinformation for the current decision node (see below). The interval I then corresponds to a newly defined discrete value of x.

(2) If, for all classes k in I, it holds that

$$\delta_2(n_k, n, \alpha) < S, \quad (16)$$

that is, if no class dominates, then also close interval I and begin construction of the next adjacent interval.

(3) If neither (1) nor (2) holds, a class decision is rejected and the interval I will be extended by the next attribute value in the ordering on x to enlarge the statistics. Special heuristics (decision for a majority class) are applied in the case where an interval remains in dimension x, which satisfies neither hypothesis (1) nor hypothesis (2), and no extension of the interval is possible due to the finite training set. Finally, adjacent intervals with the same class label are merged.

This procedure is repeated recursively until all intervals of x are constructed after which the transinformation for the discretized attribute x is computed (see Section 2). Note that the statistics for a constructed interval I, that is, the set of class probabilities estimated from the training objects it includes, must be retained for the sake of computing the transinformation, even in case (1) where a unique class decision is possible.

This computation of transinformation is done for all available attributes apart from the one immediately preceding the actual node. The attribute that delivers the maximum transinformation is used for branching. Branching stops if either terminal nodes of the branch contain a unique class label or the statistics in an interval is too poor to make either decision (1) or (2). We will give a more detailed and precise description of the algorithm modified to include example-dependent costs in Section 4.

Note also that interval I and the transinformation both depend on the path from the root of the tree to the actual node (not indicated here for easier reading). Also, because an attribute might be used several times on the same path in the tree, a larger interval of attribute x, which was already constructed at a higher level, may become split up into smaller ones. For example, "large" might become "medium large" and "very large" if the attribute x means "size."

If the costs for not recognizing some classes are given, the original CAL5 uses class dependent thresholds S_k. From decision theory, it follows [20] that one has to choose

$$S_k \approx \frac{\text{const}}{\text{cost}_k}, \quad (17)$$

where cost_k is the cost of the misclassification of an object belonging to class k. The cost must be provided by the user and depends solely on the class. The main aim of this paper is to introduce object-dependent costs given with the training objects. This allows the use of these costs *locally* for the construction of an interval I (defining a region in the feature space together with the intervals of the other attributes on the path) and independently of experts' subjective estimates. This is explained in more detail in the following sections. In Section 5 it will be shown that high costs of misclassification lead to an enlargement of the decision regions for the corresponding classes.

3.2. Comparison with Other Decision Tree Algorithms. CAL5 has been compared with other algorithms for classification learning, particularly within the scope of the famous European STATLOG project described in detail in the book [1]. STATLOG compared 24 algorithms for classification learning (classical statistical methods, modern statistical techniques, machine learning of rules and trees, neural networks) by means of 24 datasets in different types of practical applications (image and letter recognition, bank loans, medical diagnoses, shuttle control, chromosomes and DNA, technical problems, tsetse fly distribution). The tree learning algorithms most similar to CAL5 were Quinlan's famous C4.5 [29] and CART [30]. A comparison will be given here in after. C4.5 uses its information measure for subtree

splitting in the same sense as our CAL5, which is why our new, cost-dependent version introduced above could apply in cases where costs are given. CART uses the Gini index as a splitting criterion that also measures the class impurity at a (temporary) node. The main difference between both algorithms compared with CAL5 is their application of "backward pruning" motivated by "overfitting," since the tree construction has a priori no stopping criterion. (Overfitting is the process of inferring more structure in the tree than justified by the population from which it is drawn (see [1, Sections 5.1.5–5.1.7] for details). In contrast to this, CAL5 uses the confidence intervals for the confident estimation of class probabilities at a leaf for the decision of (immediate) pruning as explained above. The confidence bounds δ_1, δ_2 used in the decisions are based on the parameter α defining the confidence interval, which must be optimized. Another difference is that CAL5 is able to construct more than two values (intervals) of a numerical attribute used for splitting, whereas C4.5 and CART only allow binary splits.

Considering the experimental results of the STATLOG project, CAL5 showed good results on average. Its performance averaged on the 24 datasets used in STATLOG was similar to that of C4.5 (even a little better [26]). In one case (Australian credit dataset), it even achieved the best performance (see the tables of results in [1]). Of course, different applications call for different algorithms. The main advantage of CAL5 is the construction of smaller trees as compared to algorithms using backward pruning [1]. This results in better interpretability and rules that are easier to understand. The reason for this might lie in the use of confident probability estimates and the possibility of constructing discrete-valued attributes with a potentially arbitrary number of values.

An algorithm for the construction of decision trees using "total costs" (defined as the sum of misclassification costs and the cost for measuring an attribute x) where the costs are to be given a priori is described by Ling et al. [5]. The additional use of attribute costs is a possibility to extend CAL5 (if the costs are known a priori) and CAL5_OC if example dependent attribute costs are given, too.

4. The Algorithm CAL5_OC

In this section, we describe an extended version of CAL5 capable of handling example-dependent costs. In Section 4.1 we describe a modified version of the decision rules (1)–(3) defined in the last section. In Section 4.2, a detailed description of the algorithm CAL5_OC is provided based on the ideas presented in Section 4.1.

4.1. Using Cost-Dependent Decision Thresholds in the Case of Object-Dependent Costs. Now we extend the metadecision rules (1)–(3) and the (locally applied) optimal branching procedure introduced in Section 3 for the case in which the training objects originally described by a feature vector and its class are also presented with costs for misclassification. This means that we have a training set

$$\left(\mathbf{x}^{(p)}, c^{(p)}, k^{(p)}\right), \quad p = 1, \ldots, r \tag{18}$$

with feature vector $\mathbf{x}^{(p)}$, class $k^{(p)} \in \{k_j, j = 1, \ldots, n\}$, and cost of misclassification $c^{(p)}$ as a former experience in a situation where $\mathbf{x}^{(p)}$ had been observed and a false decision was made.

The algorithm performs an interval formation if a new branch has to be constructed with attribute x selected. We assume that I is the (perhaps temporary) current interval constructed. It represents a *discrete* value x_i of x. In the following, we use x to denote both the original attribute x and its discretized version, which is to be constructed.

Based on decision theory, the following rule for class decision can be derived (see also Appendix and [18, 20]):

Decide class k_j in interval I if

(a) $c_j(I)p(k_j \mid I) > c_r(I)p(k_r \mid I)$ for all $r \neq j$ *(Bayes' decision rule)*

(b) *and simultaneously*

$$p\left(k_j \mid I\right) > \frac{\sum_{r=1}^{n} c_r(I)p(k_r \mid I) - c_0(I)}{c_j(I)} =_{\text{Def.}} S_j(I). \tag{19}$$

$p(k_j \mid I)$ is the conditional probability of k_j (to be approximated by the relative frequency) in interval I, where I is considered a stochastic variable. $c_j(I)$ is the mean cost for the misclassification of class k_j in interval I and is estimated by

$$c_j(I) \approx \frac{\sum_{\mathbf{x}^{(p)} \in I,\, \mathbf{x}^{(p)} \in k_j} c\left(\mathbf{x}^{(p)}\right)}{n_j(I)}, \tag{20}$$

where the sum ranges over all examples $\mathbf{x}^{(p)}$ for which $x^{(p)} \in I$ and $x^{(p)} \in k_j$ holds. The value of $n_j(I)$ is the number of training vectors $\mathbf{x}^{(p)}$ of class j falling into I. $c_j(I)$ can now be interpreted as $c_{k_j}(x_i)$ (locally) as explained in Section 2 since I represents the discrete value x_i of the discretized attribute x. Note that I and thus the mean cost $c_j(I)$ also depend on the path from the root of the current decision tree to the root node of the new branch (labeled with x) where it was constructed.

$S_j(I)$ defined in (b) is the decision threshold, which now depends on the mean cost $c_j(I)$ in I. $c_0(I)$ is the cost for the rejection of a dominance decision and subsequent further branching. Since it is unknown in advance, we eliminate it through the division of $S_j(I)$ by $S_i(I), i \neq j$. For all classes with indices i and j this gives us

$$\frac{S_j(I)}{S_i(I)} = \frac{c_i(I)}{c_j(I)}. \tag{21}$$

Let $c_{\min}(I)$ be the cost of the class with minimal cost in I, and $S_{\max}(I)$ its threshold. For this class we choose a constant threshold $S_{\max}(I) = S_{\max}$, which is a parameter of the algorithm and, in particular, is independent of I. Note, however, that $S_{\max}$ is related to the unknown cost of rejection

$c_0(I)$ (see the appendix). The value of $S_{\max}$ determines all other thresholds by

$$S_j(I) = \frac{c_{\min}(I)\, S_{\max}}{c_j(I)}, \tag{22}$$

$S_{\max}$ controls the complexity of the resulting tree. It does not change the relations of the other thresholds (see the appendix). The optimal value of $S_{\max}$ must be defined by the user or optimized through a systematic search as we have done in our experiments described in Section 5.

It can now be seen that, as a measure of importance, high costs $c_j(I)$ for the misclassification of a class k_j (e.g., the failed detection of cancer in a medical diagnosis) lead to a lower threshold for class decision; that is, k_j has a higher chance of being decided on in contrast to other classes with lower importance (yet perhaps with a higher degree of probability).

Using the class-dependent thresholds $S_j(I)$, we arrive at modified metadecision rules (see Section 3):

(1) If, for a class k_j in I, it holds that

(a)

$$c_j(I)p\left(k_j \mid I\right) > c_r(I)p(k_r \mid I) \tag{23}$$

or equivalently $p^c(k_j \mid I) > p^c(k_r \mid I)$ for all $r \neq j$ (Bayes' decision rule) and

(b)

$$\delta_1\left(n_j, n, \alpha\right) \geq S_j(I), \tag{24}$$

then decide that "k_j dominates in I." The formula for the estimation of $p^c(k_j \mid I)$ used in our algorithm for learning cost-dependent trees will be given below.

(2) If, for all classes k_j in I, it holds that

$$\delta_2\left(n_j, n, \alpha\right) < S_j(I), \tag{25}$$

that is, no class dominates, then a branch with the next attribute is constructed.

(3) Case (3) of the description given in Section 3 holds without modification.

For applying Bayes' decision rule (1), the value of $p^c(k_j \mid I)$ can be estimated by

$$p^c\left(k_j \mid I\right) \approx \frac{\sum_{\mathbf{x}^{(p)} \in I,\, k^{(p)}=k_j} c\left(x^{(p)}\right)}{\sum_{\mathbf{x}^{(p)} \in I} c\left(x^{(p)}\right)}, \tag{26}$$

Because we would like to find the maximum over all classes, we can skip the denominator because it only depends on the interval.

Note that the transinformation computed for each attribute occurring in the feature vector (apart from the one immediately preceding on the path in the tree) for finding the optimal attribute for the new branch is now cost-dependent, since the intervals I and the thresholds $S_j(I)$ are cost-dependent (and also class-dependent). A high cost for misclassification leads to a low threshold for the corresponding class. This is also interesting from the perspective of modeling human decision-making, controlling attention and consciousness where the subjective importance of the class of the actual object/situation and its context (defined by the path to the leaf where the class to be decided on is located) plays a major role.

4.2. Description of the CAL5_OC Algorithm. The following offers a concise summary and description of the cost-dependent construction of decision trees with CAL5_OC. The algorithm has the following input and parameters:

(i) a training set $M = \{(\mathbf{x}^{(p)}, c^{(p)}, k^{(p)}), p = 1, \ldots, r\}$, where $\mathbf{x}^{(p)}$ is a feature vector representing the p-th example, $c^{(p)}$ is the misclassification cost associated with it, and $k^{(p)}$ is its given class,

(ii) the confidence value $\alpha \in [0, 1]$ for the statistical decision about the dominance of a class in a given interval for any attribute,

(iii) the decision threshold $S_{\max} \in [0, 1]$ used to determine if the "least expensive" class dominates in a given interval.

The output of the algorithm is a decision tree with decision nodes and leaves. A decision node is labeled with an attribute x, and each branch originating from it is labeled with an interval I resulting from the interval construction, that is, discretization process as described in Section 4.1. As usual, the leaves are labeled with classes denoted as k_j.

The learning algorithm constructs intervals for an attribute x in order to determine its cost-sensitive transinformation as explained in Section 2 and for the splitting procedure. The best attribute for a given training set has the highest transinformation and is used to construct the next branch, followed by a recursive application of the algorithm.

We implement the rules (1)–(3) defined in Section 4.1 using the following Boolean *functions* that return either true or false:

(i) *bayes_optimal*(k_j, I): true if $\sum_{\mathbf{x}^{(p)} \in I\ ,k^{(p)}=k_j} c(x^{(p)})$ is maximal,

(ii) *dominates*(k_j, I): *bayes_optimal*(k_j, I) and $\delta_1(n_j, n, \alpha) \geq S_j(I)$ are true,

(iii) *no_dominant_class*(I) :for all classes k_j it is true that $\delta_2(n_j, n, \alpha) < S_j(I)$.

Note that in addition to *dominates*(k_j, I), for some k_j and *no_dominant_class*(I) there is the third possibility in rule (3) that we cannot make a statistically confident decision because there might not be enough examples in I. In the interval algorithm, this triggers the extension of a tentative interval for x so that more examples can be included.

Now we focus on a single attribute x and describe the cost-sensitive interval construction for it. The interval construction is based on the current training set which is either the one that was originally given or the one that occurs in one of the tree algorithm's recursive calls (see below). In other words, it corresponds with the objects "arriving" at a certain node in the tree for which the next attribute is to be selected.

The function *intervals*(x) returns a sequence of intervals $I_0, \ldots, I_T$ constructed for x based on the given examples. The number of intervals T is also determined by the algorithm. The intervals should not be confused with the cost-sensitive transinformation we denoted as I_x^c.

Let $I_t = (y_{l(t)}, y_{u(t)}]$ be an interval of this sequence. Its boundaries $y_{l(t)}$ and $y_{u(t)}$ correspond to specific values in the ordered sequence $y_1, \ldots, y_r$ of all x-values in the training set M extended with the values $-\infty$ and $+\infty$. These boundaries, or rather the *indices* of the lower and upper bounds, $l(t)$ and $u(t)$, will be adjusted by the following algorithm. The result of this algorithm is a sequence of intervals $I_0 = (y_{l(0)}, y_{u(0)}], \ldots, I_t = (y_{l(t)}, y_{u(t)}], \ldots, I_T = (y_{l(T)}, y_{u(T)})$ such that $l(0) = 0$ holds; that is, the lower bound of the first interval is $-\infty$, as well as $u(T) = r + 1$, that is, the upper bound of the last interval is $+\infty$, this means that the intervals cover the entire range of possible values. For two neighboring intervals it holds that $u(t) = l(t + 1)$.

Function intervals(x):

(1) *Sort* the training examples $(\mathbf{x}^{(p)}, c^{(p)}, k^{(p)})$ according to their x-values resulting in a sequence of attribute values $y_1, \ldots, y_r$.

(2) *Set* $y_0 = -\infty$ and $y_{r+1} = +\infty$.

(3) *Set* $t = 0$, $l(t) = 0$ and $u(t) = 1$, that is, the first interval considered is $I_0 = (y_0, y_1]$. t is the index of the interval currently constructed.

(4) *Repeat until* $u(t) = r$ holds, that is, until the last and largest value is reached:

(a) *If dominates*(k_j, I_t) holds for some class k_j or if it holds *no_dominant_class*(I_t), then:

(i) *Set* $t = t + 1$, that is, close the current interval and start working on the next interval.

(ii) *Set* $l(t) = u(t - 1)$ and $u(t) = u(t - 1) + 1$. This means that the next largest attribute value is added to the next interval.

(b) *Else* (when no decision is possible in I)*:*

(i) Extend interval I_t to the right by setting $u(t + 1) = u(t + 1) + 1$, that is, the next largest attribute value from the training set is included.

(5) *Set* $u(t) = r + 1$ (extend last interval to $+\infty$). The total number of intervals is set to $T = t$.

(6) *Return* the sequence of intervals $I_0, \ldots, I_T$.

In a postprocessing stage, it is possible to merge two intervals I_t and I_{t+1} with identical characteristics; that is, either there is the same dominant class, there is no dominant class, or no decision was possible in either interval. It is also possible—and reasonable—to change the interval boundaries so that they lie *in the middle* of two consecutive attribute values. This is achieved for $t < T$ by changing the interval bounds of $I_t = (y_{l(t)}, yu(t)]$ to

$$I_t = \left(\frac{y_{l(t)} + y_{l(t)+1}}{2}, \frac{y_{u(t)} + y_{u(t)+1}}{2} \right). \tag{27}$$

Note that this transformation does not change the statistical values that were computed. We are now ready to define the algorithm for constructing the tree.

Function CAL5_OC $(M, \alpha, S_{\max})$*:*

(1) Evaluate *intervals* (x) for every attribute x and determine its cost-sensitive transinformation I_x^c.

(2) *Let* x be the attribute that has the highest transinformation I_x^c. Let the associated intervals be $I_0, \ldots, I_T$.

(3) Create a decision node labeled with x.

(4) *For* $t = 0, \ldots, T$

(a) *Let* M_t denote the set of examples in M that lie in I_t. This set is defined as

$$M_t = \left\{ \left(\mathbf{x}^{(p)}, c^{(p)}, k^{(p)} \right) \text{ with } x^{(p)} \in I_t \right\}. \tag{28}$$

(b) *If dominates*(k_j, I_t) holds for some k_j, *then*

(i) create a leaf labeled with k_j.

(c) *Else if no_dominant_class*(I_t) holds, *then*

(i) create a subtree recursively by evaluating *CAL5_OC* $(M_t, \alpha, S_{\max})$.

(d) *Else* (if no decision is possible)

(i) create a leaf labeled with a class k_j so that *bayes_optimal*(k_j, I) holds.

(e) *Connect* the decision node and the created leaf or subtree using a branch labeled with the interval I_t.

(5) Return the tree.

5. Experiments

The theoretical considerations from the previous sections are demonstrated in following by experiments using two artificial datasets and one example of application [18, 19].

5.1. Experiments without Costs. Figure 1 shows a training set (NSEP) consisting of two overlapping classes each taken from a Gaussian distribution in the (x_1, x_2)-plane. The attribute x_1 is irrelevant; that is, the classification does not depend on it and can be performed with x_2 alone. The original CAL5 algorithm with parameters $\alpha = 0.3$ and $S_{\max} = 0.6$ (and no costs) constructs two discriminating straight lines parallel to the x-axis. The region between them indicates that no statistically safe decision is possible here due to the nonseparability and equal probabilities of both classes, as attribute x_1 is irrelevant and therefore cannot deliver any improvement of classification.

The dataset (MULTI) shown in Figure 2 consists of two classes containing two subclasses (clusters) each. All four clusters are derived from Gaussian distributions. CAL5

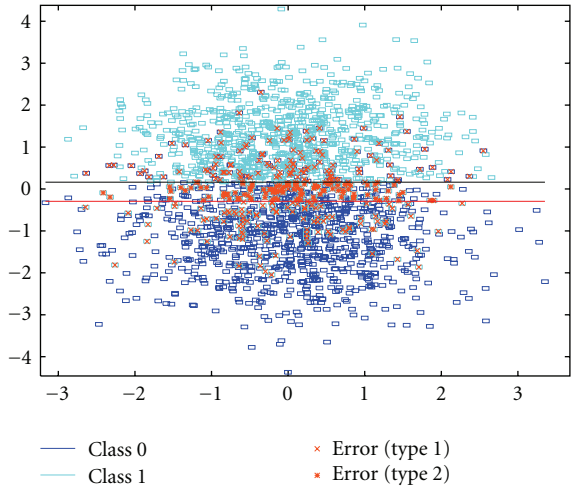

FIGURE 1: Classifier for dataset NSEP (without costs). Errors of type 1 correspond to misclassifications for both classes occurring in intervals for which a class decision was made. Errors of type 2 correspond to misclassifications in intervals where no confident class decision could be made and the final class was assigned based on a majority decision.

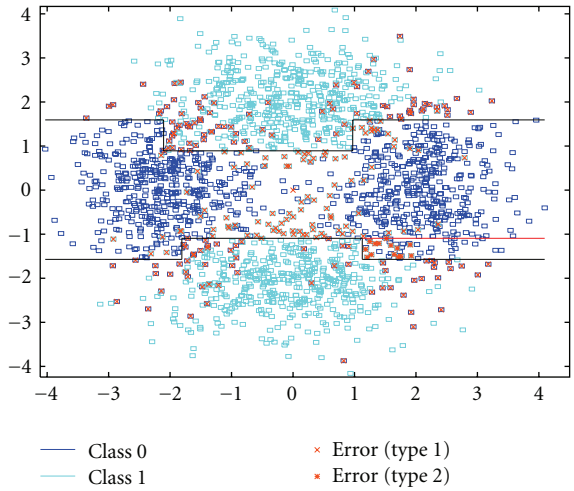

FIGURE 2: Classifier for the MULTI dataset (without costs).

constructs two piecewise linear functions which discriminate between the classes. Subdivisions within the same class built by CAL5 are omitted here.

5.2. Experiments with Object-Dependent Costs. From the NSEP dataset in Section 5.1, a new dataset, NSEP_OC, is constructed using object-dependent costs. These are computed from the cost functions

$$c_0(x_1, x_2) = \frac{2}{1 + \exp(-x_1)} \quad \text{for objects of class 0,} \tag{29}$$

$$c_1(x_1, x_2) = \frac{2}{1 + \exp(+x_1)} \quad \text{for objects of class 1} \tag{30}$$

(Figure 3), both of which are only dependent on the *irrelevant* feature x_1.

Figure 4 shows the training dataset and class discrimination function computed by using the modified CAL5 (CAL5_OC), which includes the cost functions "learned" as mean costs $c_j(I)$ from the object-dependent costs given in the NSEP_OC training set as described in Section 4. The value of the appropriate cost function for a training object is indicated by the size of the rectangles representing the training objects.

Now the discrimination function is piecewise linear: the decision region for class 1 is enlarged in the positive half-space of attribute x_1 and for class 0 in the negative half-space of x_1, respectively, that is, for the high costs of misclassifications. This means that the attribute that was originally irrelevant, x_1, now becomes relevant for class discrimination due to the cost dependence; that is, the decision tree is not optimized in regard to the minimum classification error as it was before, but rather with respect to the minimum cost for misclassification. The costs defining the (local) decision thresholds are "learned" from the individual costs specified by the training objects. Note also that the area in the middle of Figure 1, in which no class decision is possible due to the overlapping of both classes with equal probabilities, is reduced in size in Figure 4.

For the sake of comparison, we ran both the original CAL5 (constructed without costs) and CAL5_OC (constructed with object-dependent costs as described in Section 4) with the NSEP_OC dataset using the (optimized) parameters $\alpha = 0.3, S = 0.6$ in the first and $\alpha = 0.2, S_{\max} = 0.6$ in the second case. For optimization, a systematic search and evaluation with 10-fold cross-validation were conducted. As a result, we arrived at a mean cost of 0.165 in the first case and 0.150 in the second. This means that the mean cost declined by about 9% in the object-dependent, that is, cost-optimized case where CAL5_OC was used. The classification error increased from 16.45% to 16.85%. Note that the classification error *must* increase for classes with different costs of misclassification, here within each class.

For the MULTI dataset, the cost functions

$$c_1(x_1, x_2) = 1, \quad \text{for class 1,} \tag{31}$$

$$c_0(x_1, x_2) = \frac{2}{1 + \exp(-x_1)}, \quad \text{for class 0} \tag{32}$$

are chosen. Figure 5 shows the resulting dataset and discrimination functions that define the decision regions for both classes.

Comparing this with Figure 2 (our cost-independent case), one can see the enlargement of the decision region containing the right cluster of class 0, which consists of objects with high costs for misclassification. This means that there is some overgeneralization of class 0 increasing the error, yet optimizing the mean risk (cost).

5.3. Application in a Real-Life Domain: German Credit Dataset. We conducted experiments with the German credit dataset from the STATLOG project [1]. This dataset comprises 700 examples of the "good costumer" class (class +1)

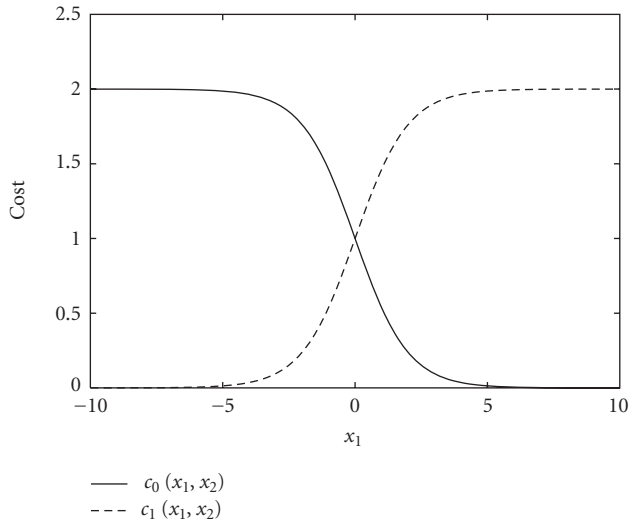

Figure 3: Cost functions for NSEP.

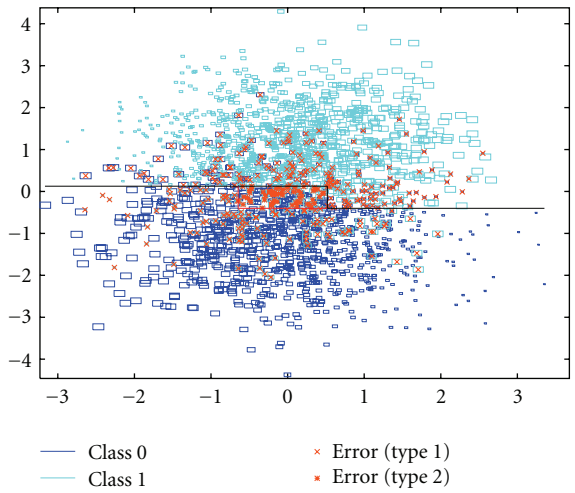

Figure 4: Classifier for NSEP_OC dataset (with object-dependent costs).

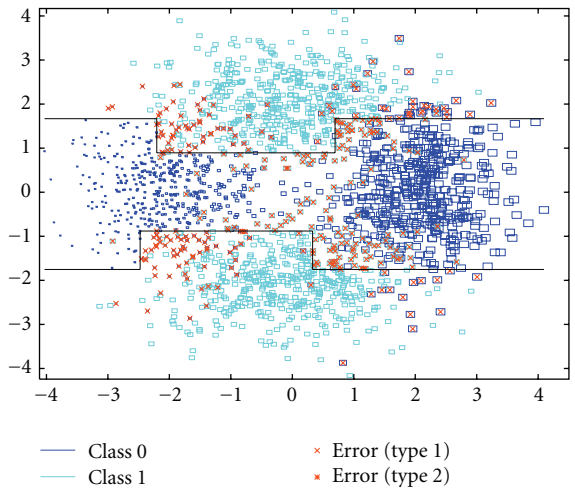

Figure 5: Classifier for MULTI_OC dataset (with object-dependent costs).

and 300 examples of the "bad customer" class (class −1). Each example is described by 24 attributes. The dataset comes with class-dependent costs estimated by experts, that is, a cost-matrix. For the German credit dataset, human experts estimated a relative cost of 5 for not recognizing a bad customer (class −1) as compared to a cost of 1 for not recognizing a good customer (class +1) and cost 0 in the case of correct classification (cost matrix). The original version of CAL5 had already achieved good results for this dataset in the past and learned decision trees with an average of two nodes ([1, page 154]).

In order to evaluate CAL5_OC, we designed the following cost model which features example-dependent costs as opposed to only class-dependent costs. If a good customer is incorrectly classified as a bad customer, we assumed a cost of

$$0.1 * \text{duration} * \frac{\text{amount}}{12}, \tag{33}$$

where the attribute *duration* is the duration of the loan in months, and *amount* is the credit amount. Duration and amount are just two of the 24 attributes used for the sample description in the German credit dataset mentioned above. Here we assumed an effective yearly interest rate of 0.1, that is, 10% for every loan, because the actual interest rates are not given in the dataset itself.

If a bad customer is incorrectly classified as a good costumer, we assumed that 75% of the entire credit amount will be lost (normally a customer will pay back at least part of the loan). When averaging the example-dependent costs for each class, we arrived at a ratio close to that originally given by experts, 1 : 5, which underpins the plausibility of our model. Note that when applying our approach to data from a real bank, we do not have to design a cost function based on the attributes. Instead, the cost values are naturally specified for individual customers. In the case of the German credit dataset, however, we did not have access to these values. In the following, we examine the example-dependent costs defined by the cost function above as the real costs of individual cases.

In our experiments we aimed at a comparison of the results that use example-dependent costs with the results where only class-dependent costs were given. This means that learning with CAL5 was based on a cost matrix, while learning with CAL5_OC was based on the example-dependent costs. However, evaluation was performed in regard to the example-dependent costs, since we viewed them as the real costs of the examples. In the case of CAL_OC, we did not use the given cost matrix estimated by the experts, but rather the new matrix estimated from the example-dependent costs. We constructed this new cost matrix by computing the average of the costs of class +1 and class −1 examples from the training set, which resulted in 6.27 and 29.51, respectively (the credit amounts were normalized so that they lied in the interval $[0, 100]$). The ratio of the misclassification costs in this new cost matrix corresponds roughly with the ratio in the original matrix given by the experts.

We ran CAL5 with the new matrix and CAL5_OC with object-dependent costs using the German credit dataset with the (optimized) parameters $\alpha = 0.25$ and $S_{\max} = 0.93$. As a result, we arrived at a mean cost of 3.34 in the first case and 2.97 in the second. This means that the mean cost declined by about 11% for the object-dependent, that is, optimized case. The classification error increased from 38.2% to 41.5%, which is due to the fact that we optimized the misclassification costs instead of the error rate.

Table 1: German credit dataset: results for different learning algorithms. "Default" is the classifier obtained when always predicting the class that maximizes $B_{k_j}\ p(k_j)$(see Bayes' decision rule).

Algorithm	Estimated mean costs
Default	4.38
SVMLight	3.07
DIPOL_OC	3.67
Resampling DIPOL	3.61
CAL 5_OC	2.97

6. Comparison with Other Algorithms of Classification Learning

We compared the modified decision tree algorithm CAL5_OC with an extended perceptron algorithm (DIPOL, a piecewise linear classifier [1, 26, 31]) modified to include object-dependent costs. We also performed experiments with the support vector machine. In contrast to the experiments with an extended Matlab implementation of the 2-norm SVM described in [14], we used the SVMLight [32] here. This particular implementation of a 1-norm SVM allows the use of individual costs as example weights given as real values. Since these weights are only used for learning, not for evaluating classifiers, we added the required functionality and embedded the learning algorithms into a 10-fold cross-validation. For the experiments, an RBF kernel was utilized and we performed an extensive parameter optimization.

We also took a cost-proportional resampling method into consideration, as described in [13, 14]. The resampling method we developed consists of building a new, cost-free dataset based on the original dataset with costs (see also [33, 34]). This is achieved through a stochastic sampling method that includes an example in the new dataset with a probability proportional to the cost given for that example (as an example-dependent costs or through a cost matrix). This in turn means that the new dataset can be seen as being sampled from the cost-transformed probabilities $p^c(x_i \mid k_j)$ and $p^c(k_j)$ described in the introduction of this paper. This resampling method allows the application of any cost-insensitive learning method to the derived dataset, while still achieving cost-minimization with respect to the original dataset. In the experiments described in [13], we used DIPOL together with the resampling method.

We applied all four approaches to the modified version of the German credit dataset described in Section 5.3. Table 1 shows the costs for the different classifiers estimated by 10-fold cross-validation. It should be noted that CAL5_OC performs better than DIPOL_OC and the resampling method, and slightly better than the SVMLight. Because it is well known that neural networks and kernel methods usually perform better than decision tree algorithms in terms of error, we do not want to draw any general conclusions from this result. However, decision trees allow symbolic rules to be derived and are therefore important in data mining applications. This cannot be achieved in an easy manner with either neural networks or kernel methods. Moreover, it has been shown in the STATLOG project that the performance of CAL5 is comparable to that of C4.5 and even somewhat better in terms of the average rank achieved [26].

The datasets NSEP_OC and MULTI_OC have mainly been designed for demonstrating the qualitative behaviour of the costs-sensitive learning methods. However, for the costs-sensitive versions of DIPOL and the SVM, diagrams corresponding to the ones for CAL5_OC (Figures 4 and 5) can be found in [13, 14]. The specific shape of the class borders differs: CAL5_OC computes axis-parallel borders and DIPOL_OC finds separate hyperplanes in a general position, while the shape of the borders computed by the SVM is determined by the kernel used. The RBF kernel used in the experiments results in a curvy class border and, in the case of NSEP_OC, possibly also disconnected class regions. However, all three approaches locally move the borders towards the classes that contain the examples that are less expensive to be misclassified.

7. Conclusions

In this article we introduced a new, cost-dependent information measure and described how object-dependent costs can be used to learn decision trees (decoders) for cost optimal decisions instead of error minimal decisions. This was demonstrated through the use of decision theory and by defining the cost-minimizing extension CAL5_OC from the CAL5 algorithm, which automatically converts real-valued attributes into discrete-valued ones by constructing intervals. The cost-dependent information measure was used for the selection of the (locally) best next attribute for tree building. It can be used in other algorithms for decision tree learning, but it is of general importance for information theory, modeling in cognitive science and human-computer interaction because the control of behavior by error is replaced by control through costs for false decisions. There are many practical applications in classification learning where minimizing the costs of decisions plays a role, such as in medical diagnosis and financial areas.

Experiments with two artificial datasets and one example of application show the feasibility of our approach and that it is more adequate than a method that uses cost matrices given by experts if cost-dependent training objects are available. Since decision trees constructed with CAL5_OC also separate the classes in the feature space by axis-parallel hyperplanes, it can be used to attain symbolic representations of classes and rules depending on and ordered by their importance. In the

future, it would be interesting to introduce misclassification costs into methods for constructing decision trees with hyperplanes in a general position, using distances to the hyperplanes as a measure of confidence for class decisions [27, 28].

In contrast, for instance, to the cost-sensitive extension of DIPOL, CAL5_OC in its current form is not able to handle misclassification costs that depend not only on the original class of the example but also on the class into which it might be classified incorrectly. This means that each example in the training set must come with a whole vector of cost values corresponding to the different possible classes. We think that, in practice, these cost vectors per example might be difficult to obtain, whereas a single cost value per example (as it is used by CAL5_OC) could be given as the cost that occurred for the respective example in the past.

We also did not consider costs for measuring attributes (e.g., [27, 28]) although it might be possible to incorporate them in the presented framework. We leave this to future work.

Appendix

The Decision Rule for a Class in a Completed Interval I Including the Possibility of Rejection of the Class Decision

In Section 4 we used a decision rule for a class in a completed interval I (labeling a leaf):

Decide for class k_j in interval I if

(a) $c_j(I)p(k_j \mid I) > c_r(I)p(k_r \mid I)$ for all $r \neq j$(Bayes' decision rule) and simultaneously

(b)

$$p\left(k_j \mid I\right) \geq \frac{\sum_{r=1}^{n} c_r(I)p(k_r \mid I) - c_0(I)}{c_j(I)} =_{\text{Def.}} S_j(I), \tag{A.1}$$

where $c_0(I)$ is the cost of the rejection of a class decision, that is, for further branching out in general. Note that the value of $c_0(I)$ is generally unknown prior to learning; it will be approximately, yet implicitly, optimized by choosing the value for $S_{\max}$ (see Section 4 and below) which leads to a minimization of the expected costs for misclassification.

For the proof of the necessity of also using the second rule, we start with the formulation of a condition for the rejection of a class decision. This is given by the following.

Decide for the rejection of a class decision if for all classes k_s, $s = 1,\ldots,n$ *occurring in I, it holds that*

$$\sum_{r=1,\ldots,n,r\neq s} c_r(I)p(k_r \mid I) > c_0(I). \tag{A.2}$$

The sum in the LHS of the inequation is the expected value of the costs of the misclassification that would arise during tree application for all other classes $k_r \neq k_s$ if k_s is ultimately decided on in the process of tree construction, that is, if I is replaced by the label k_s. This expected value is estimated from the set of training objects occurring in I.

The inequation can be transformed to

$$\sum_{r=1}^{n} c_r(I)p(k_r \mid I) - c_s(I)p(k_s \mid I) > c_0(I) \tag{A.3}$$

or its equivalent

$$p(k_s \mid I) < \frac{\sum_{r=1}^{n} c_r(I)p(k_r \mid I) - c_0(I)}{c_s(I)} =_{\text{Def.}} S_s(I), \tag{A.4}$$

which means that it must hold for all $s(s = 1, \ldots, n)$ as a condition for the rejection of a class decision in I.

The value of $\sum_{r=1}^{n} c_r(I)p(k_r \mid I)$ corresponds with the mean cost B_I in interval I; that is, it corresponds with the value defined in Section 2, where x_i is one of the values for attribute x. In general, the value of the rejection cost $c_0(I)$ is not known in advance. Considering the class with the minimum costs $c_{\min}(I)$ in I as a special case, we therefore get

$$\frac{B_I - c_0(I)}{c_{\min}(I)} = S_{\max}(I) \tag{A.5}$$

or its equivalent

$$B_I - c_0(I) = c_{\min}(I)S_{\max}(I). \tag{A.6}$$

The value of $S_{\max}(I)$ is replaced by $S_{\max}$, that is, by a general parameter, which is independent of I. The parameter value $S_{\max}$ has to be optimized or specified by the user. $S_{\max}$ is a factor contained in all values S_j, which can be enlarged or reduced without changing the relations between them. Therefore, $S_{\max}$ also controls the complexity of the resulting tree.

The rule for a decision for a single class in I, part (b) stated above, is a negation of the rejection rule:

Do not reject a class decision if there is at least one class k_j with

$$\begin{aligned} p\left(k_j \mid I\right) &\geq \frac{\sum_{r=1}^{n} c_r(I)p(k_r \mid I) - c_0(I)}{c_j(I)} \\ &= S_j(I) = \frac{c_{\min}(I)S_{\max}}{c_j(I)}. \end{aligned} \tag{A.7}$$

If condition (a) also holds for class k_j, that is, if the value of $c_j(I)p(k_j \mid I)$ is the maximum value, then a decision can be made for k_j at this leaf.

References

[1] D. Michie, D. H. Spiegelhalter, and C. C. Taylor, *Machine Learning, Neural and Statistical Classification*, Series in Artificial Intelligence, Ellis Horwood, 1994.

[2] L. Saitta, Ed., *Machine Learning—A Technological Roadmap*, University of Amsterdam, Amsterdam, The Netherlands, 2000.

[3] P. D. Turney, "Types of cost in inductive concept learning," in *Proceedings of the Workshop on Cost-Sensitive Learning at the 17th International Conference on Machine Learning*, Stanford University, Stanford, Calif, USA, 2000.

[4] Ch. X. Ling and V. S. Sheng, "Cost–sensitive learning and the class imbalance problem," in *Encyclopedia of Machine Learning*, C. Sammut, Ed., Springer, 2008.

[5] Ch. X. Ling, Q. Yang, J. Wang, and S. Zhang, "Decision trees with minimal costs," in *Proceedings of International Conference on Machine Learning (ICML '04)*, 2004.

[6] C. Drummond and R. Holte, "Exploiting the cost (in)sensitivity of decision tree splitting criteria," in *Proceedings of the 17th International Conference on Machine Learning (ICML '00)*, pp. 239–246, Stanford, Calif, USA, June 2000.

[7] D. D. Margineantu and T. G. Dietterich, "Bootstrap methods for the cost-sensitive evaluation of classifiers," in *Proceedings of the 17th International Conference on Machine Learning*, pp. 583–590, Morgan Kaufmann, San Francisco, Calif, USA, 2000.

[8] C. Elkan, "The foundations of cost-sensitive learning," in *Proceedings of the 17th International Conference on Artificial Intelligence (IJCAI '01)*, B. Nebel, Ed., pp. 973–978, Morgan Kaufmann, San Francisco, Calif, USA, August 2001.

[9] M. Kukar and I. Kononenko, "Cost-sensitive learning with neural networks," in *Proceedings of the 13th European Conference on Artificial Intelligence (ECAI '98)*, H. Prade, Ed., pp. 445–449, John Wiley & Sons, Chichester, UK, 1998.

[10] Y. Lin, Y. Lee, and G. Wahba, "Support vector machines for classification in nonstandard situations," *Machine Learning*, vol. 46, no. 1–3, pp. 191–202, 2002.

[11] A. Lenarcik and Z. Piasta, "Rough classifiers sensitive to costs varying from object to object," in *Proceedings of the 1st International Conference on Rough Sets and Current Trends in Computing (RSCTC '98)*, L. Polkowski and A. Skowron, Eds., vol. 1424 of *Lecture Notes in Computer Science*, pp. 222–230, Springer, Berlin, Germany, June 1998.

[12] P. Geibel and F. Wysotzki, "Perceptron based learning with example dependent and noisy costs," in *Proceedings of the 20th International Conference on Machine Learning (ICML '03)*, Washington, DC, USA, 2003.

[13] P. Geibel and F. Wysotzki, "Learning perceptrons and piecewise linear classifiers sensitive to example dependent costs," *Applied Intelligence*, vol. 21, no. 1, pp. 45–56, 2004.

[14] P. Geibel, U. Brefeld, and F. Wysotzki, "Perceptron and SVM learning with generalized cost models," *Intelligent Data Analysis*, vol. 8, no. 5, pp. 439–455, 2004.

[15] R. Klinkenberg and S. Rüping, "Concept drift and the importance of examples," in *Text Mining—Theoretical Aspects and Applications*, J. Franke, G. Nakhaeizadeh, and I. Renz, Eds., Springer, 2003.

[16] P. Geibel, *Risk-Sensitive Approaches for Reinforcement Learning*, Shaker, 2006.

[17] L. Li, V. Bulitko, and R. Greiner, "Batch reinforcement learning with state importance," in *Proceedings of the European Conference on Machine Learning (ECML '04)*, pp. 566–568, Springer, 2004.

[18] A. Bendadi, O. Benn, P. Geibel, M. Hudik, T. Knebel, and F. Wysotzki, "Lernen von Entscheidungsbäumen bei objektabhängigen Fehlklassifikationskosten," Technischer Bericht 2004-18, Technischen Universität Berlin, Berlin, Germany, 2005.

[19] F. Wysotzki and P. Geibel, "Computer-supported decision making with object dependent costs for misclassifications," in *Human Interaction with Machines, Proceedings of the 6th International Workshop Held at the Shanghai Jiao Tong University*, G. Hommel and S. Huanye, Eds., pp. 129–139, Springer, March 2005.

[20] S. Unger and F. Wysotzki, *Lernfähige Klassifizierungssysteme (Classifier Systems Which Are Able to Learn)*, Akademie-Verlag, Berlin, Germany, 1981.

[21] W. Müller and F. Wysotzki, "Automatic construction of decision trees for classification," *Annals of Operations Research*, vol. 52, no. 4, pp. 231–247, 1994.

[22] W. Müller and F. Wysotzki, "The decision-tree algorithm CAL5 based on a statistical approach to its splitting algorithm," in *Machine Learning and Statistics*, G. Nakhaeizadeh and C. C. Taylor, Eds., pp. 45–65, John Wiley & Sons, New York, NY, USA, 1997.

[23] C. E. Shannon and W. Weaver, *The Mathematical Theory of Communication*, University of Illinois Press, Urbana, Ill, USA, 1949.

[24] A. Ethem, *Machine Learning*, MIT Press, 2004.

[25] J. Körner and F. Wysotzki, "Die Rolle der Übergeneralisierung in der Neurosenbildung," *Forum der Psychoanalyse*, vol. 22, no. 4, pp. 321–341, 2006.

[26] F. Wysotzki, W. Müller, and B. Schulmeister, "Automatic construction of decision trees and neural nets for classification using statistical considerations," in *Learning, Networks and Statistics*, G. Della Riccia, H.-J. Lenz, and R. Kruse, Eds., CISM Courses and Lectures no. 382, pp. 121–134, Springer, New York, NY, USA, 1997.

[27] N. Tóth and B. Pataki, "On classification confidence and ranking using decision trees," in *Proceedings of the 11th International Conference on Intelligent Engineering Systems (INES '07)*, pp. 133–138, Budapest, Hungary, June-July 2007.

[28] N. Tóth and B. Pataki, "Classification confidence weighted majority voting using decision tree classifiers," *International Journal of Intelligent Computing and Cybernetics*, vol. 1, no. 2, pp. 169–192, 2008.

[29] J. R. Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann, San Mateo, Calif, USA, 1993.

[30] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and Regression Trees*, Wadsworth and Brooks, Monterey, Calif, USA, 1984.

[31] B. Schulmeister and F. Wysotzki, "DIPOL-a hybrid piecewise linear classifier," in *Machine Learning and Statistics: The Interface*, R. Nakeiazadeh and C. C. Taylor, Eds., pp. 133–151, John Wiley & Sons.

[32] T. Joachims, "Making large-scale SVM learning practical," in *Advances in Kernel Methods—Support Vector Learning*, B. Schölkopf, C. Burges, and A. Smola, Eds., MIT-Press, 1999.

[33] P. K. Chan and S. J. Stolfo, "Toward scalable learning with non-uniform class and cost distributions: a case study in credit card fraud detection," in *Proceedings of the Knowledge Discovery and Data Mining (KDD '98)*, pp. 164–168, New York, NY, USA, 1998.

[34] B. Zadrozny and C. Elkan, "Learning and making decisions when costs and probabilities are both unknown," in *Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '01)*, F. Provost and R. Srikant, Eds., pp. 204–214, ACM Press, New York, NY, USA, August 2001.