# Global Data Analysis and the Fragmentation Problem in Decision Tree Induction

Ricardo Vilalta, Gunnar Blix, and Larry Rendell

Beckman Institute and Computer Science Department
University of Illinois at Urbana-Champaign
405 North Mathews Avenue, Urbana, IL 61801 USA
phone: (217) 244-1620
fax: (217) 244-8371
email: vilalta@cs.uiuc.edu

**Abstract.** We investigate an inherent limitation of top-down decision tree induction in which the continuous partitioning of the instance space progressively lessens the statistical support of every partial (i.e. disjunctive) hypothesis, known as the *fragmentation problem*. We show, both theoretically and empirically, how the fragmentation problem adversely affects predictive accuracy as variation $\nabla$ (a measure of concept difficulty) increases. Applying feature-construction techniques at every tree node, which we implement on a decision tree inducer *DALI*, is proved to only partially solve the fragmentation problem. Our study illustrates how a more robust solution must also assess the value of each partial hypothesis by recurring to all available training data, an approach we name *global data analysis*, which decision tree induction alone is unable to accomplish. The value of global data analysis is evaluated by comparing modified versions of *C4.5*rules with *C4.5*trees and *DALI*, on both artificial and real-world domains. Empirical results suggest the importance of combining both feature construction and global data analysis to solve the fragmentation problem.

## 1 Introduction

In this study, we investigate the internal mechanism of top-down decision tree inducers [14, 1]. We focus on the fragmentation problem: a limitation of the divide-and-conquer strategy in which the continuous partitioning of the training set at every tree node reduces the number of examples (i.e. the statistical support) at lower-level nodes. One noticeable effect of this problem is the replication of subtrees along the output tree [12, 9], also known as the *replication problem*. The fragmentation problem has been attacked in different ways: by constructing compound features at every tree node [12, 18]; by reducing the number of possible partitions [5, 16]; and by using alternative concept representations, e.g., sets of rules [15], decision graphs [6, 11], SE-Trees [22], decision lists [12, 21]. Nonetheless, no clear solution has emerged.

Our analysis of the causes and effects of the fragmentation problem elucidates relevant issues: the fragmentation problem is not coined to decision tree

induction alone, but might affect other inductive learning models; replication and fragmentation are not separate problems, but rather the former is simply an effect of the latter. We use concept variation, $\nabla$ (a measure of concept difficulty [20, 13]), to prove that as $\nabla$ increases, the fragmentation problem is further aggravated, partly explaining the inadequacy of decision tree induction when applied to *difficult* domains.

We test a new decision-tree inducer, *DALI*, to show that constructing new features at every tree node mitigates the fragmentation problem, but does not completely eliminate it. For a more robust solution, our study reveals the importance of analyzing all training data when assessing the value of every induced hypothesis, an approach we name *global data analysis*. Decision tree induction does not analyze data in this manner, neither alone nor when augmented with feature construction. Our experiments compare *C4.5*rules, *C4.5*trees, and *DALI*, empirically evaluating the importance of global data analysis by isolating this component in *C4.5*rules. Our results suggest the importance of combining both feature construction and global data analysis to solve the fragmentation problem.

This paper is organized as follows. Section 2 provides an overview of decision tree induction; Sect. 3 defines the fragmentation problem; Sect. 4 explains the scenarios in which the fragmentation problem is critical, and details on the importance of global data analysis during learning; Sect. 5 shows experimental results. Lastly, Sect. 6 gives a summary and conclusions.

## 2 Preliminaries

For simplicity, we focus on domains where each example $X$ is described by $n$ boolean features (i.e., attributes, variables), $x_1, x_2, \cdots, x_n$, and where an underlying target concept $C : \{0,1\}^n \mapsto \{-,+\}$ classifies the space of all $2^n$ examples, also referred to as the *instance space*, into 2 classes. A learning mechanism (i.e., inducer) attempts to discover $C$ by analyzing the information given by a training set $S : \{(X_i, c_i)\}_{i=1}^m$, where $c_i$ is the class assigned by $C$ to $X_i$, i.e., $C(X_i) = c_i$. The result of the analysis is a hypothesis/classifier $H$ approximating $C$. Our main interest is in the ability of $H$ to correctly *predict* the class of examples outside $S$. We look for hypotheses not only consistent with $S$, but that *generalize* beyond that set.

A decision tree inducer uses a divide-and-conquer strategy for learning. Proceeding top-down, the root of the tree is formed by selecting a function $f : \{0,1\}^n \mapsto \{0,1\}$ that splits the training set into mutually exclusive subsets $S_0, S_1$, such that $S_0 = \{X \in S \mid f(X) = 0\}$, $S_1 = \{X \in S \mid f(X) = 1\}$, $S = S_0 \cup S_1$, and $S_0 \cap S_1 = \emptyset$. Commonly $f$ is a single feature – selected via some impurity measure, e.g., entropy, gini, Laplace, $\chi^2$ –, which yields axis-parallel partitions over the instance space, but other combinations are possible [2, 8]. The same methodology is recursively applied on $S_0$ and $S_1$ to construct the left and right subtrees respectively. A subset $S'$ represents a leaf if all examples in $S'$ belong to the same class, or if $|S'| < \beta$, where $\beta$ is user defined; the majority class in $S'$ is associated with that leaf. An example $X$ is classified,

starting from the root of the tree, by following the branch that matches the output of every splitting function (i.e., by iteratively following the left branch if $f(X) = 0$, or the right branch if $f(X) = 1$). At the end of the path, the class attached to that leaf is assigned to $X$.

## 3 The Fragmentation Problem

Under a DNF representation, a target concept $C$ is expressed as the disjunction of several subconcepts, such that $C = C_1 + C_2 + \cdots + C_l$. A hypothesis $H$ approximating $C$ can be expressed as a set of disjunctive hypotheses $H_1, H_2, \cdots, H_l$, where $H_i$ approximates $C_i$. The set of examples covered by hypothesis $H_i$ on training set $S$, $\text{COV}(H_i) = \{X \in S \mid H_i(X) = +\}$, is referred to as the source of support or evidential credibility for $H_i$ [23].

A decision tree inducer adopts a DNF concept representation: each branch from the root of the tree to a positive leaf is equivalent to a disjunctive hypothesis $H_i$. The final set of disjunctive hypotheses must be mutually exclusive, i.e., $\text{COV}(H_1) \cap \text{COV}(H_2) \cap \cdots \cap \text{COV}(H_l) = \emptyset$. The method to find every $H_i$ carries out a continuous partition-refinement over the instance space; every tree branch is grown until a terminal node or leaf delineates a single-class region. A limitation inherent to this approach is that, while searching for a disjunctive hypothesis $H_i$, each splitting of the training data may separate or pull apart examples in support of a different hypothesis $H_j$ – due to the irrelevancy of the splitting function to $H_j$. This situation not only requires that several approximations $H_j'$, $H_j''$,$H_j'''$ etc., be found on dispersed regions of the instance space, giving rise to replicated subtrees along the output tree, but also reduces the support or evidential credibility of each individual hypothesis, eventually complicating its identification. This problem is known as the *fragmentation problem.*

The fragmentation problem stems from two main causes:

1. The requirement that the coverage of disjunctive hypotheses be mutually exclusive precludes the representation of any $H_i$ and $H_j$ such that $\text{COV}(H_i) \cap \text{COV}(H_j) \neq \emptyset$. The examples in $\text{COV}(H_i) \cap \text{COV}(H_j)$ are directed towards $H_i$ or $H_j$, but not both. This is illustrated in the following example. Assume a 3-dimensional boolean space where each point represents an example $X = (x_1, x_2, x_3)$, with target function $C = x_1x_2 + x_1x_3$, as shown in Fig. 1a. Concept $C$ can be decomposed into two subconcepts: $C_1 = x_1x_2$, with examples $(1,1,0)$ and $(1,1,1)$, and $C_2 = x_1x_3$, with $(1,0,1)$ and $(1,1,1)$. Let $H_1$ and $H_2$ be the disjunctive hypotheses approximating $C_1$ and $C_2$ respectively. With all examples available, $\beta = 1$, and single features as splitting functions, two possible decision trees are depicted in Figs. 1b and 1c. In the tree for Fig. 1b, splitting on feature $x_2$ directs two positive examples to the right branch in support of $H_1$, but only one positive example (out of two) to the left branch in support of $H_2$. As a consequence, $H_1 = C_1$ but $H_2 \neq C_2$, since the irrelevant condition $\bar{x_2}$ is incorporated: $H_2 = x_1\bar{x_2}x_3$. This was caused because $\text{COV}(C_1) \cap \text{COV}(C_2) = \{(1,1,1)\}$, which could only be covered

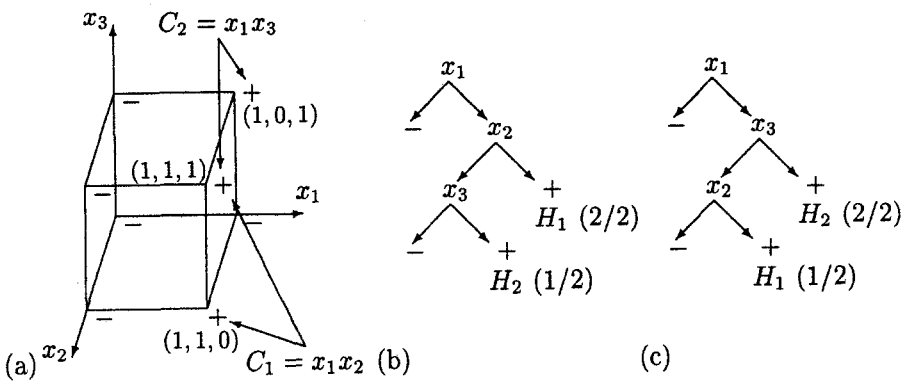


**Fig. 1.** (a) A 3-dimensional boolean space for target concept $C = C_1 + C_2$, where $C_1 = x_1x_2$ and $C_2 = x_1x_3$. (b) and (c) Two decision trees for (a) where the fragmentation problem reduces the support of either $H_1$ or $H_2$, as shown by the fraction of examples that belong to $C_1$ and $C_2$ arriving at each positive leaf.

by $H_1$ or $H_2$. The same phenomenon occurs in the tree on Fig. 1c, except here the loss of support occurs to $H_1$.

2. Each partition over the instance space is too coarse, such that many steps are required to delimit a single-class region. The search for a disjunctive hypothesis $H_i$ inevitably results in the fragmentation of a different disjunctive hypothesis $H_j$. Consider the tree in Fig. 2a for boolean concept $C = x_1x_2 + x_3x_4$, where $C_1 = x_1x_2$ and $C_2 = x_3x_4$. Assume all possible examples available, $\beta = 1$, $H_1$ the approximation to $C_1$, and $H_2{}'$ and $H_2{}''$ the approximations to $C_2$. Splitting on feature $x_1$ separates the examples in $C_2$, directing two positive examples (out of four) to the left branch in support of $H_2{}'$, and two examples to the right branch. Splitting on feature $x_2$ reduces the support of $H_2{}''$ to only one example. Since $C_2$ is represented by $H_2{}'$ and $H_2{}''$, the final tree replicates subtrees. This replication effect originates from the fragmentation of $H_2$ (into $H_2{}'$ and $H_2{}''$) at the root of the tree.

One approach to combat the fragmentation problem is to conjoin several features at every tree node, which results in more refined partitions over the instance space. As shown in Fig. 2b, using the conjunction of single features as splitting functions eliminates the replication of the subtree approximating $C_2$. Nevertheless, $H_2$ continues experiencing loss of support, since $\mathrm{COV}(H_1) \cap \mathrm{COV}(H_2) = \{(1,1,1,1)\} \neq \emptyset$. Hence, using multiple-feature tests at every tree node can reduce the number of partition-steps required to delimit single-class regions (cause 2), but cannot avoid pulling apart examples lying in the intersection of several partial subconcepts (cause 1).

The fragmentation problem is not exclusive to decision tree inducers but to any learning mechanism that progressively lessens the evidential credibility of its

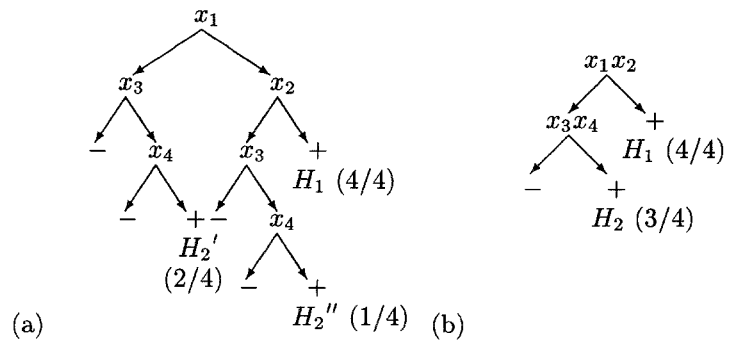


**Fig. 2.** (a) A decision tree for $C = C_1 + C_2$, where $C_1 = x_1x_2$ and $C_2 = x_3x_4$. Examples in $C_2$ are separated into two regions after splitting on feature $x_1$. (b) A decision tree for (a) with multiple-feature tests. The replication of subtrees is eliminated, but $H_2$ experiences loss of support when splitting on $x_1x_2$.

induced hypotheses. Consider the separate-and-conquer strategy common to the construction of rule-based systems [10, 4, 28]. In this case, an iterative process starts by selecting a positive example or *seed* on the training data; this example is generalized to produce the next disjunctive hypothesis $H_i$. The set of examples covered by $H_i$, COV($H_i$), is *removed* before another seed is selected, potentially weakening the support of other disjunctive hypotheses. A similar effect occurs in the mechanism for building decision lists [12, 21].

## 4 Detrimental Effects and Global Data Analysis

One may argue against the significance of the fragmentation problem based on the success of decision tree inducers on many real-world applications. As explained shortly (and demonstrated in Sect. 5), a detrimental effect is evident only among domains with high variation $\nabla$.

Decision tree inducers, as well as many other inductive mechanisms, adopt a *similarity-based bias*, which assumes any pair of examples $X_i$,$X_j$ lying close to each other in the instance space (i.e., sharing many similar feature-values) generally belong to the same class, i.e., $C(X_i) = C(X_j)$. This bias is adequate when a concept is characterized by few disjunctive subconcepts, each subconcept covering many examples, because proximity in the instance space correlates to class similarity [19]; these domains we denote as *simple*. By contrast, domains with instance spaces populated by many dispersed regions, each disjunctive subconcept covering few examples, violate this assumption, and thus become inadequate; these domains we denote as *difficult*.

The degree of difficulty of a concept can be known through concept variation $\nabla$ [20, 13], which provides an estimate of the probability that any two neighbor

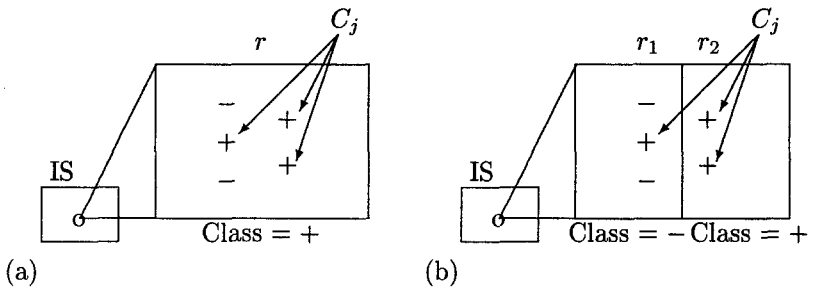


**Fig. 3.** (a) A region $r$ of an instance space (IS) where the support for the prediction of class + is unstable. (b) After partitioning $r$ into $r_1$ and $r_2$, $C_j$ is fragmented. The majority of negative examples in $r_1$ changes the prediction to class −.

examples differ in class value, roughly measuring the amount of irregularity in the distribution of examples along the instance space. $\nabla$ is defined as follows. Let $X_1, X_2, \cdots, X_n$ be the $n$ closest neighbors – at Hamming distance one – of an example $X$ in an $n$-dimensional boolean space. The degree of class *dissimilarity* of the neighborhood around $X$ can be estimated simply as

$$\sigma(X) = \sum_{j=1}^{n} \text{diff}(C(X), C(X_j)) \ , \tag{1}$$

where $\text{diff}(C(X, X_i)) = 1$ if $C(X) \neq C(X_i)$ and 0 otherwise. A normalization factor $\overline{\sigma}(X) = \frac{\sigma(X)}{n}$ gives a value in $[0, 1]$. Concept variation is defined as the average of this factor when applied to every example in the instance space:

$$\nabla = \frac{1}{2^n} \times \sum_{i=1}^{2^n} \overline{\sigma}(X_i) \ \in [0, 1] \ . \tag{2}$$

The effect of the fragmentation problem relates to $\nabla$ (i.e. to concept variation) in the following way. The terminal node or leaf of a tree branch, corresponding to a disjunctive hypothesis $H_i$, classifies a region $r$ of examples according to a majority-class vote on the training examples in $r$. Let $\theta$ be defined as the difference between the number of training positive and negative examples in $r$; then for any $X \in r$,

$$C(X) = \begin{cases} + & \text{if } \theta \geq 0 \\ - & \text{otherwise} \end{cases} \tag{3}$$

The fragmentation problem is irrelevant over domains with low $\nabla$ (i.e. over simple domains), because, in the presence of disjunctive subconcepts covering many examples of similar class value, each hypothesis (i.e. tree branch) delimits a region of examples $r$ for which $\theta >> 0$ (i.e. for which $\theta$ is stable). But when dissimilarity in the vicinity of any example is high, as is characteristic in domains with high $\nabla$ (i.e., in difficult domains), then separating the few examples covered by each disjunctive subconcept reduces the support of its hypothesis(es). If $\theta \sim 0$

**Algorithm 1:** Learning with Global Data Analysis
**Input:** Given unknown target concept $C = C_1 + C_2 + \cdots + C_l$, Training Set $S$, Metric $M$
**Output:** Final Hypothesis $H_f$
*GDA*-MECHANISM($S$)
(1) Let $H_f = \emptyset$
(2) **foreach** $i = 1 \cdots l$
(3) Generate hypotheses approximating subconcept $C_i$
(4) Evaluate each hypothesis by using all examples in $S$
(5) Select best approximation $H_i$ acording to $M$
(6) Let $H_f = H_f + H_i$
(7) **end for**
(8) Refine/Prune $H_f$ by using all examples in $S$
(9) **return** $H_f$

**Fig. 4.** General learning mechanism with global data analysis.

(i.e. $\theta$ is unstable), then removing examples from $r$ may cause $\theta$ shift sign, thereby causing the misclassification of all $X \in r$. In addition, observe that if $\nabla > 0.5$, a similarity-based bias becomes totally inadequate, even without the presence of the fragmentation problem, because, on average, more than 50% of the vicinity of any example $X$ would differ in class value with $X$.

To illustrate these ideas, Fig. 3a shows a region $r$ of an instance space where $\theta \geq 0$, such that for any $X \in r$, class + is predicted. If the set of training positive examples in $r$ belong to a subconcept $C_j$ (and possibly other subconcepts as well), then finding an approximation to a subconcept $C_i$ before $C_j$ may lead to a partitioning of $r$ into $r_1$ and $r_2$, as shown in Fig. 3b. The positive training example representing $C_j$ in $r_1$ may be mistakenly perceived as a noise signal. The instability of $\theta$ in $r$ causes $\theta < 0$ in $r_1$, forcing a change of classification to every example $X \in r_1$; this is unlikely to occur if $\nabla$ is low because all examples in a small region are expected to belong to the same class.

We claim an important step to solve the fragmentation problem consists of building/refining each partial hypothesis independently, by assessing its value against all training examples together, in this way avoiding misclassification of regions of examples for which little support is found. Figure 4 depicts a general learning mechanism that incorporates a *global data analysis*. The main idea is to better estimate the value of each partial hypothesis by avoiding the effects of previously induced hypotheses. Under this learning framework, an approximation $H_i$ to a subconcept $C_i$ is built under the support of all available data (lines 4-5, Fig. 4). The final hypothesis $H_f$ may also be refined (e.g., pruned) in this way (line 8, Fig. 4).

In contrast, the search for disjunctive hypotheses in decision tree induction is not global but local: often a hypothesis is supported by only a fraction of the examples of the subconcept being approximated. This holds irrespective of the modifications exerted on the learning mechanism (e.g., splitting function,

pruning mechanism, stopping criteria, etc.), because such search is limited by the learning strategy. Henceforth, we identify two major operations during the development of learning systems: 1) the search for partial hypotheses, and 2) the refinement of the final hypothesis comprising all best partial hypotheses. Both steps can be attained through a global data analysis, but the inherent mechanism of decision tree induction *omits* this operation. In the next section we evaluate the importance of a global data analysis over the refinement of the final hypothesis (step 2); better results are expected if the same methodology is carried on over the construction of all partial hypotheses (step 1).

# 5 Experiments

## 5.1 The Learning Systems Used for Testing

We use *C4.5*trees [16] to represent a decision tree inducer where each splitting function tests on a single feature. The importance of global data analysis is underlined in modified versions of *C4.5*rules [15, 16], as explained in Sect. 5.3. For a decision tree inducer with multiple-feature tests, we developed a new version of the *LFC* system [18]; the new version is called *DALI* [26] (Dynamic Adaptive Lookahead Induction). In both *DALI* and *LFC*, a splitting function is defined as the conjunction of several boolean features (see Fig. 2b), which allows for more refined partitions over the instance space. Unlike *LFC*, *DALI* obviates user-defined parameters (e.g., lookahead depth and beam width), with a faster response time, and similar performance in terms of predictive accuracy. We now briefly compare *DALI* and *LFC*, but the reader can safely skip to the next subsection if uninterested in such differences.

Figure 5 outlines *DALI*'s search mechanism. At each tree node, both *DALI* and *LFC* conduct a beam search over the space of all boolean-feature conjuncts, or *monomials*. In *LFC*, the search space is limited by user-defined width and depth; the search continues until the maximum depth $d$ is attained, at which point the best monomial – of any size in $[1-d]$ – is returned as the next splitting function. By contrast, *DALI* extends the search depth until no more monomials can be generated, and selects the beam width dynamically. *DALI* mainly differs from *LFC* on two steps:

1. A systematic search to avoid redundant combinations [22, 27], Lines 3-4, Fig. 5. Each monomial $F_i$ conjoins several boolean features (or their complements), e.g., $F_i = x_1\bar{x}_3x_5$. Because conjunction is commutative, the search space is defined by avoiding any state $F_j$ that is identical to a state $F_i$ except for the order in which features appear, e.g., $F_j = \bar{x}_3x_5x_1$.
2. A global-pruning technique [27, 17], Line 5, Fig. 5. Define $F_{\text{best}}$ as the best explored monomial according to some impurity measure $H$ (e.g., entropy), such that, for all currently explored monomials $F_i$, $H(F_{\text{best}}) < H(F_i)$. As long as $H$ is monotonic, a monomial $F_i$ can be eliminated if the best value $F_i$ can ever attain along its search path – according to $H$ – is worse than $F_{\text{best}}$.

**Algorithm 2:** Search Mechanism in *DALI*
**Input:** A list of literals (i.e., boolean features and their complements) $L_{\text{bool}} \leftarrow \{x_1, \bar{x}_1, x_2, \bar{x}_2, \cdots x_n, \bar{x}_n\}$
**Output:** Best monomial $F_{\text{best}}$
*DALI*-SEARCH($L_{\text{bool}}$)
(1) $L_{\text{beam}} \leftarrow$ best literals in $L_{\text{bool}}$ according to entropy
(2) **while** (**true** )
(3) $L_{\text{new}} \leftarrow$ Systematically form the conjunction
(4) of every $F_i \in L_{\text{beam}}$ with every $F_j \in L_{\text{bool}}$
(5) Apply global-pruning into $L_{\text{new}}$
(6) **if** $L_{\text{new}} = \emptyset$
(7) **return** best monomial $F_{\text{best}}$
(8) $L_{\text{beam}} \leftarrow$ best combinations in $L_{\text{new}}$ according to entropy
(9) **end while**

**Fig. 5.** *DALI*'s search mechanism at every tree node. The best constructed feature (i.e., best monomial), is used as the next splitting function.

The combination of systematic search and global pruning makes the search space sufficiently manageable so that a limitation on depth or breadth of search is no longer necessary. The ease of use of *DALI* favors this system over *LFC* for our experimental purposes.

### 5.2 Methodology

Since variation $\nabla$ can be computed only when the target concept is known, our experiments mainly focus on artificial boolean concepts (defined on both 9 and 12-features; see Appendix A). The concepts include DNF formulae, CNF formulae, Multiplexor (MUX), Majority (MAJ), and Parity (PAR), covering a range of varying $\nabla$.

Learning curves[1], not presented here for space considerations, show that greater differences in accuracy occur when small samples are used for training. Our results reflect the largest effects found at 20% training-set size for 9-feature concepts and 10% training-set size for 12-feature concepts. Each reported value is the average over 50 runs; predictive accuracy is computed as the percentage of correct classifications for all examples outside the training set. Experiments on real-world domains estimate predictive accuracy by using stratified 10-fold cross-validation [7], averaged over five repetitions. Since *DALI* is limited to boolean domains, we performed an initial discretization step on all numeric (following [3]) and nominal features (constructing a boolean feature for each nominal value). All systems were set to default parameters. Significant differences are computed using a two-sided $t$-test. Runs were performed on a SPARCstation 10/31.

[1] Graphs for all learning curves are accessible upon request to the authors.

### 5.3 The Value of Global Data Analysis

To measure the gains obtained when global data analysis is used to tackle the fragmentation problem, we first compared modified versions of *C4.5*rules with *C4.5*trees as explained next. The mechanism for *C4.5*rules (see [16] for details) can be summarized in three steps:

1. Given a decision tree $T$, form a rule $R$ from every branch in $T$ that starts at the root node and ends on a leaf node; $R$ is an implication: if $\text{cond}_1$ and $\text{cond}_2$ and $\cdots$ and $\text{cond}_{d-1} \rightarrow$ c, where $\text{cond}_i$ is the feature-value (i.e. splitting-function value) encountered on every node along a branch of length $d$, and $c$ the class assigned to the leaf node.
2. Eliminate all irrelevant conditions from every rule $R$ in step 1. Let $R'$ equals $R$ except for condition $\text{cond}_i$ being removed. Based on the information given by all training data, $R$ and $R'$ are both globally evaluated – and only one retained – according to a pessimistic estimation of their corresponding error rates.
3. Apply the minimum description principle, according to a particular bit-encoding scheme, to remove rules from the rule set in step 2.

While each individual rule is originally obtained from a previously constructed decision tree, Step 2 refines the final set of rules through a global data analysis: each rule is analyzed independently and modified according to its credibility on all training data. This differs from step 3 where rules are assessed in terms of description lengths. To isolate each learning component, we defined three system versions: *C4.5*rules-*Std*, comprising steps 1, 2, and 3 (i.e., Standard); *C4.5*rules-*GDA*, comprising steps 1 and 2 (i.e., isolating the Global Data Analysis component); and *C4.5*rules-*MDL*, comprising steps 1 and 3 (i.e. isolating the Minimum Description Length component). We also compared the effects of tree pruning as a form of refinement operation; it mainly differs from a global data analysis in that each disjunctive hypothesis – or tree branch – is not analyzed independently, but remains intertwined to the tree structure.

Table 1 illustrates results for predictive accuracy on all artificial and real-world domains. Each group of 9- and 12-feature concepts is ordered by increasing variation $\nabla$. Columns for the different versions of *C4.5*rules and *C4.5*trees-pruned show the increase/decrease of accuracy against *C4.5*trees-unpruned. On the set of artificial concepts, both *C4.5*rules-*Std* and *C4.5*rules-*GDA* increasingly outperform *C4.5*trees-unpruned as $\nabla$ grows higher (the effect being more evident for 12-feature than for 9-feature concepts), except when $\nabla > 50\%$ (explained in Sect. 4) This trend is not observed on *C4.5*rules-*MDL*. The use of pruning exhibits a significant gain only until $\nabla$ is high. The same results are depicted in Figs. 6a and 6b for 9- and 12- feature concepts respectively, where we computed a regression line for each version of *C4.5*rules and *C4.5*trees-pruned against $\nabla$. An overall comparison for *C4.5*rules reveals *C4.5*rules-*Std* attains the highest advantage, proving the benefit of combining both *C4.5*rules-*GDA* and *C4.5*rules-*MDL*, and that *C4.5*rules-*GDA* is the component providing the most

**Table 1.** Tests on predictive accuracy for both artificial and real-world domains. Columns for the different versions of *C4.5*rules and *C4.5*trees-pruned show the increase/decrease of accuracy against *C4.5*trees-unpruned. The column for *DALI*rules is relative to *DALI*trees. Significant differences (at the p = 0.05 level) are marked with an asterisk.

| Concept | $\nabla$ | *C4.5*trees- | | *C4.5*rules- | | | *DALI*- | |
|---|---|---|---|---|---|---|---|---|
| | (%) | unpruned | pruned | *Std* | *GDA* | *MDL* | trees | rules |
| DNF9*a* | 14 | 99.2 | −0.7 | +0.5 | +0.3 | +0.0 | 100.0 | +0.0 |
| CNF9*a* | 17 | 99.5 | −0.6 | +0.5* | +0.5* | +0.0 | 100.0 | +0.0 |
| MAJ9*a* | 17 | 100.0 | +0.0 | +0.0 | −0.8 | +0.0 | 100.0 | +0.0 |
| MAJ9*b* | 21 | 82.1 | −0.1 | +2.1* | +1.4* | −0.2 | 85.3 | +0.0 |
| CNF9*b* | 22 | 94.0 | +2.3* | +6.0* | +3.8* | +0.0 | 100.0 | −0.2 |
| MUX9 | 22 | 86.8 | 0.9 | +8.5* | +7.0* | −0.8 | 99.0 | −0.2 |
| DNF9*b* | 24 | 83.0 | +0.6 | +10.7* | +5.0* | −0.7 | 99.4 | +0.0 |
| PAR9*a* | 33 | 62.5 | +13.5* | +27.7* | +21.7* | −2.2 | 98.4 | +1.3* |
| PAR9*b* | 67 | 43.1 | +1.9* | +2.6* | +0.6* | +3.6* | 43.6 | +0.1 |
| MAJ12*a* | 13 | 100.0 | +0.0 | +0.0 | +0.0 | +0.0 | 100.0 | +0.0 |
| DNF12*a* | 15 | 99.7 | +0.0 | +0.3* | +0.3* | +0.0 | 99.9 | +0.0 |
| CNF12*a* | 15 | 99.6 | −0.1 | +0.4* | +0.2* | +0.0 | 100.0 | +0.0 |
| MAJ12*b* | 18 | 84.5 | −0.1 | +3.0* | +2.7* | −0.6* | 87.9 | +0.7* |
| CNF12*b* | 19 | 89.8 | +0.7 | +9.8* | +5.1* | −1.3 | 99.7 | −0.1 |
| DNF12*b* | 20 | 89.5 | +3.2* | +9.6* | +6.6* | −0.6 | 99.2 | +0.0 |
| MUX12 | 21 | 84.9 | +0.1 | +12.9* | +8.7* | −2.9* | 99.4 | −0.2 |
| PAR12*a* | 33 | 58.6 | +11.9* | +33.7* | +24.0* | −3.8* | 92.9 | +6.4* |
| PAR12*b* | 67 | 46.5 | +0.7* | +1.6* | +0.2* | +2.5* | 46.0 | +0.5* |
| TicTacToe | | 85.8 | −0.3* | +13.3* | +10.4* | +0.5* | 98.4 | −0.4 |
| Lympho[2] | | 80.2 | +3.3* | +3.0* | +2.1* | −0.5* | 87.9 | −0.2 |
| Lympho[3] | | 74.4 | +3.1* | +6.1* | +5.4* | +1.7* | 84.6 | −0.4 |
| Promoters | | 81.9 | −0.6 | +5.0* | +3.5* | −0.9 | 83.4 | +0.0 |
| Cancer | | 95.1 | −0.6* | +0.8* | +0.5* | −0.3* | 95.7 | +0.3 |
| Hepatitis | | 82.3 | −1.2* | −0.4 | −0.6 | −0.5 | 80.2 | −0.6 |
| ABS AVRG | | 83.46 | 85.0 | 90.0 | 86.5 | 83.2 | 90.9 | 91.2 |

significant contribution. For real-world domains, the improvement of *C4.5*rules-*Std* and *C4.5*rules-*GDA* over *C4.5*trees-unpruned is observed on the Tic-Tac-Toe, Lymphography, and Promoters domains, where $\nabla$ may be relatively high due to interaction among features bearing a low representation to the target concept (e.g., board-configurations used to determine win or loose), but not so evident on the Hepatitis domain, which may be characterized by comprising highly representative features (i.e. denoting low $\nabla$).

To test the effect of using a global data analysis over a decision tree with multiple-feature tests on every node, we compared the difference in predictive accuracy between *DALI* (Sect. 5.1) and a modified version of *C4.5*rules that accepts as input a decision tree from *DALI*. The new version, named *DALI*rules, operates as follows:

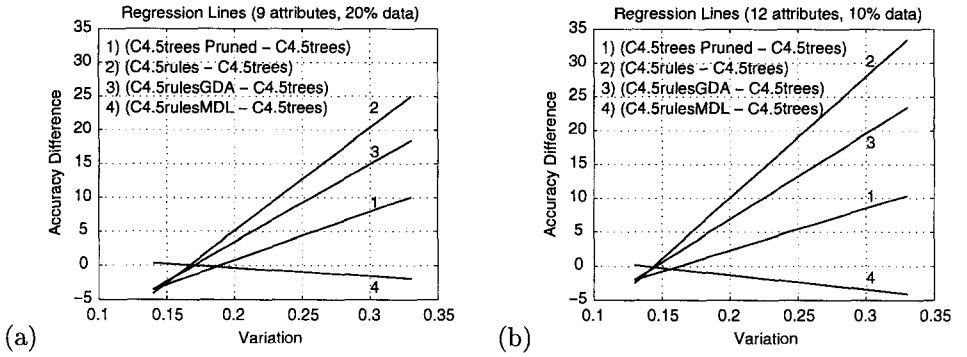


**Fig. 6.** Regression lines for the columns of *C4.5* (all different versions) on Table 1 vs. Variation $\nabla$ on (a) 9- and (b) 12-attribute concepts. Numbers enclosed in parentheses show the mean of the residuals between the linear model and the actual values (the same applies to Fig. 5.3).

1. Given a tree, $T_m$, with multiple feature tests on each node, define a new training sample $S_m$, such that every feature $x_{m_i}$ in $S_m$ corresponds to a tree node in $T_m$ (i.e., every new feature is a combination of the original-feature set, used as a splitting function on $T_m$).

2. Apply *C4.5*rules-*Std* to the tree $T_m$ output by *DALI*, and to the corresponding new training sample $S_m$, which is now described in terms of feature set $(x_{m_1}, x_{m_2} \cdots, x_{m_r})$.

Table 1 shows the results of comparing *DALI*rules with *DALI* in terms of predictive accuracy. The trend of accuracy increase as $\nabla$ grows is apparently delayed until $\nabla$ gets close to 50%. We note the use of more refined partitions over the instance space alleviates the effects of the fragmentation problem but does not eliminate it (Sect. 4), as evidenced by the results on parity concepts PAR9a and PAR12a (see [25]), where the advantage for *DALI*rules is significant. None of the real-world domains may achieve this high $\nabla$, where no significant difference is observed between these two systems. Figures 7a and 7b depict regression lines for the differences on predictive accuracy between *DALI*rules and *DALI*. An increase of predictive accuracy is evident for *DALI*rules on 12-feature concepts.

We finally compared absolute predictive accuracy averaged over all artificial and real-world domains, as shown on the last row of Table 1. The performance of *DALI*rules supports the claims of the importance of combining 1) feature-construction techniques (see *DALI*trees' performance), and 2) a global evaluation of each disjunctive hypothesis (see *C4.5*rules-*Std*' and *C4.5*rules-*GDA*' performance), to solve the fragmentation problem.

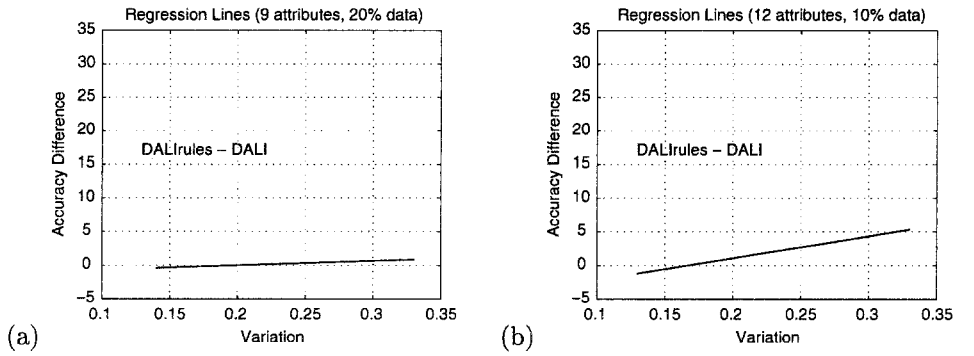


**Fig. 7.** Regression lines for the column of *DALI*-rules on Table 1 vs. Variation $\nabla$ on (a) 9- and (b) 12-attribute concepts.

## 6 Summary and Conclusions

The divide-and-conquer implementation of decision tree induction is responsible for a progressive loss of statistical support at every new partition, as the number of examples giving credibility to every disjunctive hypothesis progressively diminishes. This fragmentation problem has little effect on domains with low variation $\nabla$ (i.e., on simple domains), because every disjunctive hypothesis is supported by large regions of positive examples; but the same problem is severely aggravated by the instability imposed by high variation over the instance space.

We experimented with a new decision tree inducer, *DALI*, to prove the benefit of using refined partitions over the instance space in combating the fragmentation problem. We identified an additional important step to solve this problem consisting of independently assessing the value of each disjunctive hypothesis against all training data. This "global data analysis", embedded in *C4.5*rules, proved effective in improving the classifications made by *C4.5*trees (single-feature tests), and *DALI* (multiple-feature tests), with a positive correlation to $\nabla$ (i.e. predictive accuracy increased as $\nabla$ grew higher), except when $\nabla > 50\%$ because of the similarity-based assumption. We suggest combining feature construction with global data analysis as a robust solution against the fragmentation problem.

One important conclusion can be drawn from this study: that a better understanding of what causes a learning algorithm to succeed or fail can be attained if the algorithm is viewed as the combination of multiple components, each component exerting a particular effect during learning. The development of learning algorithms could be guided by the combination of those – well understood – learning components known to provide the correct generalizations under the class of domains of study (e.g., all structured real-world domains, since no universal learner is attainable [24]).

## Acknowledgments

We are grateful to the anonymous reviewers for their helpful suggestions. Our work was supported in part by National Science Foundation (IRI 92–04473), Consejo Nacional de Ciencia y Tecnología (México), and Norges Forskningsråd (Norway).

### A. Definitions for Artificial Concepts

Let: $X = (x_1, x_2, \ldots, x_n)$,
$\text{address}(x_1, x_2, \ldots, x_m) = 2^0 x_1 + 2^1 x_2 + \ldots + 2^{m-1} x_m$

**Definitions:**

DNF9$a$ : $x_2x_3 + \bar{x_2}x_3x_7 + x_2\bar{x_3}x_8 + x_2\bar{x_7}\bar{x_8} + \bar{x_3}x_7x_8$

DNF9$b$ : $x_1x_2x_3 + \bar{x_1}\bar{x_2} + x_7x_8x_9 + \bar{x_7}\bar{x_9}$

DNF12$a$ : $x_1x_2x_9x_{12} + x_1x_2\bar{x_9}x_{11} + \bar{x_1}x_2x_5x_9 + \bar{x_1}\bar{x_2}x_8\bar{x_9}$

DNF12$b$ : $x_1x_2\bar{x_7}\bar{x_8} + x_1x_2x_{11}x_{12} + x_1\bar{x_2}\bar{x_{11}}x_{12} +$
$\bar{x_1}x_2x_{11}\bar{x_{12}} + \bar{x_1}\bar{x_2}\bar{x_{11}}\bar{x_{12}} + x_7x_8x_{11}x_{12}$

CNF9$a$ : $(x_4 + x_6)(\bar{x_4} + \bar{x_5} + x_7)$

CNF9$b$ : $(x_1 + \bar{x_2} + x_3)(\bar{x_1} + x_2 + x_5)(x_1 + x_2 + \bar{x_3})$

CNF12$a$ : $(x_8 + x_9)(x_7 + x_6 + x_5)(\bar{x_8} + \bar{x_6} + x_2)$

CNF12$b$ : $(x_1 + x_2 + x_6)(\bar{x_1} + \bar{x_2} + x_7)(x_9 + x_{10} + x_{12})$
$(\bar{x_6} + \bar{x_7} + x_8)(x_9 + \bar{x_{10}} + \bar{x_{12}})(\bar{x_8} + \bar{x_9} + \bar{x_7})$

MAJ9$a$ : $\{X \mid (\sum_{i=1}^{9} x_i) \geq 3\}$

MAJ9$b$ : $\{X \mid (\sum_{i=1}^{9} x_i) \geq 6\}$

MAJ12$a$ : $\{X \mid (\sum_{i=1}^{12} x_i) \geq 4\}$

MAJ12$b$ : $\{X \mid (\sum_{i=1}^{12} x_i) \geq 8\}$

PAR9$a$ : $\{X \mid ((\sum_{i=1}^{3} x_i) \bmod 2) > 0\}$

PAR9$b$ : $\{X \mid ((\sum_{i=1}^{6} x_i) \bmod 2) > 0\}$

PAR12$a$ : $\{X \mid ((\sum_{i=1}^{4} x_i) \bmod 2) > 0\}$

PAR12$b$ : $\{X \mid ((\sum_{i=1}^{8} x_i) \bmod 2) > 0\}$

MUX9 : $\{X \mid x_{(\text{address}(x_1,x_2)+3)} = 1\}$

MUX12 : $\{X \mid x_{(\text{address}(x_1,x_2,x_3)+4)} = 1\}$

## References

1. Breiman L., Friedman J. H., Olshen R. A., Stone C. J.: Classification And Regression Trees. Wadsworth, Belmont, CA. (1984)
2. Brodley C. E., Utgoff P. E.: Multivariate Decision Trees. Machine Learning, 19:1, (1995) 45–78
3. Catlett J.: On Changing Continuous Attributes into Ordered Discrete Attributes. In Proceedings of the European Working Session on Learning, (1991) 164–178
4. Clark P., Niblett T.: The CN2 Induction Algorithm. Machine Learning, 3:4, (1989) 261–284.
5. Fayyad U.: Branching on Attribute Values in Decision Tree Generation. In Proceedings of the 12th National Conference on Artificial Intelligence, (1994) 601–606
6. Kohavi R.: Bottom-Up Induction of Oblivious Read-Once Decision Graphs. In Proceedings of the European Conference on Machine Learning, (1994) 154–169

7. Kohavi R.: A Study of Cross Validation and bootstrap for Accuracy Estimation and Model Selection. In Proceedings of the 24th International Joint Conference on Artificial Intelligence, (1995) 1137-1145
8. Luc D., László G., Gábor L.: A Probabilistic Theory of Pattern Recognition. Springer-Verlag, New York (1996)
9. Matheus C. J.: Feature Construction: An Analytical Framework and an Application to Decision Trees. Ph.D. thesis. University of Illinois at Urbana-Champaign.
10. Michalsky R., et al.: The Multipurpose Incremental Learning System AQ15 and its Testing Application to Three Medical Domains. In Proceedings of the 5th National Conference on Artificial Intelligence, (1986) 1041–1045
11. Oliveira A. L.: Inferring Reduced Ordered Decision Graphs of Minimum Description Length. In Proceedings of the 12th International Conference on Machine Learning, (1995) 421–429
12. Pagallo G., Haussler D.: Boolean Feature Discovery in Empirical Learning. Machine Learning, 5:1, (1990) 71–99
13. Pérez E., Rendell L. A.: Learning Despite Concept Variation by Finding Structure in Attribute-based Data. In Proceedings of the 13th International Conference on Machine Learning, (1996) 391–399
14. Quinlan J. R.: Induction of Decision Trees. Machine Learning, 1:1, (1986) 81–106
15. Quinlan, J. R.: Generating Production Rules from Decision Trees. In Proceedings of the 10th International Joint Conference on Artificial Intelligence, (1987) 304–307
16. Quinlan J. R.: *C4.5* Programs for Machine Learning. Morgan Kaufmann Publishers, Inc., Palo Alto, CA. (1994)
17. Quinlan J. R., Cameron J.: Oversearching and Layered Search in Empirical Learning. In Proceedings of the 24th International Joint Conference on Artificial Intelligence, (1995) 1019–1024
18. Ragavan H., Rendell L. A.: Lookahead Feature Construction for Learning Hard Concepts. In Proceedings of the 10th International Conference on Machine Learning, (1993) 252–259
19. Rendell L. A.: A General Framework for Induction and a Study of Selective Induction. Machine Learning, 1:2, (1986) 177–226
20. Rendell L. A., Seshu R.: Learning Hard Concepts through Constructive Induction: Framework and Rationale. Computational Intelligence, 6, (1990) 247–270
21. Rivest L.R.: Learning Decision Lists. Machine Learning, 2:3, (1987) 229-246
22. Rymon, R.: An SE-tree Based Characterization of the Induction Problem. In Proceedings of the 10th International Conference on Machine Learning, (1993) 268–275
23. Satosi W.: Knowing and Guessing. John Wiley & Sons, New York (1969)
24. Schaffer J.: A Conservation Law for Generalization Performance. In Proceedings of the 11th International Conference on Machine Learning (1994), 259–265
25. Thornton C.: Parity: The Problem that Won't Go Away. In Proceedings of the 11th Biennial Conference of the Canadian Society for Computational Studies of Intelligence, Toronto Ontario, Canada, (1996) 362–374
26. Vilalta R., Blix G., Rendell L.: Techniques for Efficient Feature Construction in Decision Tree Induction. Document in Preparation. Beckman Institute, University of Illinois at Urbana-Champaign (1997)
27. Webb G.I.: OPUS: An Efficient Admissible Algorithm for Unordered Search. Journal of Artificial Intelligence Research, 3 (1995), 431-465
28. Weiss S., Indurkhya N.: Optimized Rule Induction. IEEE Expert, 8:6, (1993) 61–69