Knowl Inf Syst (2005)
DOI 10.1007/s10115-005-0211-z

**Knowledge and Information Systems**

RESEARCH ARTICLE

**J. Zhang · D.-K. Kang · A. Silvescu · V. Honavar**

# Learning accurate and concise naïve Bayes classifiers from attribute value taxonomies and data

Received: 1 November 2004 / Revised: 25 January 2005 / Accepted: 19 February 2005 / Published online: 24 June 2005


**Abstract** In many application domains, there is a need for learning algorithms that can effectively exploit attribute value taxonomies (AVT)—hierarchical groupings of attribute values—to learn compact, comprehensible and accurate classifiers from data—including data that are *partially specified*. This paper describes AVT-NBL, a natural generalization of the naïve Bayes learner (NBL), for learning classifiers from AVT and data. Our experimental results show that AVT-NBL is able to generate classifiers that are substantially more compact and more accurate than those produced by NBL on a broad range of data sets with different percentages of partially specified values. We also show that AVT-NBL is more efficient in its use of training data: AVT-NBL produces classifiers that outperform those produced by NBL using substantially fewer training examples.

**Keywords** Attribute value taxonomies · AVT-based naïve Bayes learner · Partially specified data

## 1 Introduction

Synthesis of accurate and compact pattern classifiers from data is one of the major applications of data mining. In a typical inductive learning scenario, instances

This paper is an extended version of a paper published in the 4th IEEE International Conference on Data Mining, 2004.

J. Zhang(✉) · D.-K. Kang · A. Silvescu
Department of Computer Science, Artificial Intelligence Research Laboratory, Computational Intelligence, Learning, and Discovery Program, Iowa State University, Ames, Iowa 50011-1040, USA E-mail: jzhang@cs.iastate.edu

V. Honavar
Department of Computer Science, Artificial Intelligence Research Laboratory; Computational Intelligence, Learning, and Discovery Program; Bioinformatics and Computational Biology Program, Iowa State University, Ames, Iowa 50011-1040, USA

to be classified are represented as ordered tuples of attribute values. However, attribute values can be grouped together to reflect assumed or actual similarities among the values in a domain of interest or in the context of a specific application. Such a hierarchical grouping of attribute values yields an attribute value taxonomy (AVT). Such AVT are quite common in biological sciences. For example, the Gene Ontology Consortium is developing hierarchical taxonomies for describing many aspects of macromolecular sequence, structure and function [5]. Undercoffer et al. have developed a hierarchical taxonomy that captures the features that are observable or measurable by the target of an attack or by a system of sensors acting on behalf of the target [41]. Several ontologies being developed as part of the Semantic Web-related efforts [7] also capture hierarchical groupings of attribute values. Kohavi and Provost have noted the need to be able to incorporate background knowledge in the form of hierarchies over data attributes in e-commerce applications of data mining [25, 26]. Against this background, algorithms for learning from AVT and data are of significant practical interest for several reasons:

(a) An important goal of machine learning is to discover comprehensible, yet accurate and robust, classifiers [34]. The availability of AVT presents the opportunity to learn classification rules that are expressed in terms of *abstract* attribute values leading to simpler, accurate and easier-to-comprehend rules that are expressed using familiar hierarchically related concepts [25, 44].
(b) Exploiting AVT in learning classifiers can potentially perform regularization to minimize overfitting when learning from relatively small data sets. A common approach used by statisticians when estimating from small samples involves *shrinkage* [29] to estimate the relevant statistics with adequate confidence. Learning algorithms that exploit AVT can potentially perform *shrinkage* automatically, thereby yielding robust classifiers and minimizing overfitting.
(c) The presence of explicitly defined AVT allows specification of data at different levels of precision, giving rise to *partially specified instances* [45]. The attribute value of a particular attribute can be specified at different levels of precision in different instances. For example, the medical diagnostic test results given by different institutions are presented at different levels of precision. Partially specified data are unavoidable in knowledge acquisition scenarios that call for integration of information from semantically heterogeneous information sources [10]. Semantic differences between information sources arise as a direct consequence of differences in ontological commitments [7]. Hence, algorithms for learning classifiers from AVT and partially specified data are of great interest.

Against this background, this paper introduces AVT-NBL, an AVT-based generalization of the standard algorithm for learning naïve Bayes classifiers from partially specified data. The rest of the paper is organized as follows: Sect. 2 formalizes the notions on learning classifiers with AVT taxonomies; Sect. 3 presents the AVT-NBL algorithm; Sect. 4 discusses briefly on alternative approaches; Sect. 5 describes our experimental results and Sect. 6 concludes with summary and discussion.

## 2 Preliminaries

In what follows, we formally define AVT and its induced instance space. We introduce the notion of partially specified instances and formalize the problem of learning from AVT and data.

### 2.1 Attribute value taxonomies

Let $A = \{A_1, A_2, \ldots, A_N\}$ be an ordered set of nominal attributes and let $\text{dom}(A_i)$ denote the set of values (the domain) of attribute $A_i$. We formally define attribute value taxonomy (AVT) as follows:

**Definition 2.1 (Attribute Value Taxonomy)** *Attribute value taxonomy $\mathcal{T}_i$ for attribute $A_i$ is a Tree-structured concept hierarchy in the form of a partially order set* $(\text{dom}(A_i), \prec)$*, where* $\text{dom}(A_i)$ *is a finite set that enumerates all attribute values in $A_i$ and $\prec$ is the partial order that specifies a relationships among attribute values in* $\text{dom}(A_i)$*. Collectively, $T = \{\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_N\}$ represents the ordered set of attribute value taxonomies associated with attributes $A_1, A_2, \ldots, A_N$.*

Let $Nodes(\mathcal{T}_i)$ represent the set of all values in $\mathcal{T}_i$, and $\text{Root}(\mathcal{T}_i)$ stand for the root of $\mathcal{T}_i$. The set of leaves of the tree, $Leaves(\mathcal{T}_i)$, corresponds to the set of *primitive values* of attribute $A_i$. The internal nodes of the tree (i.e. $Nodes(\mathcal{T}_i) - Leaves(\mathcal{T}_i)$) correspond to *abstract values* of attribute $A_i$. Each *arc* of the tree corresponds to a relationship over attribute values in the AVT. Thus, an AVT defines an abstraction hierarchy over values of an attribute.

For example, Fig. 1 shows two attributes with corresponding AVTs for describing students in terms of their *student status* and *work status*. With regard to the AVT associated with *student status*, *Sophomore* is a primitive value while *Undergraduate* is an abstract value. *Undergraduate* is an abstraction of *Sophomore*, whereas *Sophomore* is a further specification of *Undergraduate*. We can similarly define AVT over ordered attributes as well as intervals defined over numerical attributes.

After [21], we define a cut $\gamma_i$ for $\mathcal{T}_i$ as follows:

**Definition 2.2 (Cut)** *A cut $\gamma_i$ is a subset of elements in $Nodes(\mathcal{T}_i)$ satisfying the following two properties:* (1) *For any leaf $m \in Leaves(\mathcal{T}_i)$, either $m \in \gamma_i$ or $m$*

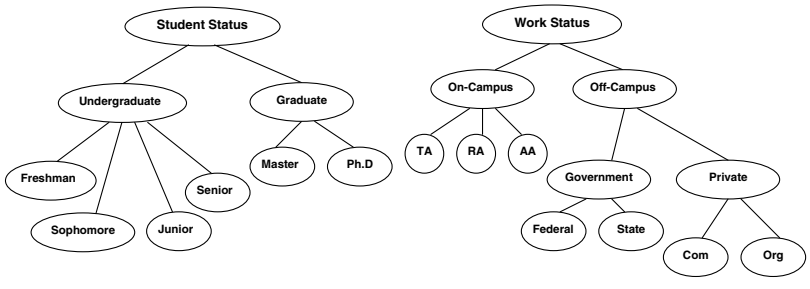



**Fig. 1** Two attribute value taxonomies on student status and work status.

*is a descendant of an element* $n \in \gamma_i$*; and* (2) *for any two nodes* $f, g \in \gamma_i$*,* $f$ *is neither a descendant nor an ancestor of* $g$.

The set of abstract attribute values at any given cut of $\mathcal{T}_i$ form a partition of the set of values at a lower level cut and also induce a partition of all primitive values of $A_i$. For example, in Fig. 1, the cut {*Undergraduate*, *Graduate*} defines a partition over all the primitive values {*Freshman*, *Sophomore*, *Junior*, *Senior*, *Master*, *PhD*} in the *student status* attribute, and the cut {*On-Campus*, *Off-Campus*} defines a partition over its lower level cut {*On-Campus*, *Government*, *Private*} in the *work status* attribute.

For attribute $A_i$, we denote $\Delta_i$ to be the set of all valid cuts in $\mathcal{T}_i$. We denote $\Delta = \times_{i=1}^{N} \Delta_i$ to be the Cartesian product of the cuts through the individual AVTs. Hence, $\Gamma = \{\gamma_1, \gamma_2, \ldots, \gamma_N\}$ defines a global cut through $T = \{\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_N\}$, where each $\gamma_i \in \Delta_i$ and $\Gamma \in \Delta$.

**Definition 2.3 (Cut Refinement)** *We say that a cut* $\hat{\gamma}_i$ *is a refinement of a cut* $\gamma_i$ *if* $\hat{\gamma}_i$ *is obtained by replacing at least one attribute value* $v \in \gamma_i$ *by its descendants* $\psi(v, \mathcal{T}_i)$*. Conversely,* $\gamma_i$ *is an abstraction of* $\hat{\gamma}_i$*. We say that a global cut* $\hat{\Gamma}$ *is a refinement of a global cut* $\Gamma$ *if at least one cut in* $\hat{\Gamma}$ *is a refinement of a cut in* $\Gamma$*. Conversely, the global cut* $\Gamma$ *is an abstraction of the global cut* $\hat{\Gamma}$.

As an example, Fig. 2 illustrates a demonstrative cut refinement process based on the AVTs shown in Fig. 1. The cut $\gamma_1 = \{$*Undergraduate*, *Graduate*$\}$ in the *student status* attribute has been refined to $\hat{\gamma}_1 = \{$*Undergraduate*, *Master*, *PhD*$\}$ by replacing *Graduate* with its two children, *Master*, *PhD*. Therefore, $\hat{\Gamma} = \{$*Undergraduate*, *Master*, *PhD*, *On-Campus*, *Off-Campus*$\}$ is a cut refinement of $\Gamma = \{$*Undergraduate*, *Graduate*, *On-Campus*, *Off-Campus*$\}$.

## 2.2 AVT-induced abstract-instance space

A classifier is built on a set of labeled training instances. The original instance space $I$ without AVTs is an instance space defined over the domains of all attributes. We can formally define AVT-induced instance space as follows:

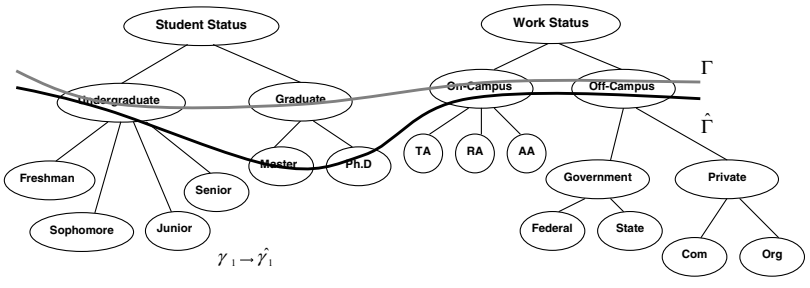



**Fig. 2** Cut refinement. The cut $\gamma_1 = \{$*Undergraduate*, *Graduate*$\}$ in the *student status* attribute has been refined to $\hat{\gamma}_1 = \{$*Undergraduate*, *Master*, *PhD*$\}$, such that the global cut $\Gamma = \{$*Undergraduate*, *Graduate*, *On-Campus*, *Off-Campus*$\}$ has been refined to $\hat{\Gamma} = \{$*Undergraduate*, *Master*, *PhD*, *On-Campus*, *Off-Campus*$\}$.

**Definition 2.4 (Abstract Instance Space)** *Any choice* $\Gamma$ *of* $\Delta = \times_i \Delta_i$ *defines an abstract instance space* $I_\Gamma$. *When* $\exists i\ \gamma_i \in \Gamma$ *such that* $\gamma_i \neq Leaves(T_i)$, *the resulting instance space is an abstraction of the original instance space* $I$. *The original instance space is given by* $I = I_{\Gamma_0}$, *where* $\forall i\ \gamma_i \in \Gamma_0$, $\gamma_i = Values(A_i) = Leaves(T_i)$, *that is, the primitive values of the attributes* $A_1 \ldots A_N$.

**Definition 2.5 (AVT-Induced Instance Space)** *A set of AVTs,* $T = \{T_1 \ldots T_N\}$, *associated with a set of Attributes,* $A = \{A_1 \ldots A_N\}$, *induces an instance space* $I_T = \cup_{\Gamma \in \Delta} I_\Gamma$ *(the union of instance spaces induced by all of the cuts through the set of AVTs* $T$*).*

## 2.3 Partially specified data

In order to facilitate precise definition of partially specified data, we define two operations on AVT $\mathcal{T}_i$ associated with attribute $A_i$.

- $depth(\mathcal{T}_i, v(A_i))$ returns the length of the path from root to an attribute value $v(A_i)$ in the taxonomy;
- $leaf(\mathcal{T}_i, v(A_i))$ returns a Boolean value indicating if $v(A_i)$ is a leaf node in $\mathcal{T}_i$, that is, if $v(A_i) \in Leaves(\mathcal{T}_i)$.

**Definition 2.6 (Partially Specified Data)** *An instance* $X_p$ *is represented by a tuple* $(v_{1p}, v_{2p}, \ldots, v_{Np})$. $X_p$ *is*

- *a partially specified instance if one or more of its attribute values are not primitive:* $\exists v_{ip} \in X_p, depth(\mathcal{T}_i, v_{ip}) \geq 0 \wedge \neg leaf(\mathcal{T}_i, v_{ip})$
- *a completely specified instance if* $\forall i\ v_{ip} \in Leaves(\mathcal{T}_i)$.

Thus, a partially specified instance is an instance in which at least one of the attribute values is partially specified (or *partially missing*). Relative to the AVT shown in Fig. 1, the instance (*Senior*, *TA*) is a fully specified instance. Some examples of partially specified instances are (*Undergraduate*, *RA*), (*Freshman*, *Government*), (*Graduate*, *Off-Campus*). The conventional missing value (normally recorded as ?) is a special case of partially specified attribute value, whose attribute value corresponds to the root of its AVT and contains no descriptive information about that attribute. We call this kind of missing *totally missing*.

**Definition 2.7 (A Partially Specified Data Set)** *A partially specified data set,* $D_T$ *(relative to a set* $T$ *of attribute value taxonomies), is a collection of instances drawn from* $I_T$, *where each instance is labeled with the appropriate class label from* $C = \{c_1, c_2, \ldots, c_M\}$, *a finite set of mutually disjoint classes. Thus,* $D_T \subseteq I_T \times C$.

## 2.4 Learning classifiers from data

The problem of learning classifiers from AVT and data is a natural generalization of the problem of learning classifiers from data without AVT. The original data set $D$ is simply a collection of labeled instances of the form $(X_p, c_{X_p})$, where $X_p \in I$ and $c_{X_p} \in C$ is a class label. A classifier is a hypothesis in the form of a function

$h : I \rightarrow C$, whose domain is the instance space $I$ and whose range is the set of classes $C$. An hypothesis space $H$ is a set of hypotheses that can be represented in some hypothesis language or by a parameterised family of functions (e.g. decision trees, naïve Bayes classifiers, SVM, etc.). The task of learning classifiers from the original data set $D$ entails identifying a hypothesis $h \in H$ that satisfies some criteria (e.g. a hypothesis that is most likely given the training data $D$).

The problem of learning classifiers from AVT and data can be stated as follows:

**Definition 2.8 (Learning Classifiers from AVT and Data)** *Given a user-supplied set of AVTs, $T$, and a data set, $D_T$, of (possibly) partially specified labeled instances, construct a classifier $h_T : I_T \rightarrow C$ for assigning appropriate class labels to each instance in the instance space $I_T$.*

Of special interest are the cases in which the resulting hypothesis space $H_T$ has structure that makes it possible to search it efficiently for a hypothesis that is both concise as well as accurate.

## 3 AVT-based naïve Bayes learner

### 3.1 Naïve Bayes learner (NBL)

Naïve Bayes classifier is a simple and yet effective classifier that has competitive performance with other more sophisticated classifiers [18]. Naïve Bayes classifier operates under the assumption that each attribute is independent of others given the class. Thus, the joint probability given a class can be written as the product of individual class conditional probabilities for each attribute. The Bayesian approach to classifying an instance $X_p = (v_{1p}, v_{2p}, \ldots, v_{Np})$ is to assign it the most probable class $c_{\text{MAP}}(X_p)$:

$$\begin{aligned} c_{\text{MAP}}(X_p) &= \underset{c_j \in C}{\text{argmax}}\, P(v_{1p}, v_{2p}, \ldots, v_{Np}|c_j)p(c_j) \\ &= \underset{c_j \in C}{\text{argmax}}\, p(c_j) \prod_i P(v_{ip}|c_j). \end{aligned}$$

The task of the naïve Bayes Learner (NBL) is to estimate $\forall c_j \in C$ and $\forall v_{i_k} \in \text{dom}(A_i)$, relevant class probabilities, $p(c_j)$, and the class conditional probabilities, $P(v_{i_k}|c_j)$, from training data, $D$. These probabilities, which completely specify a naïve Bayes classifier, can be estimated from a training set, $D$, using standard probability estimation methods [31] based on relative frequency counts of the corresponding classes and attribute value and class label co-occurrences observed in $D$. We denote $\sigma_i(v_k|c_j)$ as the frequency count of value $v_k$ of attribute $A_i$ given class label $c_j$ and $\sigma(c_j)$ as the frequency count of class label $c_j$ in a training set $D$. Hence, these relative frequency counts completely summarize the information needed for constructing a naïve Bayes classifier from $D$, and they constitute *sufficient statistics* for naïve Bayes learner [9, 10].

### 3.2 AVT-NBL

We now introduce AVT-NBL, an algorithm for learning naïve Bayes classifiers from AVT and data. Given an ordered set of AVTs, $T = \{\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_N\}$, corresponding to the attributes $A = \{A_1, A_2, \ldots, A_N\}$ and a data set $D = \{(X_p, c_{X_p})\}$ of labeled examples of the form $(X_p, c_{X_p})$, where $X_p \in I_T$ is a partially or fully specified instance and $c_{X_p} \in C$ is the corresponding class label, the task of AVT-NBL is to construct a naïve Bayes classifier for assigning $X_p$ to its most probable class, $c_{\text{MAP}}(X_p)$. As in the case of NBL, we assume that each attribute is independent of the other attributes given the class.

Let $\Gamma = \{\gamma_1, \gamma_2, \ldots, \gamma_N\}$ be a global cut, where $\gamma_i$ stands for a cut through $\mathcal{T}_i$. A naïve Bayes classifier defined on the instance space $I_\Gamma$ is completely specified by a set of class conditional probabilities for each value of each attribute. Suppose we denote the table of class conditional probabilities associated with values in $\gamma_i$ by $CPT(\gamma_i)$. Then the naïve Bayes classifier defined over the instance space $I_\Gamma$ is specified by $h(\Gamma) = \{CPT(\gamma_1), CPT(\gamma_2), \ldots, CPT(\gamma_N)\}$.

If each cut, $\gamma_i \in \Gamma_0$ is chosen to correspond to the primitive values of the respective attribute, i.e. $\forall i\ \gamma_i = Leaves(\mathcal{T}_i)$. $h(\Gamma_0)$ is simply the standard naïve Bayes classifier based on the attributes $A_1, A_2, \ldots, A_N$. If each cut $\gamma_i \in \Gamma$ is chosen to pass through the root of each AVT, i.e. $\forall i\ \gamma_i = \{\text{Root}(\mathcal{T}_i)\}$, $h(\Gamma)$ simply assigns each instance to the class that is a priori most probable.

AVT-NBL starts with the naïve Bayes classifier that is based on the most abstract value of each attribute (the most general hypothesis in $H_T$) and successively refines the classifier (hypothesis) using a criterion that is designed to trade off between the accuracy of classification and the complexity of the resulting naïve Bayes classifier. Successive refinements of $\Gamma$ correspond to a partial ordering of naïve Bayes classifiers based on the structure of the AVTs in $T$.

For example, in Fig. 2, $\hat{\Gamma}$ is a cut refinement of $\Gamma$, and hence corresponding hypothesis $h(\hat{\Gamma})$ is a refinement of $h(\Gamma)$. Relative to the two cuts, Table 1 shows the conditional probability tables that we need to compute during learning for $h(\Gamma)$ and $h(\hat{\Gamma})$, respectively (assuming $C = \{+, -\}$ as two possible class labels). From the class conditional probability table, we can count the number of class conditional probabilities needed to specify the corresponding naïve Bayes classifier. As shown in Table 1, the total number of class conditional probabilities for $h(\Gamma)$ and $h(\hat{\Gamma})$ are 8 and 10, respectively.

#### *3.2.1 Class conditional frequency counts*

Given an attribute value taxonomy $\mathcal{T}_i$ for attribute $A_i$, we can define a tree of class conditional frequency counts $CCFC(\mathcal{T}_i)$ such that there is a one-to-one correspondence between the nodes of the AVT $\mathcal{T}_i$ and the nodes of the corresponding $CCFC(\mathcal{T}_i)$. It follows that the class conditional frequency counts associated with a nonleaf node of $CCFC(\mathcal{T}_i)$ should correspond to the aggregation of the corresponding class conditional frequency counts associated with its children. Because each cut through an AVT $\mathcal{T}_i$ corresponds to a partition of the set of possible values in $Nodes(\mathcal{T}_i)$ of the attribute $A_i$, the corresponding cut $\gamma_i$ through $CCFC(\mathcal{T}_i)$ specifies a valid class conditional probability table $CPT(\gamma_i)$ for the attribute $A_i$.

**Table 1** Conditional probability tables. This table shows the entries of conditional probability tables associated with two global cut $\Gamma$ and $\hat{\Gamma}$ shown in Fig. 2 (assuming C = +, −).

| Value | Pr(+) | Pr(−) |
|---|---|---|
| CPT for $h(\Gamma)$ | | |
| Undergraduate | $P(Undergraduate\|+)$ | $P(Undergraduate\|-)$ |
| Graduate | $P(Graduate\|+)$ | $P(Graduate\|-)$ |
| On-Campus | $P(On\text{-}Campus\|+)$ | $P(On\text{-}Campus\|-)$ |
| Off-Campus | $P(Off\text{-}Campus\|+)$ | $P(Off\text{-}Campus\|-)$ |
| CPT for $h(\hat{\Gamma})$ | | |
| Undergraduate | $P(Undergraduate\|+)$ | $P(Undergraduate\|-)$ |
| Master | $P(Master\|+)$ | $P(Master\|-)$ |
| PhD | $P(Ph.D.\|+)$ | $P(PhD\|-)$ |
| On-Campus | $P(On\text{-}Campus\|+)$ | $P(On\text{-}Campus\|-)$ |
| Off-Campus | $P(Off\text{-}Campus\|+)$ | $P(Off\text{-}Campus\|-)$ |

In the case of numerical attributes, AVTs are defined over intervals based on observed values for the attribute in the data set. Each cut through the AVT corresponds to a partition of the numerical attribute into a set of intervals. We calculate the class conditional probabilities for each interval. As in the case of nominal attributes, we define a tree of class conditional frequency counts $CCFC(\mathcal{T}_i)$ for each numerical attribute $A_i$. $CCFC(\mathcal{T}_i)$ is used to calculate the conditional probability table $CPT(\gamma_i)$ corresponding to a cut $\gamma_i$.

When all of the instances in the data set $D$ are fully specified, estimation of $CCFC(\mathcal{T}_i)$ for each attribute is straightforward: we simply estimate the class conditional frequency counts associated with each of the primitive values of $A_i$ from the data set $D$ and use them recursively to compute the class conditional frequency counts associated with the nonleaf nodes of $CCFC(\mathcal{T}_i)$.

When some of the data are partially specified, we can use a two-step process for computing $CCFC(\mathcal{T}_i)$: First we make an upward pass aggregating the class conditional frequency counts based on the specified attribute values in the data set. Then we propagate the counts associated with partially specified attribute values down through the tree, augmenting the counts at lower levels according to the distribution of values along the branches based on the subset of the data for which the corresponding values are fully specified[1].

Let $\sigma_i(v|c_j)$ be the frequency count of value $v$ of attribute $A_i$ given class label $c_j$ in a training set $D$ and $p_i(v|c_j)$ the estimated class conditional probability of value $v$ of attribute $A_i$ given class label $c_j$ in a training set $D$. Let $\pi(v, \mathcal{T}_i)$ be the set of all children (direct descendants) of a node with value $v$ in $\mathcal{T}_i$; $\Lambda(v, \mathcal{T}_i)$ the list of ancestors, including the root, for $v$ in $\mathcal{T}_i$. The procedure of computing $CCFC(\mathcal{T}_i)$ is shown below.

**Algorithm 3.1** *Calculating class conditional frequency counts.*

**Input:** *Training data D and* $\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_N$.

[1] This procedure can be seen as a special case of EM (expectation maximization) algorithm [15] to estimate sufficient statistics for $CCFC(\mathcal{T}_i)$.

**Output:** $CCFC(\mathcal{T}_1), CCFC(\mathcal{T}_2), \ldots, CCFC(\mathcal{T}_N)$.

**Step 1:** *Calculate frequency counts* $\sigma_i(v|c_j)$ *for each node* $v$ *in* $\mathcal{T}_i$ *using the class conditional frequency counts associated with the specified values of attribute* $A_i$ *in training set* $D$.
**Step 2:** *For each attribute value* $v$ *in* $\mathcal{T}_i$ *that received nonzero counts as a result of step 1, aggregate the counts upward from each such node* $v$ *to its ancestors,* $\Lambda(v, \mathcal{T}_i)$*:* $\sigma_i(w|c_j)_{w\in\Lambda(v,\mathcal{T}_i)} \leftarrow \sigma_i(w|c_j) + \sigma_i(v|c_j)$.
**Step 3:** *Starting from the root, recursively propagate the counts corresponding to partially specified instances at each node* $v$ *downward according to the observed distribution among its children to obtain updated counts for each child* $u_l \in \pi(v, \mathcal{T}_i)$*:*

$$\sigma_i(u_l|c_j) = \begin{cases} \left(\frac{\sigma_i(v|c_j)}{|\pi(v,\mathcal{T}_i)|}\right) & \text{If } \sum_{k=1}^{|\pi(v,\mathcal{T}_i)|} \sigma_i(u_k|c_j)=0, \\ \sigma_i(u_l|c_j)\left(1 + \frac{\sigma_i(v|c_j) - \sum_{k=1}^{|\pi(v,\mathcal{T}_i)|} \sigma_i(u_k|c_j)}{\sum_{k=1}^{|\pi(v,\mathcal{T}_i)|} \sigma_i(u_k|c_j)}\right) & \text{Otherwise.} \end{cases}$$

We use a simple example to illustrate the estimation of class conditional frequency counts when some of the instances are partially specified. On the AVT for *student status* shown in Fig. 3(A), we mark each attribute value with a count showing the total number of positively labeled (+) instances having that specific value. First, we aggregate the counts upward from each node to its ancestors. For example, in Fig. 3(B), the four counts 10, 20, 5, 15 on primitive attribute values *Freshman*, *Sophomore*, *Junior*, and *Senior* add up to 50 as the count for *Undergraduate*. Because we also have 15 instances that are partially specified with the value *Undergraduate*, the two counts (15 and 50) aggregate again toward the root. Next, we distribute the counts of a partially specified attribute value downward according to the distributions of values among their descendant nodes. For example, 15, the count of partially specified attribute value *Undergraduate*, is propagated down into fractional counts 3, 6, 1.5, 4.5 for *Freshman*, *Sophomore*, *Junior* and *Senior* (see Fig. 3(C) for values in parentheses). Finally, we update the estimated frequency counts for all attribute values as shown in Fig. 3(D).

Now that we have estimated the class conditional frequency counts for all attribute value taxonomies, we can calculate the conditional probability table with regard to any global cut $\Gamma$. Let $\Gamma = \{\gamma_1, \ldots, \gamma_N\}$ be a global cut, where $\gamma_i$ stands for a cut through $CCFC(\mathcal{T}_i)$. The estimated conditional probability table $CPT(\gamma_i)$ associated with the cut $\gamma_i$ can be calculated from $CCFC(\mathcal{T}_i)$ using Laplace estimates [31, 24].

$$p_i(v|c_j)_{v\in\gamma_i} \leftarrow \frac{1/|D| + \sigma_i(v|c_j)}{|\gamma_i|/|D| + \sum_{u\in\gamma_i} \sigma_i(u|c_j)}$$

Recall that the naïve Bayes classifier $h(\Gamma)$ based on a chosen global cut $\Gamma$ is completely specified by the conditional probability tables associated with the cuts in $\Gamma$: $h(\Gamma) = \{CPT(\gamma_1), \ldots, CPT(\gamma_N)\}$.

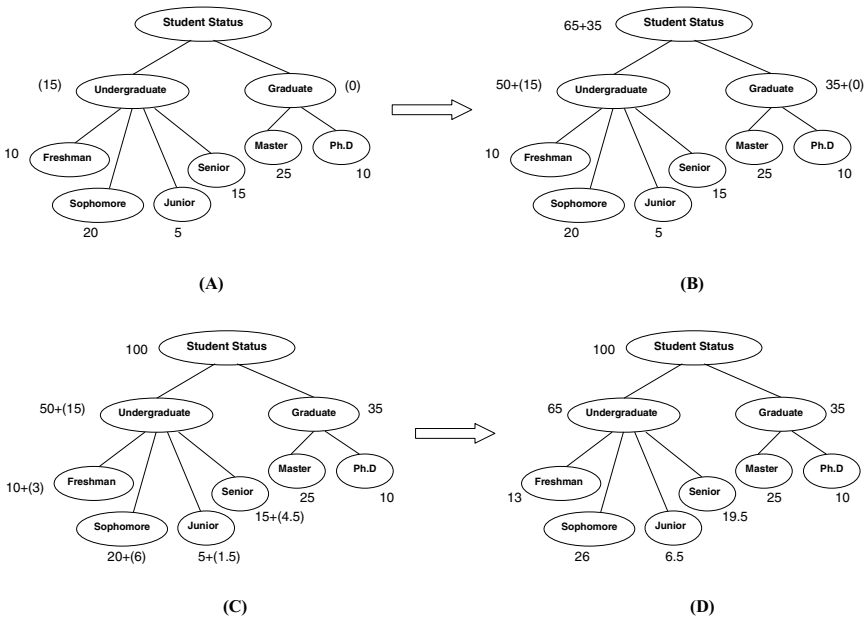



**Fig. 3** Estimation of class conditional frequency counts. (A) Initial counts associated with each attribute value showing the number of positively labeled instances. (B) Aggregation of counts upwards from each node to its ancestors. (C) Distribution of counts of a partially specified attribute value downwards among descendant nodes. (D) Updating the estimated frequency counts for all attribute values.

### *3.2.2 Searching for a compact naïve Bayes classifier*

The scoring function that we use to evaluate a candidate AVT-guided refinement of a naïve Bayes classifier is based on a variant of the minimum description length (MDL) score [37], which provides a basis for trading off the complexity against the error of the model. MDL score captures the intuition that the goal of a learner is to compress the training data $D$ and encode it in the form of a hypothesis or a model $h$ so as to minimize the length of the message that encodes the model $h$ and the data $D$ given the model $h$. [19] suggested the use of a conditional MDL (CMDL) score in the case of hypotheses that are used for classification (as opposed to modelling the joint probability distribution of a set of random variables) to capture this tradeoff. In general, computation of CMDL score is not feasible for Bayesian networks with arbitrary structure [19]. However, in the case of naïve Bayes classifiers induced by a set of AVT, as shown below, it is possible to efficiently calculate the CMDL score.

$$CMDL(h|D) = \left(\frac{\log|D|}{2}\right) size(h) - CLL(h|D)$$

$$where, CLL(h|D) = |D| \sum_{p=1}^{|D|} \log P_h(c_{X_p}|v_{1p}, \ldots, v_{Np})$$

Here, $P_h(c_{X_p}|v_{1p}, \ldots, v_{Np})$ denotes the conditional probability assigned to the class $c_{X_p} \in C$ associated with the training sample $X_p = (v_{1p}, v_{2p}, \ldots, v_{Np})$ by

the classifier $h$, $size(h)$ is the number of parameters used by $h$, $|D|$ the size of the data set, and $CLL(h|D)$ is the conditional log likelihood of the hypothesis $h$ given the data $D$. In the case of a naïve Bayes classifier $h$, $size(h)$ corresponds to the total number of class conditional probabilities needed to describe $h$. Because each attribute is assumed to be independent of the others given the class in a naïve Bayes classifier, we have

$$CLL(h|D) = |D| \sum_{p=1}^{|D|} \log \left( \frac{P(cx_p) \prod_i P_h(v_{ip}|cx_p)}{\sum_{j=1}^{|C|} P(c_j) \prod_i P_h(v_{ip}|c_j)} \right),$$

where $P(c_j)$ is the prior probability of the class $c_j$, which can be estimated from the observed class distribution in the data $D$.

It is useful to distinguish between two cases in the calculation of the conditional likelihood $CLL(h|D)$ when $D$ contains partially specified instances: (1) When a partially specified value of attribute $A_i$ for an instance lies on the cut $\gamma$ through $CCFC(\mathcal{T}_i)$ or corresponds to one of the descendants of the nodes in the cut. In this case, we can treat that instance as though it were fully specified relative to the naïve Bayes classifier based on the cut $\gamma$ of $CCFC(\mathcal{T}_i)$ and use the class conditional probabilities associated with the cut $\gamma$ to calculate its contribution to $CLL(h|D)$. (2) When a partially specified value (say $v$) of $A_i$ is an ancestor of a subset (say $\lambda$ ) of the nodes in $\gamma$. In this case, $p(v|c_j) = \sum_{u_i \in \lambda} p(u_i|c_j)$, such that we can aggregate the class conditional probabilities of the nodes in $\lambda$ to calculate the contribution of the corresponding instance to $CLL(h|D)$.

Because each attribute is assumed to be independent of others given the class, the search for the AVT-based naïve Bayes classifier (AVT-NBC) can be performed efficiently by optimizing the criterion independently for each attribute. This results in a hypothesis $h$ that intuitively trades off the complexity of naïve Bayes classifier (in terms of the number of parameters used to describe the relevant class conditional probabilities) against accuracy of classification. The algorithm terminates when none of the candidate refinements of the classifier yield statistically significant improvement in the CMDL score. The procedure is outlined below.

---

**Algorithm 3.2** *Searching for compact AVT-based naïve Bayes classifier.*

**Input:** *Training data* $D$ *and* $CCFC(\mathcal{T}_1), CCFC(\mathcal{T}_2), \ldots, CCFC(\mathcal{T}_N)$.
**Output:** *A naïve Bayes classifier trading off the complexity against the error.*

1. *Initialize each* $\gamma_i$ *in* $\Gamma = \{\gamma_1, \gamma_2, \ldots, \gamma_N\}$ *to* $\{\text{Root}(\mathcal{T}_i)\}$.
2. *Estimate probabilities that specify the hypothesis* $h(\Gamma)$.
3. *For each cut* $\gamma_i$ *in* $\Gamma = \{\gamma_1, \gamma_2, \ldots, \gamma_N\}$*:*
   A. *Set* $\delta_i \leftarrow \gamma_i$
   B. *Until there are no updates to* $\gamma_i$
      i. *For each* $v \in \delta_i$
         a. *Generate a refinement* $\gamma_i^v$ *of* $\gamma_i$ *by replacing* $v$ *with* $\pi(v, \mathcal{T}_i)$*, and refine* $\Gamma$ *accordingly to obtain* $\hat{\Gamma}$. *Construct corresponding hypothesis* $h(\hat{\Gamma})$

*b. If* $CMDL(h(\hat{\Gamma})|D) < CMDL(h(\Gamma)|D)$*, replace* $\Gamma$ *with* $\hat{\Gamma}$ *and* $\gamma_i$ *with* $\gamma_i^v$

*ii.* $\delta_i \leftarrow \gamma_i$

*4. Output* $h(\Gamma)$

## 4 Alternative approaches to learning classifiers from AVT and data

Besides AVT-NBL, we can envision two alternative approaches to learning classifiers from AVT and data.

### 4.1 Approaches that treat partially specified attribute values as if they were totally missing

Each partially specified (and hence partially missing) attribute value is treated as if it were totally missing, and the resulting data set with missing attribute values is handled using standard approaches for dealing with missing attribute values in learning classifiers from an otherwise fully specified data set in which some attribute values are missing in some of the instances values. A main advantage of this approach is that it requires no modification to the learning algorithm. All that is needed is a simple preprocessing step in which all partially specified attribute values are turned into missing attribute values.

### 4.2 AVT-based propositionalisation methods

The data set is represented using a set of Boolean attributes obtained from $Nodes(\mathcal{T}_i)$ of attribute $A_i$ by associating a Boolean attribute with each node (except the root) in $\mathcal{T}_i$. Thus, each instance in the original data set defined using $N$ attributes is turned into a Boolean instance specified using $\tilde{N}$ Boolean attributes, where $\tilde{N} = \sum_{i=1}^{N}(|Nodes(\mathcal{T}_i)| - 1)$.

In the case of the *student status* taxonomy shown in Fig. 1, this would result in binary features that correspond to the propositions such as (student = *Undergraduate*), (student = *Graduate*), (student = *Freshman*), ... (student = *Senior*), (student = *Master*), (student = *PhD*). Based on the specified value of an attribute in an instance, e.g. (student = *Master*), the values of its ancestors in the AVT (e.g. student = *Graduate*) are set to True because the AVT asserts that *Master* students are also *Graduate* students. But the Boolean attributes that correspond to descendants of the specified attribute value are treated as unknown. For example, when the value of the *student status* attribute is partially specified in an instance, e.g. (student = *Graduate*), the corresponding Boolean attribute is set to True, but the Boolean attributes that correspond to the descendants of *Graduate* in this taxonomy are treated as *missing*. The resulting data with some missing attribute values can be handled using standard approaches to dealing with missing attribute values. For numerical attributes, the Boolean attributes are the intervals that correspond

to nodes of the respective AVTs. If a numerical value falls in a certain interval, the corresponding Boolean attribute is set to True, otherwise it is set to False. We call the resulting algorithm—NBL applied to AVT-based propositionalized version of the data—Prop-NBL.

Note that the Boolean features created by the propositionalisation technique described above are not independent given the class. A Boolean attribute that corresponds to any node in an AVT is necessarily correlated with Boolean attributes that correspond to its descendants as well as its ancestors in the tree. For example, the Boolean attribute (student = *Graduate*) is correlated with (student = *Master*). (Indeed, it is this correlation that enables us to exploit the information provided by AVT in learning from partially specified data). Thus, a naïve Bayes classifier that would be optimal in the maximal a posteriori sense [28] when the original attributes *student status* and *work status* are independent given class would no longer be optimal when the new set of Boolean attributes are used because of strong dependencies among the Boolean attributes derived from an AVT.

A main advantage of the AVT-based propositionalisation methods is that they require no modification to the learning algorithm. However, it does require preprocessing of partially specified data using the information supplied by an AVT. The number of attributes in the transformed data set is substantially larger than the number of attributes in the original data set. More important, the statistical dependence among the Boolean attributes in the propositionalised representation of the original data set can degrade the performance of classifiers, e.g. naïve Bayes that rely on independence of attributes given class. Against this background, we experimentally compare AVT-NBL with Prop-NBL and the standard naïve Bayes algorithm (NBL).

## 5 Experiments and results

### 5.1 Experiments

Our experiments were designed to explore the performance of AVT-NBL relative to that of NBL and PROP-NBL.

Although partially specified data and hierarchical AVT are common in many application domains, at present, there are few standard benchmark data sets of partially specified data and the associated AVT. We select 37 data sets from the UC Irvine Machine Learning Repository, among which 8 data sets use only nominal attributes and 29 data sets have both nominal attributes and numerical attributes. Every numerical attribute in the 29 data sets has been discretised into a maximum of 10 bins. For only three of the data sets (i.e. *Mushroom*, *Soybean*, and *Nursery*), AVTs were supplied by domain experts. For the remaining data sets, no expert-generated AVTs are readily available. Hence, the AVTs on both nominal and numerical attributes were generated using AVT-Learner, a hierarchical agglomerative clustering algorithm to construct AVTs for learning [23].

The first set of experiments compares the performance of AVT-NBL, NBL, and PROP-NBL on the original data.

The second set of experiments explores the performance of the algorithms on data sets with different percentages of totally missing and partially missing attribute values. Three data sets with a prespecified percentage (10%, 30% or 50%,

*excluding* the missing values in the original data set) of totally or partially missing attribute values were generated by assuming that the missing values are uniformly distributed on the nominal attributes [45]. From the original data set $D$, a data set $D_p$ of partially (or totally) missing values was generated as follows: Let $(n_l, n_{l-1}, \ldots, n_0)$ be the path from the fully specified primitive value $n_l$ to the root $n_0$ of the corresponding AVT. select one of the nodes (excluding $n_l$) along this path with uniform probability. Read the corresponding attribute value from the AVT and assign it as the partially specified value of the corresponding attribute. Note that the selection of the root of the AVT would result in a totally missing attribute value.

In each case, the error rate and the size (as measured by the number of class conditional probabilities used to specify the learned classifier) were estimated using 10-fold cross-validation, and we calculate 90% confidence interval on the error rate.

A third set of experiments were designed to investigate the performance of classifiers generated by AVT-NBL, Prop-NBL and NBL as a function of the training-set size. We divided each data set into two disjoint parts: a training pool and a test pool. Training sets of different sizes, corresponding to 10%, 20%, ..., 100% of the training pool, were sampled and used to train naïve Bayes classifiers using AVT-NBL, Prop-NBL, and NBL. The resulting classifiers were evaluated on the entire test pool. The experiment was repeated 9 times for each training-set size. The entire process was repeated using 3 different random partitions of data into training and test pools. The accuracy of the learned classifiers on the examples in the test pool were averaged across the $9 \times 3 = 27$ runs.

## 5.2 Results

### *5.2.1 AVT-NBL yields lower error rates than NBL and PROP-NBL on the original fully specified data*

Table 2 shows the estimated error rates of the classifiers generated by the AVT-NBL, NBL and PROP-NBL on 37 UCI benchmark data sets. According to the results, the error rate of AVT-NBL is substantially smaller than that of NBL and PROP-NBL. It is worth noting that PROP-NBL (NBL applied to a transformed data set using Boolean features that correspond to nodes of the AVTs) generally produces classifiers that have higher error rates than NBL. This can be explained by the fact that the Boolean features generated from an AVT are generally not independent given the class.

### *5.2.2 AVT-NBL yields classifiers that are substantially more compact than those generated by PROP-NBL and NBL*

The shaded columns in Table 2 compare the total number of class conditional probabilities needed to specify the classifiers produced by AVT-NBL, NBL, and PROP-NBL on original data. The results show that AVT-NBL is effective in exploiting the information supplied by the AVT to generate accurate yet compact classifiers. Thus, AVT-guided learning algorithms offer an

**Table 2** Comparison of error rate and size of classifiers generated by NBL, PROP-NBL and AVT-NBL on 37 UCI benchmark data. The error rates and the sizes were estimated using 10-fold cross-validation. We calculate 90% confidence interval on the error rates. The size of the classifiers for each data set is constant for NBL and Prop-NBL, and for AVT-NBL, the size shown represents the average across the 10 cross-validation experiments.

| | NBL | | Prop-NBL | | AVT-NBL | |
|---|---|---|---|---|---|---|
| Data Set | Error | Size | Error | Size | Error | Size |
| Anneal | 6.01 (±1.30) | 954 | 10.69 (±1.69) | 2886 | 1.00 (±0.55) | 666 |
| Audiology | 26.55 (±5.31) | 3696 | 27.87 (±5.39) | 8184 | 23.01 (±5.06) | 3600 |
| Autos | 22.44 (±4.78) | 1477 | 21.46 (±4.70) | 5187 | 13.17 (±3.87) | 805 |
| Balance-scale | 8.64 (±1.84) | 63 | 11.52 (±2.09) | 195 | 8.64 (±1.84) | 60 |
| Breast-cancer | 28.32 (±4.82) | 84 | 27.27 (±4.76) | 338 | 27.62 (±4.78) | 62 |
| Breast-w | 2.72 (±1.01) | 180 | 2.86 (±1.03) | 642 | 2.72 (±1.01) | 74 |
| Car | 14.47 (±1.53) | 88 | 15.45 (±1.57) | 244 | 13.83 (±1.50) | 80 |
| Colic | 17.93 (±3.28) | 252 | 20.11 (±3.43) | 826 | 16.58 (±3.18) | 164 |
| Credit-a | 14.06 (±2.17) | 204 | 18.70 (±2.43) | 690 | 13.48 (±2.13) | 124 |
| Credit-g | 24.50 (±2.23) | 202 | 26.20 (±2.28) | 642 | 24.60 (±2.23) | 154 |
| Dermatology | 2.18 (±1.38) | 876 | 1.91 (±1.29) | 2790 | 2.18 (±1.38) | 576 |
| Diabetes | 22.53(+2.47) | 162 | 25.65 (±2.58) | 578 | 22.01 (±2.45) | 108 |
| Glass | 22.90(+4.71) | 637 | 28.04 (±5.04) | 2275 | 19.16 (±4.41) | 385 |
| Heart-c | 14.19 (±3.29) | 370 | 16.50 (±3.50) | 1205 | 12.87 (±3.16) | 210 |
| Heart-h | 13.61 (±3.28) | 355 | 14.97 (±3.41) | 1155 | 13.61 (±3.28) | 215 |
| Heart-statlog | 16.30 (±3.69) | 148 | 16.30 (±3.69) | 482 | 13.33 (±3.39) | 78 |
| Hepatitis | 10.97 (±4.12) | 174 | 9.03 (±3.78) | 538 | 7.10 (±3.38) | 112 |
| Hypothyroid | 4.32 (±0.54) | 436 | 6.68 (±0.67) | 1276 | 4.22 (±0.54) | 344 |
| Ionosphere | 7.98 (±2.37) | 648 | 8.26 (±2.41) | 2318 | 5.41 (±1.98) | 310 |
| Iris | 4.00 (±2.62) | 123 | 4.67 (±2.82) | 435 | 5.33 (±3.01) | 90 |
| Kr-vs-kp | 12.11 (±0.95) | 150 | 12.20 (±0.95) | 306 | 12.08 (±0.95) | 146 |
| Labor | 8.77 (±6.14) | 170 | 10.53 (±6.67) | 546 | 10.53 (±6.67) | 70 |
| Letter | 27.17 (±0.52) | 4186 | 34.40 (±0.55) | 15002 | 29.47 (±0.53) | 2652 |
| Lymph | 14.19 (±4.70) | 240 | 18.24 (±5.21) | 660 | 15.54 (±4.88) | 184 |
| Mushroom | 4.43 (±1.30) | 252 | 4.45 (±1.30) | 682 | 0.14 (±0.14) | 202 |
| Nursery | 9.67 (±1.48) | 135 | 10.59 (±1.54) | 355 | 9.67 (±1.48) | 125 |
| Primary-tumor | 49.85 (±4.45) | 836 | 52.51 (±4.45) | 1782 | 52.21 (±4.45) | 814 |
| Segment | 10.91 (±1.06) | 1183 | 11.86 (±1.10) | 4193 | 10.00 (±1.02) | 560 |
| Sick | 2.52 (±0.42) | 218 | 4.51 (±0.55) | 638 | 2.17 (±0.39) | 190 |
| Sonar | 0.96 (±1.11) | 1202 | 0.96 (±1.11) | 4322 | 0.48 (±0.79) | 312 |
| Soybean | 7.03 (±1.60) | 1900 | 8.19 (±1.72) | 4959 | 5.71 (±1.45) | 1729 |
| Splice | 4.64 (±0.61) | 864 | 4.08 (±0.57) | 2727 | 4.23 (±0.58) | 723 |
| Vehicle | 33.33 (±2.66) | 724 | 32.98 (±2.65) | 2596 | 32.15 (±2.63) | 368 |
| Vote | 9.89 (±2.35) | 66 | 9.89 (±2.35) | 130 | 9.89 (±2.35) | 64 |
| Vowel | 64.24 (±2.50) | 1320 | 63.33 (±2.51) | 4675 | 57.58 (±2.58) | 1122 |
| Waveform-5000 | 35.96 (±1.11) | 1203 | 36.38 (±1.12) | 4323 | 34.92 (±1.11) | 825 |
| Zoo | 6.93 (±4.57) | 259 | 5.94 (±4.25) | 567 | 3.96 (±3.51) | 245 |

approach to compressing class conditional probability distributions that are different from the statistical independence-based factorization used in Bayesian networks.

### *5.2.3 AVT-NBL yields significantly lower error rates than NBL and PROP-NBL on partially specified data and data with totally missing values*

Table 3 compares the estimated error rates of AVT-NBL with that of NBL and PROP-NBL in the presence of varying percentages (10%, 30% and 50%)

**Table 3** Comparison of error rates on data with 10%, 30% and 50% partially or totally missing values. The error rates were estimated using 10-fold cross-validation, and we calculate 90% confidence interval on each error rate.

| Data | | Partially missing | | | Totally missing | | |
|---|---|---|---|---|---|---|---|
| Methods | | NBL | Prop-NBL | AVT-NBL | NBL | Prop-NBL | AVT-NBL |
| Mushroom | 10% | 4.65 (±1.33) | 4.69 (±1.34) | 0.30 (±0.30) | 4.65 (±1.33) | 4.76 (±1.35) | 1.29 (±071) |
| | 30% | 5.28 (±1.41) | 4.84 (±1.36) | 0.64 (±0.50) | 5.28 (±1.41) | 5.37 (±1.43) | 2.78 (±1.04) |
| | 50% | 6.63 (±1.57) | 5.82 (±1.48) | 1.24 (±0.70) | 6.63 (±1.57) | 6.98 (±1.61) | 4.61 (±1.33) |
| Nursery | 10% | 15.27 (±1.81) | 15.50 (±1.82) | 12.85 (±1.67) | 15.27 (±1.81) | 16.53 (±1.86) | 13.24 (±1.70) |
| | 30% | 26.84 (±2.23) | 26.25 (±2.21) | 21.19 (±2.05) | 26.84 (±2.23) | 27.65 (±2.24) | 22.48 (±2.09) |
| | 50% | 36.96 (±2.43) | 35.88 (±2.41) | 29.34 (±2.29) | 36.96 (±2.43) | 38.66 (±2.45) | 32.51 (±235) |
| Soybean | 10% | 8.76 (±1.76) | 9.08 (±1.79) | 6.75 (±1.57) | 8.76 (±1.76) | 9.09 (±1.79) | 6.88 (±1.58) |
| | 30% | 12.45 (±2.07) | 11.54 (±2.00) | 10.32 (±1.90) | 12.45 (±2.07) | 12.31 (±2.05) | 10.41 (±1.91) |
| | 50% | 19.39 (±2.47) | 16.91 (±234) | 16.93 (±234) | 19.39 (±2.47) | 19.59 (±2.48) | 17.97 (±2.40) |

of partially missing attribute values and totally missing attribute values. Naïve Bayes classifiers generated by AVT-NBL have substantially lower error rates than those generated by NBL and PROP-NBL, with the differences being more pronounced at higher percentages of partially (or totally) missing attribute values.

### *5.2.4 AVT-NBL produces more accurate classifiers than NBL and Prop-NBL for a given training set size*

Figure 4 shows the plot of the accuracy of the classifiers learned as a function of training set size for *Audiology* data. We obtained similar results on other benchmark data sets used in this study. Thus, AVT-NBL is *more efficient* than NBL and Prop-NBL in its use of training data.

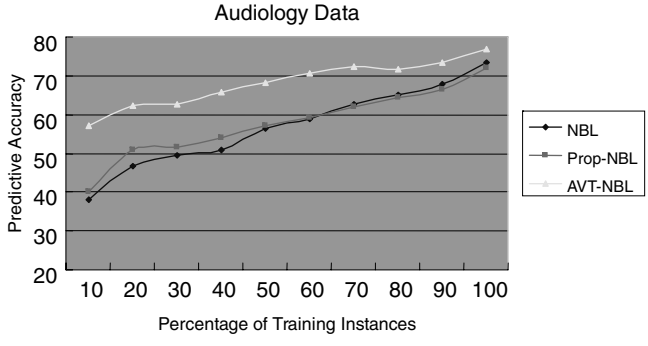



**Fig. 4** Classifier accuracy as a function of training set size on audiology data by AVT-NBL, Prop-NBL and NBL, respectively. Note that the $X$ axis shows the percentage of training instances that has been sampled in training the naïve Bayes classifier, and the $Y$ axis shows the predictive accuracy in percentage.

## 6 Summary and discussion

### 6.1 Summary

In this paper, we have described AVT-NBL[2], an algorithm for learning classifiers from attribute value taxonomies (AVT) and data in which different instances may have attribute values specified at different levels of abstraction. AVT-NBL is a natural generalization of the standard algorithm for learning naïve Bayes classifiers. The standard naïve Bayes learner (NBL) can be viewed as a special case of AVT-NBL by collapsing a multilevel AVT associated with each attribute into a corresponding single-level AVT whose leaves correspond to the primitive values of the attribute.

Our experimental results presented in this paper show that:

1. AVT-NBL is able to learn substantially compact and more accurate classifiers on a broad range of data sets than those produced by standard NBL and Prop-NBL (applying NBL to data with an augmented set of Boolean attributes).
2. When applied to data sets in which attribute values are partially specified or totally missing, AVT-NBL can yield classifiers that are more accurate and compact than those generated by NBL and Prop-NBL.
3. AVT-NBL is more efficient in its use of training data. AVT-NBL produces classifiers that outperform those produced by NBL using substantially fewer training examples.

Thus, AVT-NBL offers an effective approach to learning compact (hence more comprehensible) accurate classifiers from data—including data that are *partially specified*. AVT-guided learning algorithms offer a promising approach to knowledge acquisition from autonomous, semantically heterogeneous information sources, where domain-specific AVTs are often available and data are often partially specified.

### 6.2 Related work

There is some work in the machine-learning community on the problem of learning classifiers from attribute value taxonomies (sometimes called tree-structured attributes) and fully specified data in the case of decision trees and rules. [32] outlined an approach to using ISA hierarchies in decision-tree learning to minimize misclassification costs. [36] mentions handling of tree-structured attributes as a desirable extension to C4.5 decision-tree package and suggests introducing nominal attributes for each level of the hierarchy and encoding examples using these new attributes. [1] proposes a technique for choosing a node in an AVT for a binary split using the information-gain criterion. [2] consider a multiple split test, where each test corresponds to a cut through AVT. Because number of cuts and hence the number of tests to be considered grows exponentially in the number of leaves of the hierarchy, this method scales poorly with the size of the hierarchy. [17], [39] and [22] describe the use of AVT in rule learning. [20] proposed a method

[2] A Java implementation of AVT-NBL and the data sets and AVTs used in this study are available at http://www.cs.iastate.edu/˜jzhang/ICDM04/index.html

for exploring hierarchically structured background knowledge for learning association rules at multiple levels of abstraction. [16] suggested the use of abstraction-based search (ABS) to learn Bayesian networks with compact structure. [45] describe AVT-DTL, an efficient algorithm for learning decision-tree classifiers from AVT and partially specified data. There has been very little experimental investigation of these algorithms in learning classifiers using data sets and AVT from real-world applications. Furthermore, with the exception of AVT-DTL, to the best of our knowledge, there are no algorithms for learning classifiers from AVT and partially specified data.

Attribute value taxonomies allow the use of a hierarchy of abstract attribute values (corresponding to nodes in an AVT) in building classifiers. Each abstract value of an attribute corresponds to a set of primitive values of the corresponding attribute. Quinlan's C4.5 [36] provides an option called subsetting, which allows C4.5 to consider splits based on subsets of attribute values (as opposed to single values) along each branch. [13] has also incorporated set-valued attributes in the RIPPER algorithm for rule learning. However, set-valued attributes are not constrained by an AVT. An unconstrained search through candidate subsets of values of each attribute during the learning phase can result in compact classifiers if compactness is measured in terms of the number of nodes in a decision tree. However, this measure of compactness is misleading because, in the absence of the structure imposed over sets of attribute values used in constructing the classifier, specifying the outcome of each test (outgoing branch from a node in the decision tree) requires enumerating the members of the set of values corresponding to that branch, making each rule a conjunction of arbitrary disjunctions (as opposed to disjunctions constrained by an AVT), making the resulting classifiers difficult to interpret. Because algorithms like RIPPER and C4.5 with subsetting have to search the set of candidate value subsets for each attribute under consideration, while adding conditions to a rule or a node to trees, they are computationally more demanding than algorithms that incorporate the AVTs into learning directly. At present, algorithms that utilize set-valued attributes do not include the capability to learn from partially specified data. Neither do they lend themselves to exploratory data analysis wherein users need to explore data from multiple perspectives (which correspond to different choices of AVT).

There has been some work on the use of class taxonomy (CT) in the learning of classifiers in scenarios where class labels correspond to nodes in a predefined class hierarchy. [12] have proposed a revised entropy calculation for constructing decision trees for assigning protein sequences to hierarchically structured functional classes. [27] describes the use of taxonomies over class labels to improve the performance of text classifiers. But none of them address the problem of learning from partially specified data (where class labels and/or attribute values are partially specified).

There is a large body of work on the use of *domain theories* to guide learning. The use of prior knowledge or domain theories specified typically in first-order logic to guide learning from data in the ML-SMART system [6]; the FOCL system [33]; and the KBANN system, which initializes a neural network using a domain theory specified in propositional logic [40]. AVT can be viewed as a restricted class of domain theories. [3] used background knowledge to generate

relational features for knowledge discovery. [4] applied breadth-first marker propagation to exploit background knowledge in rule learning. However, the work on exploiting domain theories in learning has not focused on the effective use of AVT to learn classifiers from partially specified data. [42] first used the taxonomies in information retrieval from large databases. [14] and [11] proposed database models to handle imprecision using partial values and associated probabilities, where a partial value refers to a set of possible values for an attribute. [30] proposed aggregation operators defined over partial values. While this work suggests ways to aggregate statistics so as to minimize information loss, it does not address the problem of learning from AVT and partially specified data.

Automated construction of hierarchical taxonomies over attribute values and class labels is beginning to receive attention in the machine-learning community. Examples include distributional clustering [35], extended FOCL and statistical clustering [43], information bottleneck [38], link-based clustering on relational data [8]. Such algorithms provide a source of AVT in domains where none are available. The focus of work described in this paper is on algorithms that use AVT in learning classifiers from data.

### 6.3 Future work

Some promising directions for future work in AVT-guided learning include

1. Development AVT-based variants of other machine-learning algorithms for construction of classifiers from partially specified data and from distributed, semantically heterogeneous data sources [9, 10]. Specifically, it would be interesting to design AVT- and CT-based variants of algorithms for constructing bag-of-words classifiers, Bayesian networks, nonlinear regression classifiers, and hyperplane classifiers (Perceptron, Winnow Perceptron, and Support Vector Machines).
2. Extensions that incorporate class taxonomies (CT). It would be interesting to explore approaches that exploit the hierarchical structure over class labels directly in constructing classifiers. It is also interesting to explore several possibilities for combining approaches to exploiting CT with approaches to exploiting AVT to design algorithms that make the optimal use of CT and AVT to learn robust, compact and easy-to-interpret classifiers from partially specified data.
3. Extensions that incorporate richer classes of AVT. Our work has so far focused on tree-structured taxonomies defined over nominal attribute values. It would be interesting to extend this work in several directions motivated by the natural characteristics of data: (a) Hierarchies of intervals to handle numerical attribute values; (b) ordered generalization hierarchies, where there is an ordering relation among nodes at a given level of a hierarchy (e.g. hierarchies over education levels); (c) tangled hierarchies that are represented by directed acyclic graphs (DAG) and incomplete hierarchies, which can be represented by a forest of trees or DAGs.
4. Further experimental evaluation of AVT-NBL, AVT-DTL and related learning algorithms on a broad range of data sets in scientific knowledge discovery

applications, including: (a) census data from official data libraries[3]; (b) data sets for macromolecular sequence-structure-function relationships discovery, including Gene Ontology Consortium[4] and MIPS[5]; (c) data sets of system and network logs for intrusion detection.

**Acknowledgements** This research was supported in part by grants from the National Science Foundation (NSF IIS 0219699) and the National Institutes of Health (GM 066387).

## References

1. Almuallim H, Akiba Y, Kaneda S (1995) On handling tree-structured attributes. In: Proceedings of the twelfth international conference on machine learning. Morgan Kaufmann, pp 12–20
2. Almuallim H, Akiba Y, Kaneda S (1996) An efficient algorithm for finding optimal gain-ratio multiple-split tests on hierarchical attributes in decision tree learning. In: Proceedings of the thirteenth national conference on artificial intelligence and eighth innovative applications of artificial intelligence conference, vol 1. AAAI/MIT Press, pp 703–708
3. Aronis J, Provost F, Buchanan B (1996) Exploiting background knowledge in automated discovery. In: Proceedings of the second international conference on knowledge discovery and data mining. AAAI Press, pp 355–358
4. Aronis J, Provost F (1997) Increasing the efficiency of inductive learning with breadth-first marker propagation. In: Proceedings of the third international conference on knowledge discovery and data mining. AAAI Press, pp 119–122
5. Ashburner M, et al (2000) Gene ontology: tool for the unification of biology. The Gene Ontology Consortium. Nat Gen 25:25–29
6. Bergadano F, Giordana A (1990) Guiding induction with domain theories. Machine learning—an artificial intelligence approach, vol. 3. Morgan Kaufmann, pp 474–492
7. Berners-Lee T, Hendler J, Lassila O (2001) The semantic web. Sci Am pp 35–43
8. Bhattacharya I, Getoor L (2004) Deduplication and group detection using links. KDD workshop on link analysis and group detection, Aug. 2004. Seattle
9. Caragea D, Silvescu A, Honavar V (2004) A framework for learning from distributed data using sufficient statistics and its application to learning decision trees. Int J Hybrid Intell Syst 1:80–89
10. Caragea D, Pathak J, Honavar V (2004) Learning classifiers from semantically heterogeneous data. In: Proceedings of the third international conference on ontologies, databases, and applications of semantics for large scale information systems. pp 963–980
11. Chen A, Chiu J, Tseng F (1996) Evaluating aggregate operations over imprecise data. IEEE Trans Knowl Data En 8:273–284
12. Clare A, King R (2001) Knowledge discovery in multi-label phenotype data. In: Proceedings of the fifth European conference on principles of data mining and knowledge discovery. Lecture notes in computer science, vol 2168. Springer, Berlin Heidelberg New York, pp 42–53
13. Cohen W (1996) Learning trees and rules with set-valued features. In: Proceedings of the thirteenth national conference on artificial intelligence. AAAI/MIT Press, pp 709–716
14. DeMichiel L (1989) Resolving database incompatibility: an approach to performing relational operations over mismatched domains. IEEE Trans Knowl Data Eng 1:485–493
15. Dempster A, Laird N, Rubin D (1977) Maximum likelihood from incomplete data via the EM algorithm. J Royal Stat Soc, Series B 39:1–38
16. desJardins M, Getoor L, Koller D (2000) Using feature hierarchies in Bayesian network learning. In: Proceedings of symposium on abstraction, reformulation, and approximation 2000. Lecture notes in artificial intelligence, vol 1864, Springer, Berlin Heidelberg New York, pp 260–270

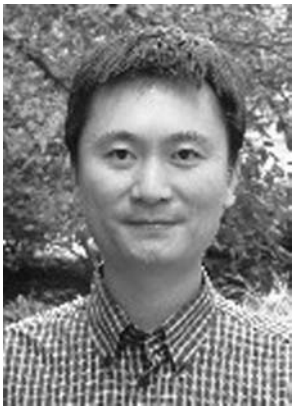
[3] http://www.thedataweb.org/

[4] http://www.geneontology.org/

[5] http://mips.gsf.de/

17. Dhar V, Tuzhilin A (1993) Abstract-driven pattern discovery in databases. IEEE Trans Knowl Data Eng 5:926–938
18. Domingos P, Pazzani M (1997) On the optimality of the simple Bayesian classifier under zero-one loss. Mach Learn 29:103–130
19. Friedman N, Geiger D, Goldszmidt M (1997) Bayesian network classifiers. Mach Learn 29:131–163
20. Han J, Fu Y (1996) Attribute-oriented induction in data mining. Advances in knowledge discovery and data mining. AAAI/MIT Press, pp 399–421
21. Haussler D (1998) Quantifying inductive bias: AI learning algorithms and Valiant's learning framework. Artif Intell 36:177–221
22. Hendler J, Stoffel K, Taylor M (1996) Advances in high performance knowledge representation. University of Maryland Institute for Advanced Computer Studies, Dept. of Computer Science, Univ. of Maryland, July 1996. CS-TR-3672 (Also cross-referenced as UMIACS-TR-96-56)
23. Kang D, Silvescu A, Zhang J, Honavar V (2004) Generation of attribute value taxonomies from data for data-driven construction of accurate and compact classifiers. In: Proceedings of the fourth IEEE international conference on data mining, pp 130–137
24. Kohavi R, Becker B, Sommerfield D (1997) Improving simple Bayes. Tech. Report, Data mining and visualization group, Silicon Graphics Inc.
25. Kohavi R, Provost P (2001) Applications of data mining to electronic commerce. Data Min Knowl Discov 5:5–10
26. Kohavi R, Mason L, Parekh R, Zheng Z (2004) Lessons and challenges from mining retail E-commerce data. Special Issue: Data mining lessons learned. Mach Learn 57:83–113
27. Koller D, Sahami M (1997) Hierarchically classifying documents using very few words. In: Proceedings of the fourteenth international conference on machine learning. Morgan Kaufmann, pp 170–178
28. Langley P, Iba W, Thompson K (1992) An analysis of Bayesian classifiers. In: Proceedings of the tenth national conference on artificial intelligence. AAAI/MIT Press, pp 223–228
29. McCallum A, Rosenfeld R, Mitchell T, Ng A (1998) Improving text classification by shrinkage in a hierarchy of classes. In: Proceedings of the fifteenth international conference on machine learning. Morgan Kaufmann, pp 359–367
30. McClean S, Scotney B, Shapcott M (2001) Aggregation of imprecise and uncertain information in databases. IEEE Trans Know Data Eng 13:902–912
31. Mitchell T (1997) Machine Learning. Addison-Wesley
32. Núñez M (1991) The use of background knowledge in decision tree induction. Mach Learn 6:231–250
33. Pazzani M, Kibler D (1992) The role of prior knowledge in inductive learning. Mach Learn 9:54–97
34. Pazzani M, Mani S, Shankle W (1997) Beyond concise and colorful: learning intelligible rules. In: Proceedings of the third international conference on knowledge discovery and data mining. AAAI Press, pp 235–238
35. Pereira F, Tishby N, Lee L (1993) Distributional clustering of English words. In: Proceedings of the thirty-first annual meeting of the association for computational linguistics. pp 183–190
36. Quinlan JR (1993) C4.5: programs for machine learning. Morgan Kaufmann, San Mateo, CA
37. Rissanen J (1978) Modeling by shortest data description. Automatica 14:37–38
38. Slonim N, Tishby N (2000) Document clustering using word clusters via the information bottleneck method. ACM SIGIR 2000. pp 208–215
39. Taylor M, Stoffel K, Hendler J (1997) Ontology-based induction of high level classification rules. SIGMOD data mining and knowledge discovery workshop, Tuscon, Arizona
40. Towell G, Shavlik J (1994) Knowledge-based artificial neural networks. Artif Intell 70:119–165
41. Undercoffer J, et al (2004) A target centric ontology for intrusion detection: using DAML+OIL to classify intrusive behaviors. Knowledge Engineering Review—Special Issue on Ontologies for Distributed Systems, January 2004, Cambridge University Press

42. Walker A (1980) On retrieval from a small version of a large database. In: Proceedings of the sixth international conference on very large data bases. pp 47–54
43. Yamazaki T, Pazzani M, Merz C (1995) Learning hierarchies from ambiguous natural language data. In: Proceedings of the twelfth international conference on machine learning. Morgan Kaufmann, pp 575–583
44. Zhang J, Silvescu A, Honavar V (2002) Ontology-driven induction of decision trees at multiple levels of abstraction. In: Proceedings of symposium on abstraction, reformulation, and approximation 2002. Lecture notes in artificial intelligence, vol 2371. Springer, Berlin Heidelberg New York, pp 316–323
45. Zhang J, Honavar V (2003) Learning decision tree classifiers from attribute value taxonomies and partially specified data. In: Proceedings of the twentieth international conference on machine learning. AAAI Press, pp 880–887
46. Zhang J, Honavar V (2004) AVT-NBL: an algorithm for learning compact and accurate naive Bayes classifiers from attribute value taxonomies and data. In: Proceedings of the fourth IEEE international conference on data mining. IEEE Computer Society, pp 289–296



**Jun Zhang** is currently a PhD candidate in computer science at Iowa State University, USA. His research interests include machine learning, data mining, ontology-driven learning, computational biology and bioinformatics, evolutionary computation and neural networks. From 1993 to 2000, he was a lecturer in computer engineering at University of Science and Technology of China. Jun Zhang received a MS degree in computer engineering from the University of Science and Technology of China in 1993 and a BS in computer science from Hefei University of Technology, China, in 1990.

**Dae-Ki Kang** is a PhD student in computer science at Iowa State University. His research interests include ontology learning, relational learning, and security informatics. Prior to joining Iowa State, he worked at a Bay-area startup company and at Electronics and Telecommunication Research Institute in South Korea. He received a Masters degree in computer science at Sogang University in 1994 and a bachelor of engineering (BE) degree in computer science and engineering at Hanyang University in Ansan in 1992.

**Adrian Silvescu** is a PhD candidate in computer science at Iowa State University. His research interests include machine learning, artificial intelligence, bioinformatics and complex adaptive systems. He received a MS degree in theoretical computer science from the University of Bucharest, Romania, in 1997, and received a BS in computer science from the University of Bucharest in 1996.

**Vasant Honavar** received a BE in electronics engineering from Bangalore University, India, an MS in electrical and computer Engineering from Drexel University and an MS and a PhD in computer science from the University of Wisconsin, Madison. He founded (in 1990) and has been the director of the Artificial Intelligence Research Laboratory at Iowa State University (ISU), where he is currently a professor of computer science and of bioinformatics and computational biology. He directs the Computational Intelligence, Learning & Discovery Program, which he founded in 2004. Honavar's research and teaching interests include artificial intelligence, machine learning, bioinformatics, computational molecular biology, intelligent agents and multiagent systems, collaborative information systems, semantic web, environmental informatics, security informatics, social informatics, neural computation, systems biology, data mining, knowledge discovery and visualization. Honavar has published over 150 research articles in refereed journals, conferences and books and has coedited 6 books. Honavar is a coeditor-in-chief of the Journal of Cognitive Systems Research and a member of the Editorial Board of the Machine Learning Journal and the International Journal of Computer and Information Security. Prof. Honavar is a member of the Association for Computing Machinery (ACM), American Association for Artificial Intelligence (AAAI), Institute of Electrical and Electronic Engineers (IEEE), International Society for Computational Biology (ISCB), the New York Academy of Sciences, the American Association for the Advancement of Science (AAAS) and the American Medical Informatics Association (AMIA).