Understanding Accuracy Performance Through Concept Characterization and Algorithm Analysis

Ricardo Vilalta
IBM T.J. Watson Research Center
30 Saw Mill River Rd., Hawthorne N.Y., 10532 USA
vilalta@us.ibm.com

Abstract

This paper illustrates an approach to understand differences in accuracy performance among learning algorithms. The study proceeds in two steps by 1) providing a characterization of real-world domains, and 2) by analyzing the internal mechanism of two learning algorithms, *C4.5*trees and *C4.5*rules. A functional view of the internal components of these two algorithms is related to the characteristics of the domain under study; the analysis helps to predict differences in accuracy behavior. Empirical results obtained over a set of real-world domains correlate well with the predictions. This two-step approach advocates the view of meta-learning as the quest for a theory that can explain the class of domains for which a learning algorithm will output accurate predictions.

1 INTRODUCTION

The goal of classification —or supervised learning— is to find an approximation or hypothesis to a target concept $\mathcal{C}$ that assigns objects (i.e., processes, events) into different categories or classes. Classification has been the subject of much study in the past, and has proved a useful tool in real-world applications. Nevertheless, the reasons explaining why an algorithm is more successful than others in giving good approximations to $\mathcal{C}$ remain elusive (Brodley, 1993).

To elucidate the problem above, experiments employing a broad spectrum of learning algorithms have been sought to find the class of domains for which each algorithm works best (Michie, 1994; Weiss & Kulikowski, 1990). The idea is to trace a link between a learning algorithm and the domains on which this algorithm outperforms all other competitors. Other approaches include applying meta-learning techniques to find rules identifying the domains on which an algorithm attains high accuracy (Gama & Brazdil, 1995; Brazdil, Gama, & Henery, 1994), and selecting the best of a set of classifiers —using a 10-fold cross validation approach— as the best approximation to the target concept (Schaffer, 1993).

The approaches above fail to recognize two issues. On the one hand, a richer characterization of domains is necessary to produce theories that can explain differences in performance. Current studies, e.g., StatLog project (Michie, 1994), are limited to simple parameters (e.g., number of examples, number of classes, number of features), statistical measures (e.g., feature correlation, homogeneity of covariances), and information-theoretic measures (e.g., class entropy). Additional measures are required to make a distinction between concepts denoting regular patterns over the instance space that commonly lead to the discovery of concise representations, and concepts characterized by many irregularities which often lead to long representations. A proper characterization of concept complexity can explain accuracy performance by identifying the degree of match between the concept and the learning bias of the algorithm under analysis.

On the other hand, empirical comparisons of learning algorithms show no distinction among the internal components of an algorithm, and hence of the contribution of each component during learning (Vilalta, 1998). The particular bias bestowed on the algorithm is assumed to pervade the entire mechanism; the individual contribution of each learning component becomes indistinguishable. How exactly this bias is defined cannot be explained at a more refined level than the whole algorithm itself. Whatever the effects of the

components embedded in the algorithm are, these tend to be averaged altogether.

In this study I show how heeding these issues leads to a better understanding of what causes an algorithm to yield good (conversely poor) concept approximations. The general approach is to perform a functional decomposition analysis on a learning algorithm to understand its different component effects, and to relate those effects to the characteristics of the domain under analysis. To that end, I start by characterizing real-world domains according to two parameters: an estimation of the difficulty of the domain based on the amount of class variation, and an estimation of the average distance between examples in the training set. I then study the internal mechanism of two learning algorithms, a decision tree and a rule-based system (*C4.5*trees and *C4.5*rules (Quinlan, 1994)), explaining the different bias introduced by their components (Vilalta, Blix, & Rendell, 1997). The two steps above, concept characterization and algorithm analysis, are tied together by using them to predict the situations in which different —or similar— accuracy performance between the two algorithms is expected. Empirical results obtained over a set of real-world domains give support to the performance predictions, in effect casting light into the algorithm-selection problem.

The organization of this paper follows. Section 2 provides background information on concept variation. Section 3 gives a characterization of real-world domains. Section 4 analyzes the functional differences between *C4.5*trees and *C4.5*rules. Section 5 gathers information from previous sections to explain accuracy performance. Section 6 shows empirical results on real-world domains to support the ideas presented on Section 5. Section 7 ends with the conclusions.

2 PRELIMINARIES

Classification starts with a learning domain as input, defined as a 3-tuple $< T_{train}, m, \mathcal{D} >$, comprising a finite training set T_{train} of cardinality m, and a probability distribution $\mathcal{D}$. Training set T_{train} is made up of examples, $T_{train} : \{(X_i, c_i)\}_{i=1}^{m}$, where each vector X_i is characterized as a point in an n-dimensional feature space, i.e., is described by n features $< x_i^1, x_i^2, \cdots, x_i^n >$, and labeled with a specific class c_i (e.g., $+$,$-$) according to an unknown target function $\mathcal{C}$. We assume T_{train} is obtained by uniformly sampling examples from probability distribution $\mathcal{D}$. After analyzing T_{train}, the end-product of classification is a hypothesis $\mathcal{H}$ approximating $\mathcal{C}$. $\mathcal{H}$ may then be used to classify examples in an off-training set T_{test}. We shall consider an inductive learning algorithm successful if the classifications made by $\mathcal{H}$ on the off-training set T_{test} are similar to the classifications made by $\mathcal{C}$.

2.1 CONCEPT VARIATION

The degree of structure of a concept $\mathcal{C}$ can be estimated by measuring the amount of uniformity of the distribution of class labels throughout the feature space. A highly irregular space is characterized by many small disjuncts, often in need for long concept representations. In contrast, a uniform space comprises large regions of examples sharing similar class labels, potentially allowing for compact concept descriptions.

The degree of *lack* of structure has been estimated before through a measure known as concept variation ∇ (Rendell & Seshu, 1990; Pérez & Rendell, 1996). ∇ estimates the probability that any two neighbor examples differ in class value, roughly measuring the amount of irregularity in the class-label distribution. The definition of ∇ works only in boolean spaces, and assumes all examples in the space are available. Let $X_1, X_2, \cdots, X_n$ be the n closest neighbors – at Hamming distance one – of an example X_i in an n-dimensional boolean space. Let the amount of concept variation for example X_i, $\sigma(X_i)$, be defined as

$$\sigma(X_i) = \frac{1}{n} \times \sum_{j=1}^{n} \delta(C(X_i), C(X_j)) \ , \qquad (1)$$

where $\delta(C(X_i, X_j)) = 1$ if $C(X_i) \neq C(X_j)$ and 0 otherwise. Concept variation is defined as the average of this factor when applied to every example in the feature space:

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

The applicability of ∇ is limited to artificial domains where all possible examples can be generated. Pérez and Rendell (1996) show how concepts with high ∇ are difficult to learn by most conventional inductive algorithms.

3 CONCEPT CHARACTERIZATION

3.1 A NEW MEASURE OF CONCEPT VARIATION

This section gives a characterization of real-world domains based on a more general definition of concept variation. ∇ will be replaced by ∇' by considering nominal and numeric features, and by using only a limited sample of examples.

The definition of ∇ (Section 2.1) assumes we have access to the closest neighbors of a given example. Since in real-world domains that assumption no longer holds, a distance metric will be used to determine the contribution of each example to the amount of concept variation. Formally, let $X_i =< x_i^1, x_i^2, \cdots, x_i^n >$ and $X_j =< x_j^1, x_j^2, \cdots, x_j^n >$ be any two examples in training set T_{train}. Let the distance between these two examples be defined as

$$D(X_i, X_j) = \sqrt{\sum_{k=1}^{n} d(x_i^k, x_j^k)^2} \quad , \tag{3}$$

For nominal features, $d(x_i^k, x_j^k)$ is defined as

$$d(x_i^k, x_j^k) = \begin{cases} 1 & \text{if } x_i^k \neq x_j^k \\ 0 & \text{if } x_i^k = x_j^k \end{cases} \quad , \tag{4}$$

For numeric features, $d(x_i^k, x_j^k)$ is defined as

$$d(x_i^k, x_j^k) = \frac{|x_i^k - x_j^k|}{\text{MAX}(x^k) - \text{MIN}(x^k)} \quad , \tag{5}$$

where $\text{MAX}(x^k)$ and $\text{MIN}(x^k)$ are the maximum and minimum values observed for attribute k in T_{train}.

Now, if examples X_i and X_j differ in class value, i.e., $C(X_i) \neq C(X_j)$, the contribution to the amount of concept variation will depend on their distance; the closer the examples, the higher the contribution. The following function will be used to weight such contribution:

$$W(X_i, X_j) = \frac{1}{2^{\alpha \times \frac{D(X_i, X_j)}{\sqrt{n} - D(X_i, X_j)}}} \in [0, 1] \quad , \tag{6}$$

where α is user defined and modulates the effect of distance: the higher α the less contribution is given as the distance between the two examples grows larger. Here, the default value for α is two[1]. The term at the right of α forces $W(X_i, X_j)$ to reach the upper bound of one if $D(X_i, X_j) = 0$; the lower bound of zero is reached when $D(X_i, X_j) = \sqrt{n}$, corresponding to the maximum distance between two examples in an n-dimensional feature space.

Let the amount of concept variation for a single example X_i be defined as

$$\sigma'(X_i) = \frac{\sum_{j=1, j \neq i}^{m} W(X_i, X_j) \times \delta(C(X_i), C(X_j))}{\sum_{j=1, j \neq i}^{m} W(X_i, X_j)} \quad , \tag{7}$$

where $m = |T_{train}|$ and again $\delta(C(X_i, X_j)) = 1$ if $C(X_i) \neq C(X_j)$ and 0 otherwise. The new measure for concept variation, ∇', is defined as the average of this factor over all training examples:

$$\nabla' = \frac{1}{m} \times \sum_{i=1}^{m} \sigma'(X_i) \ \in [0, 1] \ . \tag{8}$$

3.2 THE AVERAGE WEIGHTED DISTANCE

While ∇' estimates the variability of class labels among examples, it fails to capture how dense or sparse is the example distribution in the training set. For example, if a training set comprises only two examples having different class labels, then $\nabla' = 1$, independently of the distance between these two examples. Section 5 will show the importance of this measure for understanding accuracy performance.

Given an example X_i, the average weighted distance between X_i and all other examples in the training set T_{train}, is defined as

$$\upsilon(X_i) = \frac{\sum_{j=1, j \neq i}^{m} W(X_i, X_j) \times D(X_i, X_j)}{\sum_{j=1, j \neq i}^{m} W(X_i, X_j)} \quad , \tag{9}$$

The average weighted distance between examples, Υ, is defined as the average of this factor over all training examples:

$$\Upsilon = \frac{1}{m} \times \sum_{i=1}^{m} \upsilon(X_i) \ \in [0, 1] \ . \tag{10}$$

[1] A value of $\alpha = 1$ over-weighted examples lying far apart, while the opposite occurred for $\alpha > 2$; the form of $W(X_i, X_j)$ is subject of ongoing study.

Table 1: Characterization of real-world domains.

Concept	∇'	Υ	Size	Features
Zoo	0.10	1.76	101	16
Voting	0.17	2.02	435	16
Mushroom	0.20	2.44	8124	21
NewThyroidHypo	0.21	0.34	215	5
Cancer	0.22	0.77	683	9
Hepatitis	0.24	1.91	112	18
NewThyroidHyper	0.24	0.34	215	5
Promoters	0.31	5.93	106	57
Lymphography3	0.38	2.50	148	18
Heart	0.38	1.77	297	13
TicTacToe	0.38	1.83	958	9
Lymphography2	0.39	2.50	148	18
Credit	0.41	1.40	126	10
Crx	0.42	1.85	653	15
StarCluster	0.42	0.59	213	6
Ionosphere	0.43	1.80	351	34
Diabetes	0.44	0.57	768	8
KrvsKp	0.47	2.84	3196	36
Bupa	0.49	0.40	345	6

Table 1 shows the result of applying ∇' and Υ over a subset of real-world domains extracted from the UCI Irvine repository[2] (Merz & Murphy, 1998); the complexity to compute both metrics is $O(nm^2)$. Domains are shown by increasing ∇', distributed in the range [0,0.5]. Υ varies within the range [0,3], except for the promoters domain exhibiting a sparse example distribution. In addition to ∇' and Υ, Table 1 completes the characterization of domains by displaying information on the size of the training set and on the number of features.

4 ALGORITHM ANALYSIS

We shall now consider the importance of scrutinizing the interior mechanisms of learning algorithms to pinpoint specific reasons for their performance. The algorithms of study are *C4.5*trees and *C4.5*rules (Quinlan, 1994). The following functional analysis seeks to understand their different performance behavior.

*C4.5*trees is a standard top-down univariate decision tree learner that recursively partitions the training set in search for regions of examples with similar class labels. Decision tree learners have the disadvantage of progressively lessening the evidential credibility of the decisions made at each node, as the number of examples in the training set dwindles on each partition. This problem, known as the *fragmentation problem* (Vilalta et al., 1997), may yield poor generalization performance when the concept under analysis requires long decision-tree structures.

*C4.5*rules, on the other hand, is a rule-based system inherently linked to *C4.5*trees. The mechanism for *C4.5*rules can be decomposed in two steps:

1. Given an unpruned decision tree from *C4.5*trees, form a rule from every branch of the tree that starts at the root node and ends on a leaf node; a rule is an implication: if cond_1 and cond_2 $\cdots$ and $\text{cond}_d \rightarrow$ c, where cond_i is the feature-value (i.e. splitting-function value) encountered on every node along the tree branch, and c the class assigned to the leaf node.
2. Eliminate all irrelevant conditions from every rule in step 1 be evaluating each rule globally, against all training examples. In addition, apply a minimum description principle, according to a particular bit-encoding scheme, to remove rules from the rule set.

Consider the functional differences between *C4.5*trees and *C4.5*rules. Since both algorithms initially construct the same decision tree, any measurable difference in performance must be caused by the steps above. The first step simply transforms the representation without affecting prediction, i.e., the unpruned decision tree and the original set of rules are equivalent hypotheses. In the second step, the (possibly) overly specific generalizations produced by the decision tree are partially alleviated by assessing changes to each rule using statistics obtained from *all* training examples. Recurring to all available training examples when assessing the value of each rule alleviates the detrimental effects caused by the continuous partitioning of the training set. This *global evaluation* of rules —or partial hypotheses— is one important element to differentiate among learning algorithms, and may help to explain differences in performance.

5 UNDERSTANDING ACCURACY PERFORMANCE

But, when do we expect a global evaluation of partial hypotheses bring gains in accuracy performance? and, how is that related to the characteristics of the domain under analysis? This section attempts to answer these questions.

A comparison between *C4.5*trees and *C4.5*rules in terms of accuracy performance can be related to con-

[2]Except for domain StarCluster (Andrews & Herzberg, 1985).

Table 2: Tests on predictive accuracy for real-world domains.

Concept	*C4.5*trees		*C4.5*rules
	unpruned	pruned	
Zoo	93.94 (0.43)	94.76 (0.78)	95.94 (1.16)
Voting	94.72 (0.16)	95.34 (0.35)	95.7 (0.47)
Mushroom	100.00 (0.0)	100.00 (0.0)	100.00 (0.0)
NewThyroidHypo	96.4 (0.49)	96.4 (0.49)	96.4 (0.49)
Cancer	95.08 (0.15)	95.20 (0.27)	95.94 (0.56)
Hepatitis	82.32 (1.73)	81.08 (0.90)	81.94 (2.02)
NewThyroidHyper	96.78 (0.47)	96.68 (0.51)	96.76 (0.40)
Promoters	81.86 (3.46)	81.28 (3.57)	86.88 (1.94)
Lymphography3	74.44 (2.78)	77.54 (1.62)	80.46 (2.13)
Heart	76.16 (1.00)	75.58 (0.11)	77.58 (1.40)
TicTacToe	85.8 (0.66)	85.46 (0.71)	99.06 (0.19)
Lymphography2	80.22 (2.05)	83.46 (1.23)	83.20 (2.26)
Credit	76.56 (1.64)	81.42 (1.39)	80.04 (0.48)
Crx	82.8 (0.74)	86.36 (0.52)	86.08 (0.47)
StarCluster	98.4 (0.24)	98.4 (0.24)	98.4 (0.24)
Ionosphere	91.18 (0.65)	91.22 (0.66)	91.34 (0.93)
Diabetes	74.7 (0.77)	74.7 (0.68)	74.88 (0.78)
KrvsKp	99.36 (0.05)	99.42 (0.08)	99.48 (0.04)
Bupa	67.14 (2.14)	66.64 (0.74)	67.26 (1.72)

cept variation (Section 3.1) and to the average distance between examples (Section 3.2) as explained next. The terminal node or leaf of a decision tree classifies a region r of examples according to a majority-class vote on the examples covered by r. Consider only two classes: positive and negative, and let θ be defined as the difference between the number of training positive and negative examples in r; then for any example $X_i \in r$,

$$C_r(X_i) = \begin{cases} + & \text{if } \theta \geq 0 \\ - & \text{otherwise} \end{cases} \quad (11)$$

where C_r is the class predicted for an example falling in r.

Consider first the case of domains having low variation. Here, the effects of the fragmentation problem are harmless because in the presence of large areas covering many examples of similar class value, each leaf node delimits a region of examples r for which $\theta >> 0$, i.e., for which θ is stable. Such stability results in only a few partitions over the training set, ultimately increasing the confidence of the prediction during classification.

What would happen, however, if dissimilarity of class labels around any example is high (i.e., variation is high)? Such domains force *C4.5*trees to produce an excessive number of partitions over the training set. Searching for single class regions by separating examples away has the effect of quickly reducing the stability of any region r. If $\theta \sim 0$ (i.e. θ is unstable), then removing examples from r may cause θ shift sign, thereby increasing the number of misclassifications.

Nevertheless, a significant difference in accuracy performance between *C4.5*trees and *C4.5*rules is not expected if, in addition to high variation, the distance between examples is short (i.e., density is high), because in this case the class-label distribution in a region r is already closely determined by the training examples; the global-evaluation refinement provided by *C4.5*rules would find little room for improvement.

But when variation is high and the distribution of examples in the training set is sparse (i.e., the distance between examples is large), a higher probability exists for the generalizations produced by *C4.5*rules to reverse the negative effects originated by the fragmentation problem. In summary, concepts characterized by a combination of high variation and a sparse example distribution fall outside the learning bias of *C4.5*trees (and decision-tree learners in general); it is on those same concepts that *C4.5*rules is expected to show an advantage over *C4.5*trees.

6 EMPIRICAL RESULTS

To prove the claims of Section 5, this section compares *C4.5*trees and *C4.5*rules in terms of predictive accuracy. Table 2 shows the results of applying a 10-fold cross validation averaged over five times on both

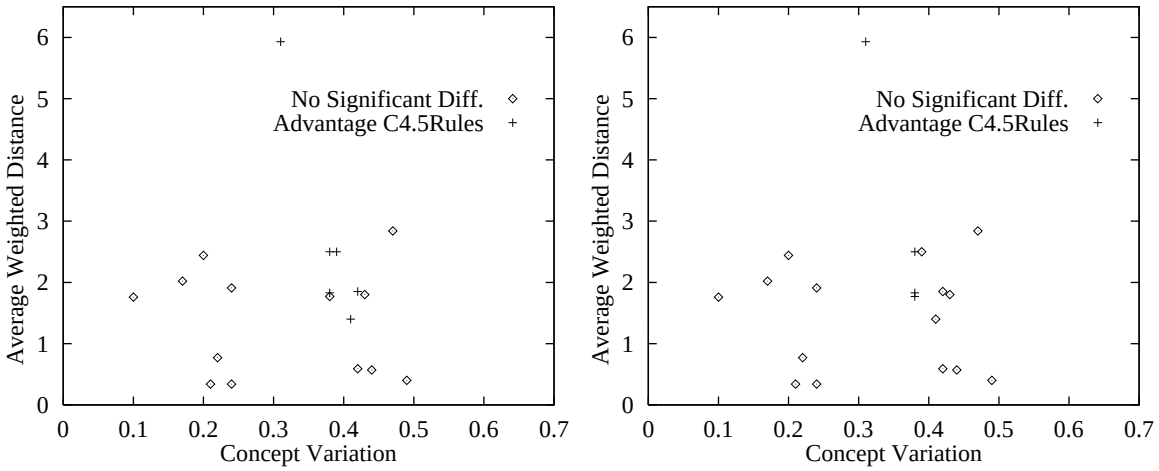

Figure 1: Domains are mapped according to concept variation ∇' and weighted average distance Υ. (left) *C4.5*rules is compared to *C4.5*trees unpruned; (right) *C4.5*rules is compared to *C4.5*trees pruned.

algorithms (numbers enclosed in parentheses represent standard deviations). A distinction is made between the unpruned and pruned trees output by *C4.5*trees (it is the first kind that serves as input to *C4.5*rules; Section 4). Each cross-validation run employed all available examples. In the following, statistical tests of significance are set at the 0.05 level assuming a two-sided t-student distribution.

Figure 1 shows a 2-dimensional plane in which each domain is mapped on two coordinates: concept variation, ∇' (Section 3.1), on the X-axis, and weighted average distance between examples, Υ (Section 3.1), on the Y-axis. Based on the results shown on Table 2, the set of domains can be divided in two: one group (labeled with diamonds) where either no significant difference in accuracy exists between the two algorithms, or the absolute difference is too small (within one percent point); and a second group (labeled with crosses) where *C4.5*rules shows a significant advantage over *C4.5*trees. Results are similar when *C4.5*rules is compared to *C4.5*trees unpruned (Figure 1 left), and when compared to *C4.5*trees pruned (Figure 1 right); it can be observed that *C4.5*rules shows an advantage in predictive accuracy on the medium-upper right region of the plane where both ∇' and Υ take on medium to high values. This agrees with the analysis of Section 5, except the same region of the plane comprises a few domains with no significant advantage for *C4.5*rules. A clear exception to the analysis seems to be the chess-end game domain (KrvsKp) at coordinates $(0.47, 2.84)$ where *C4.5*rules shows no significant advantage; repeated experiments using only 200 examples from the original set of $3,196$ (randomly sampled without replacement) to augment the value of Υ did not change the result.

The analysis of Section 5 is focused on a characterization of domains based on concept variation and example distribution, disregarding sources of difficulty such as noise or representational inadequacy. According to the experimental results, the analysis seems to provide conditions that are *necessary* to account for differences in accuracy performance, but not *sufficient*. This is a sign that additional parameters are needed to improve the concept characterization of Section 3.

7 CONCLUSIONS

The experimental results of Section 6 are in close agreement with the analysis explaining the situations in which *C4.5*rules is expected to outperform *C4.5*trees (Section 5). More parameters are needed, however, to strengthen the characterization of concepts developed in Section 3. In addition, further work is required to expand this kind of analysis to other learning paradigms, e.g., neural networks, instance-based models.

Selecting an algorithm A that can produce good concept approximations to a target concept $\mathcal{C}$ depends on two factors: the learning bias embedded in A, and how well this bias matches the characteristics of $\mathcal{C}$. This paper shows how the meta-learning problem above can be attacked in two steps: 1) find a proper characterization of $\mathcal{C}$, and 2) analyze the mechanism of A by identifying the effects produced by its internal components during

learning. Such methodology provides a causal explanation for the effects produced by the components of A according to the characteristics of $\mathcal{C}$. In contrast, simply using a global evaluation metric (e.g., predictive accuracy) to measure the performance of a new learning algorithm cannot help to identify the individual contribution of each component; no casual theory can be obtained to account for the new algorithm's behavior. Meta-learning is then more than differentiating algorithms based solely on accuracy performance, or the recursive application of learning techniques; it is the quest for a theory that can explain the class of domains for which a learning algorithm will output accurate predictions.

Acknowledgments

I am grateful to the anonymous reviewers for their valuable suggestions. This work was supported in part by Consejo Nacional de Ciencia y Tecnología (México), and by IBM T.J. Watson Research Center (New York, USA).

References

Andrews, D., & Herzberg, A. (1985). *Data: A Collection of Problems from Many Fields for the student and Research Worker*. Springer-Verlag.

Brazdil, P., Gama, J., & Henery, R. (1994). Characterizing the applicability of classification algorithms using meta level learning. In *European Conference on Machine Learning ECML-94*, pp. 83–102.

Brodley, C. E. (1993). Addressing the selective superiority problem: Automatic algorithm/model class selection. In *Proceedings of the Tenth International Conference on Machine Learning*, pp. 17–24. Morgan Kaufmann Publishers, Inc.

Gama, J., & Brazdil, P. (1995). Characterization of classification algorithms. In *7th Portuguese Conference on Artifical Intelligence, EPIA*, pp. 189–200 Funchal, Madeira Island, Portugal.

Merz, C., & Murphy, P. (1998). *UCI repository of machine learning databases*. http:// www.ics.uci.edu/ mlearn/MLRepository.html.

Michie, D. (1994). *Machine Learning, Neural and Statistical Classification*. Ellis Horwood.

Pérez, E., & Rendell, L. A. (1996). Learning despite concept variation by finding structure in attribute-based data. In *Proceedings of the Thirteenth International Conference on Machine Learning*, pp. 391–399.

Quinlan, J. R. (1994). *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers, Inc., Palo Alto, CA.

Rendell, L. A., & Seshu, R. (1990). Learning hard concepts through constructive induction: Framework and rationale. *Computational Intelligence*, *6*, 247–270.

Schaffer, C. (1993). Selecting a classification method by cross-validation. *Machine Learning*, *13*, 135–143.

Vilalta, R. (1998). *On the Development of Inductive Learning Algorithms: Generating Flexible and Adaptable Concept Representations*. Ph.D. thesis, University of Illinois at Urbana-Champaign, http://www.research.ibm.com/people/v/vilalta.

Vilalta, R., Blix, G., & Rendell, L. A. (1997). Global data analysis and the fragmentation problem in decision tree induction. In *9th European Conference on Machine Learning*, pp. 312–326. Lecture Notes in Artificial Intelligence, Vol. XXX, Springer-Verlag, Heidelberg, http://www.research.ibm.com/people/v/vilalta.

Weiss, S. M., & Kulikowski, C. A. (1990). *Computer Systems That Learn*. Morgan Kaufmann Publishers, Inc., San Mateo, CA.