

Efficient Learning of Selective Bayesian Network Classifiers

MS-CIS-95-36

Moninder Singh
and
Gregory M. Provan



University of Pennsylvania
School of Engineering and Applied Science
Computer and Information Science Department
Philadelphia, PA 19104-6389

November, 1995

Efficient Learning of Selective Bayesian Network Classifiers

Moninder Singh*

Department of Computer and Information Science
University of Pennsylvania
Philadelphia, PA 19104-6389
{*msingh@gradient.cis.upenn.edu*}

and

Gregory M. Provan†

Institute for the Study of Learning and Expertise
2164 Staunton Ct., Palo Alto, CA 94306
{*provan@camis.stanford.edu*}

*This work was supported by an IBM Cooperative Fellowship.

†This work was supported by NSF grants #IRI92-10030 and #IRI91-20330, and NLM grant #BLR 3 RO1 LMO5217-02S1.

Abstract

In this paper, we present a computationally efficient method for inducing *selective* Bayesian network classifiers. Our approach is to use information-theoretic metrics to efficiently select a subset of attributes from which to learn the classifier. We explore three conditional, information-theoretic metrics that are extensions of metrics used extensively in decision tree learning, namely Quinlan's gain and gain ratio metrics and Mantaras's distance metric.

We experimentally show that the algorithms based on gain ratio and distance metric learn selective Bayesian networks that have predictive accuracies as good as or better than those learned by existing selective Bayesian network induction approaches (K2-AS), but at a significantly lower computational cost. We prove that the subset-selection phase of these information-based algorithms has polynomial complexity as compared to the worst-case exponential time complexity of the corresponding phase in K2-AS.

We also compare the performance of this classifier with selective and non-selective *naive* Bayesian classifiers. We show that the selective Bayesian network classifier performs significantly better than both versions of the naive Bayesian classifier on almost all databases analyzed, and hence is an enhancement of the naive Bayesian classifier.

Finally, we show that the proposed selective Bayesian network classifier generates networks that are computationally simpler to evaluate and that display predictive accuracy comparable to that of Bayesian networks which model all attributes.

Contents

List of Tables	ii
List of Figures	ii
1 Introduction	1
2 Naive Bayesian Classifiers	3
3 Bayesian Network Classifiers	4
4 Efficient Learning of Bayesian Network Classifiers using Attribute Selection	6
4.1 Attribute selection metrics	6
4.2 The Info-AS algorithm	7
4.3 Complexity of the node-selection phase	8
5 Experimental Comparison of Info-AS with other induction methods	10
5.1 Experimental Design	10
5.2 Results	12
5.3 Comparison with <i>naive</i> Bayesian classifiers	13
5.4 Comparison with Bayesian Network classifiers	15
5.5 Comparison with Decision Tree classifiers	17
5.6 Discussion	17
6 Future Work	19
7 Conclusion	20

List of Tables

1	Description of the databases.	10
2	Predictive accuracies for the different algorithms.	12
3	Predictive accuracies for the different algorithms (cont.).	13
4	Comparison of CIG-AS with naive-ALL and naive-AS	13
5	Comparison of CGR-AS with naive-ALL and naive-AS	14
6	Comparison of CDC-AS with naive-ALL and naive-AS	14
7	Comparison of CIG-AS with K2-AS, K2-AS< and CB	15
8	Comparison of CGR-AS with K2-AS, K2-AS< and CB	15
9	Comparison of CDC-AS with K2-AS, K2-AS< and CB	16
10	Comparison with C4.5.	16
11	Comparison with C4.5 (cont.).	16

List of Figures

1	Learning curves for the Chess database.	25
2	Learning curves for the Mushroom database.	26
3	Learning curves for the Soybean database.	27
4	Learning curves for the Voting database.	28

1 Introduction

One of the simplest Bayesian induction methods is the *naive Bayesian classifier* [8, 9]. Despite its simplicity and the strong assumption that the attributes are conditionally independent given the class variable, the naive Bayesian classifier has been shown to perform remarkably well in some domains. The simplest *naive* classifier is one which consists of all attributes (non-selective *naive* Bayesian classifier). However, due to its strong independence assumptions, the classification performance of such classifiers can degrade immensely in domains which contains attributes that are correlated. Langley and Sage [10] proposed a method to eliminate this drawback by using only a subset of the attributes to construct the classifier (selective *naive* Bayesian classifier) in an attempt to exclude highly correlated attributes, thereby improving performance in domains with many dependencies between attributes. Pazzani [13] attempted to deal with this problem in a different way. His ‘backward sequential elimination and joining algorithm’ attempts to take into account the dependencies between various attributes by merging them into a single attribute. This approach does not deal with the question of independence directly; rather, it tries to determine whether joining attributes will improve the predictive accuracy.

We directly model dependencies between the various attributes using Bayesian networks [14], which are a natural extension of *naive* Bayesian classifiers. However, inference using Bayesian networks is an \mathcal{NP} -hard problem. Hence, learning Bayesian network classifiers from a subset of the attributes may be a good compromise between *naive* Bayesian classifiers and Bayesian networks that model all attributes. The K2-AS algorithm [18] was an attempt in this direction and has been shown to outperform both selective as well as non-selective *naive* Bayesian classifiers. Moreover, K2-AS, by selecting a subset of attributes prior to learning the networks, not only significantly improves the inference efficiency of the resulting networks, but also achieves a predictive accuracy comparable to Bayesian networks learned using the full set of attributes [15, 16]. However, one of the main drawbacks of this approach is that the subset learning phase is computationally intensive. The K2-AS algorithm uses forward sequential

search to incrementally add that attribute to the set of selected attributes that most increases the predictive accuracy of the Bayesian network generated from the current set of selected attributes and the attribute under consideration. As such, it has to perform inference on Bayesian networks at every stage of the attribute selection process.

In this paper we use information to guide the search process in order to learn better, or at least as good, selective Bayesian network classifiers in a more efficient way. Information based metrics have been widely used within the Machine Learning community to learn highly accurate decision trees [17, 2, 11]. Induction of a decision tree from a training set of cases is normally handled by a recursive partitioning algorithm which, at each non-terminal node in the tree, branches on that attribute which discriminates best between the cases filtered down to that node. A number of information-based splitting rules (that decide which of the available attributes is the “best” attribute to branch on) have been studied by various researchers. Amongst these are Quinlan’s information gain and gain ratio [17] and Lopez de Mantaras’s distance measure (d_N) [11]. In the case of information gain and gain ratio that attribute is selected (from the available attributes) which maximizes this measure. The distance measure on the other hand needs to be minimized over the attributes; hence, one often uses the complement of this measure, $1 - d_N$.

In this paper we present a novel selective Bayesian network induction method, Info-AS¹ and compare its classification performance with selective and non-selective *naive* Bayesian classifiers as well as with selective and non-selective Bayesian network classifiers. The Info-AS algorithm uses an information-based metric to select a subset of attributes prior to the network learning phase: our goal is to construct networks that are simpler to evaluate, but which still have high predictive accuracy relative to networks induced with all attributes. The metrics used in our experiments are extensions of the three metrics described above. Since a Bayesian network classifier is a natural extension of the naive Bayesian classifier in that it does not make restrictive conditional independence assumptions, it should perform better than the naive

¹It uses Information based metrics for Attribute Selection.

approach.

Our experimental results comparing selective Bayesian network and naive Bayesian methods show that the selective Bayesian network classifier performs significantly better than both versions of the naive Bayesian classifier on almost all databases analyzed, and hence is an enhancement of the naive Bayesian classifier. Moreover, we have experimentally compared selective and non-selective Bayesian network classifiers, showing that the selective approach generates networks that are computationally simpler to evaluate and that display predictive accuracy comparable to that of Bayesian networks which model all attributes. The selective Bayesian network thus seems to be a good compromise between naive Bayesian classifiers and non-selective Bayesian networks.

The remainder of the paper is organized as follows. In section 2, we discuss briefly the various induction methods for *naive* Bayesian classifiers with which we compare the performance of Info-AS. Section 3 describes the various Bayesian network induction methods used in this study. We present the Info-AS algorithm in section 4 and present experimental results of our comparison of Info-AS with other classifiers in section 5. Finally, we summarize our contributions in Section 7.

2 Naive Bayesian Classifiers

As stated before, the *naive* Bayesian classifier assumes that the attributes are conditionally independent given the class variable. The classification process involves a class variable C that can take on values c_1, c_2, \dots, c_k , and an attribute vector $\Delta = \{A_1, A_2, \dots, A_q\}$ of q attributes that can take on a tuple of values denoted by $\{v_1, v_2, \dots, v_q\}$. Given a case V represented by an instantiation $\{v_1, v_2, \dots, v_q\}$ of attribute values, our classification task is to define the class variable member c_i that V falls into.

To perform this task, we assume that we have the prior probabilities for each value c_i of the class variable, $P(c_i)$. Further, we assume that we have the conditional

probability distributions for each attribute value v_j given class value c_i , $P(v_j|c_i)$.²

Given this data, we can classify a new case $V = \bigwedge_j v_j$ using Bayes' rule:

$$P(c_i|V) = \frac{P(c_i)P(V|c_i)}{P(V)} = \frac{P(c_i)P(\bigwedge_j v_j|c_i)}{\sum_k P(\bigwedge_j v_j|c_k)P(c_k)}. \quad (1)$$

We can use the assumption of the independence of attributes within each class to rewrite the denominator of equation 1 to give

$$P(c_i|\bigwedge_j v_j) = \frac{P(c_i)\prod_j P(v_j|c_i)}{\sum_k \prod_j P(v_j|c_k)P(c_k)}. \quad (2)$$

Induction consists simply in counting the class value and the class value / attribute value pair. This allows us to estimate the required probabilities $P(c_i)$ and $P(v_j|c_i)$. We refer to this *naive* Bayesian classifier (that models all attributes) as naive-ALL.

Although, its performance is remarkably good given its simplicity, it is typically limited to learning classes that can be separated by a single decision boundary [9], and in domains in which the attributes are correlated given the class variable, its performance can be worse than other approaches which can account for such correlations. Bayesian networks can account for correlations among attributes, so they are a natural extension of the naive approach.

The selective *naive* Bayesian classifier [10] (referred to as naive-AS) is an extension to the naive Bayesian classifier, and is designed to perform better in domains with redundant attributes. The intuition is that if highly correlated attributes are not selected, the classifier should perform better given its attribute independence assumptions. Using forward selection of attributes, this approach uses a greedy search, at each point in the search, to select from the space of all attribute subsets the attribute which most improves accuracy on the entire training set. Attributes are added until the addition of any other attribute results in reduced accuracy.

3 Bayesian Network Classifiers

Cooper and Herskovits [3] proposed a Bayesian method for the induction of Bayesian networks from data. They designed a greedy algorithm, K2, that attempts to select

²For simplicity of exposition, we will restrict our discussion to nominal domains.

the network (out of a set of possible networks exponential in the number of network nodes) which maximizes the posterior probability of the network given the database of cases. Given a total ordering on the attributes, K2 takes each successive attribute in the ordering, adds it as a node in the network, and creates parents for it in a greedy fashion: it first assumes that the node has no parents, and then adds incrementally that node (amongst the predecessors in the ordering) as a parent which increases the probability of the resultant structure by the largest amount. It stops adding parents to the node when adding any additional single parent cannot increase the probability.

Whereas K2 assumes a node ordering, the CB algorithm [19, 20] uses conditional independence (CI) tests to generate a “good” node ordering from the data, and then uses the K2 algorithm to generate the Bayesian network from the database using this node ordering. Starting with the complete, undirected graph on all variables, the CB algorithm first deletes edges between adjacent nodes that are unconditionally independent (CI tests of order 0). The edges in the resultant graph are then oriented and a total ordering on the variables is obtained. This ordering is then used with the K2 algorithm to construct the corresponding network. The algorithm then repeats this process by removing edges (from the undirected graph obtained in the previous iteration) between adjacent nodes that are conditionally independent given one node (CI test of order 1). It keeps constructing networks for increasing orders of CI tests as long as the predictive accuracy of the resultant network keeps increasing.

Singh and Provan [18, 15] proposed an attribute selection based enhancement of this approach called K2-AS. Using forward selection of attributes, this approach incrementally adds that attribute to the set of selected attributes that most increases the predictive accuracy of the Bayesian network learned from the current set of selected attributes and the attribute under consideration. Attribute are added as long as the predictive accuracy increases. A variation of the algorithm adopts the more conservative strategy of adding nodes as long as the predictive accuracy either increases or remains unchanged (K2-AS \leq). The CB algorithms is then used to learn the final network from the set of selected nodes.

4 Efficient Learning of Bayesian Network Classifiers using Attribute Selection

Our learning approach consists of two main steps, node selection and network construction. In the node selection phase, we choose the set of nodes from which the final network is constructed. The nodes are selected using an information-theoretic metric. At each stage of the node selection process, that node is selected which displays the maximum discriminating ability (as measured by the metric) between the various classes, given the current set of selected nodes. In the network construction phase, we construct the network from the subset of attributes selected in the previous phase. Finally, we test the predictive accuracy of the network.

4.1 Attribute selection metrics

As stated previously, we looked at three metrics, based on splitting rules commonly used in decision tree induction, for attribute selection. The metrics we experimented with are Conditional Information Gain (CIG), Conditional Gain Ratio (CGR) and the complement of Conditional Distance i.e. 1- Conditional Distance (CDC). Let C be the class variable and A represent the attribute under consideration. Let k be the number of classes and let m be the number of possible values of A . Moreover, let $\Delta = \{A_1, \dots, A_q\}$ and let s be the cardinality of the cross product of the sets of values of these variables. Given that the set Δ is instantiated to its l^{th} unique instantiation, let $p_{i/l}$ represent the probability that the class variable is instantiated to its i^{th} value and let $p_{j/l}$ be the probability that A is instantiated to its j^{th} value. Similarly, $p_{ij/l}$ is the probability that the class variable takes on its i^{th} value and the attribute A takes on its j^{th} value, given that Δ is instantiated to its l^{th} unique instantiation. These probabilities can be calculated by constructing a 3-d contingency table from the given cases. Then, following the terminology of White and Liu [21], we can define

$$H_{Cell_l} = - \sum_{i=1}^k \sum_{j=1}^m p_{ij/l} \log p_{ij/l} \quad (3)$$

$$H_{A_l} = - \sum_{j=1}^m p_{j/l} \log p_{j/l} \quad (4)$$

$$H_{C_l} = - \sum_{i=1}^k p_{i/l} \log p_{i/l} \quad (5)$$

We can then define the three metrics as follows:

Definition 4.1

$$\begin{aligned} CIG(A, \Delta) &= Gain(A/\Delta) \\ &= \sum_{l=1}^s p_l (H_{C_l} + H_{A_l} - H_{C_{ell_l}}) \end{aligned}$$

Definition 4.2

$$\begin{aligned} CGR(A, \Delta) &= Gain_{Ratio}(A/\Delta) \\ &= \frac{\sum_{l=1}^s p_l (H_{C_l} + H_{A_l} - H_{C_{ell_l}})}{\sum_{l=1}^s p_l H_{A_l}} \end{aligned}$$

Definition 4.3

$$\begin{aligned} CDC(A, \Delta) &= 1 - d_N(A/\Delta) \\ &= \frac{\sum_{l=1}^s p_l (H_{C_l} + H_{A_l} - H_{C_{ell_l}})}{\sum_{l=1}^s p_l H_{C_{ell_l}}} \end{aligned}$$

4.2 The Info-AS algorithm

The algorithm used for the node selection phase is a forward selection algorithm, in that it starts with an empty set of attributes and adds attributes using a greedy search. This forward selection is just like K2. We now describe the different phases of the algorithm:

- **node selection phase:** In this phase, Info-AS chooses the set of attributes Δ , $\Delta \subseteq Z - class_{var}$ (Z is the set of all variables) from which (along with the $class_{var}$) the final network is constructed. The algorithm starts with the initial

assumption that Δ is the empty set. It then adds incrementally that attribute (from $Z - \Delta$) whose addition maximizes the value of the information metric. When there is no single attribute whose addition results in a positive value of the information metric ³, the algorithm stops adding attributes. We define $\mathcal{IM}(\Delta, x)$ to be the value of the information metric on adding x to the set Δ of already selected attributes. This phase can be described as follows:

```

 $\Delta \leftarrow \emptyset$ 
NotDone  $\leftarrow$  True
while NotDone do
   $\forall x \in Z - \Delta$ , let  $I_x \leftarrow \mathcal{IM}(\Delta, x)$ 
   $I_{max} \leftarrow \max_x (I_x)$ 
   $z = \arg \max_x (I_x)$ 
  if  $I_{max} > 0$  then
     $\Delta \leftarrow \Delta \cup \{z\}$ 
  else NotDone  $\leftarrow$  false
end {while};

```

- **network construction phase:** Info-AS uses the final set of nodes Δ selected in the node selection phase along with $class_{var}$ to construct a network using training data.
- **network evaluation phase:** In order to test the quality of the network, we test the network for its predictive accuracy on the test data.

4.3 Complexity of the node-selection phase

We now derive the worst-case time complexity of selecting the subset of nodes in the node-selection phase using the metrics presented. Having selected a set of attributes, say Δ , we can easily calculate the value of $\mathcal{IM}(\Delta, x)$ (where x is the node under

³We take the value of CGR or CDC to be zero if the denominator is zero.

consideration) by building a 3-d contingency table whose rows correspond to the various classes, the columns correspond to the different values of the attribute x , and the layers correspond to the unique instantiations of the attributes in Δ . Two stages contribute to the complexity of this process—forming the contingency table and then using it to calculate $\mathcal{IM}(\Delta, x)$.

For now, assume that the contingency table has already been created. Let r be the maximum number of possible values for any attribute. The total number of unique instantiations of the attributes in Δ , hence the number of layers in the contingency table, can be at most m where m is the number of cases in the training data. In order to calculate $\mathcal{IM}(\Delta, x)$, each cell in the contingency table has to be accessed. This can be done in $O(mr^2)$.

Now, consider the complexity of creating the contingency table itself. The contingency table can be created by building an *index* tree⁴ The index tree is a tree of depth $|\Delta| + 1$ where the leaves are 2-d contingency tables. Each interior node represents an attribute in Δ (with the nodes at a given level all representing the same attribute) and has an outgoing edge for each possible value that the attribute can take. Thus, a path from the root of the tree to a leaf (a 2-d contingency table) represents a unique instantiation of the attributes in Δ . Note that paths in the tree may be partial (and hence have no corresponding 2-d table) because that particular instantiation of the attributes in Δ may not occur in the training set. Since there are m cases in the training set, there will be at most m 2-d tables (as opposed to the theoretical maximum of r^k where $|\Delta| = k$). In order to enter a case into the contingency table, we must branch on or construct a path with k nodes, each of size r . Thus, a case can be entered in $O(kr)$ and the entire contingency table can be constructed in $O(mkr)$.

Now, consider the node selection phase. The process starts with $\Delta = \emptyset$, i.e., $k = 0$. At each stage, it evaluates the metric for $n - k - 1$ attributes where n is the total number of variables (including the *class_{var}*). In the worst case, it selects all the attributes and hence stops when $|\Delta| = n - 2$. Thus, the worst case time complexity of the node-selection phase is given by

⁴Cooper and Herskovits [3] used a similar technique to calculate $P(B_S, D)$ efficiently.

$$\sum_{k=0}^{n-2} (mkr + mr^2) (n - k - 1) = O(mrn^2 (r + n)).$$

5 Experimental Comparison of Info-AS with other induction methods

In our experiments we used a variety of databases acquired from the University of California, Irvine Repository of Machine Learning databases [12]. The databases we used were Michalski’s Soybean database, Schlimmer’s Mushroom and Voting databases, the Gene-Splicing database due to Towell, Noordewier and Shavlik and Shapiro’s Chess Endgame database. Further information on these databases can be obtained from the UCI repository by anonymous ftp to ics.uci.edu.

Table 1: Description of the databases.

	Attributes	Classes	Cases (TOTAL)	Cases in <i>training</i> set	Cases in <i>test</i> set
Soybean	35	15	630	430	200
Chess	36	2	3196	1000	2196
Mushroom	22	2	8124	500	7624
Voting	16	2	435	200	235
Voting1	15	2	435	200	235
Gene-splice	60	3	3175	2000	1175

Table 1 summarizes the databases used in terms of number of cases, classes and attributes (excluding the class variable). The voting1 database was derived from the voting database by deleting the most significant attribute *physician-fee-freeze* [2].

5.1 Experimental Design

The experimental method, applied to each data set, is described below:

1. Repeat 30 times:
 - (a) Randomly split the database into two parts: the first part, the *training* data, is used for learning the network; the second part, the *test* data, is

then used for determining the predictive accuracy of the learned network. The two sets are always a fixed size (Table 1).

For the K2-AS as well as the naive-AS algorithms, the *training* data is further split into two equal parts, referred to as the *subset-learn* and *subset-eval* set.

- (b) For each algorithm, learn a network using the *training* set and then determine its predictive accuracy on the *test* data. The *predictive accuracy* of an algorithm is the percentage of test cases for which it predicts the class correctly.

For the K2-AS and naive-AS algorithms, the *subset-learn* set is used for learning the network (from the current subset of nodes and the node under consideration) while the *subset-eval* is used to test this network for predictive accuracy (to decide whether to add the new node to the set of selected nodes). Once the subset of nodes has been selected, the *training* set is used to learn the final network which is then tested on the *test* data.

2. Determine the sample average and standard deviation for the predictive accuracies for each algorithm.
3. In order to check the significance of the differences in predictive accuracies, calculate the *t*-value for the two-tailed paired-*t* test between the different pairs of algorithms.

For learning the Bayesian network from the subset of nodes selected, we used the CB algorithm (section 3). We must emphasize here that our choice of CB as the learning algorithm was totally arbitrary. One could have chosen any other methods for learning BN's just as well, e.g. Heckerman et. al. [5]. The main contribution of this paper lies in showing the feasibility, advantages and effectiveness of learning Bayesian networks using feature selection, and presenting an efficient means of doing so. The CB algorithm may not be the best algorithm to use here since it is based on the K2 metric which attempts to learn the network which "best" fits the data and

does not take into account the classification accuracy of the resulting models. Note that even though a given model may be “correct” in the sense of generating the data, it need not be the best model when it comes to making predictions [4]. However, as we shall see in section 5, it does suffice in showing that feature selection is a useful tool in learning BNs that display good classification accuracy while also improving the inference efficiency of the resulting networks.

We performed inference on the networks using the Lauritzen-Spiegelhalter inference algorithm as implemented in the HUGIN [1] system.

5.2 Results

The average predictive accuracies of the networks generated by the various methods are shown in Tables 2 and 3. Each result describes an average predictive accuracy on the test set for the 30 trials, followed by the sample standard deviation. The standard deviation shows how much the predictive accuracy varied from sample to sample. We refer to Info-AS with the CIG metric as CIG-AS. Similarly, CGR-AS and CDC-AS refer to Info-AS used with the CGR and CDC metric respectively.

Since, from a learning perspective, one is interested not only in asymptotic accuracy, but also in the rate of improvement, we also generated learning curves for some of the databases. These curves are depicted in Figures 1–4. In these curves, the predictive accuracies plotted are averaged over 20 trials.

Table 2: Predictive accuracies for the different algorithms.

	Chess	Voting	Voting1
Naive-ALL	86.71 \pm 1.65	89.69 \pm 1.48	87.08 \pm 1.70
Naive-AS	93.19 \pm 1.42	93.47 \pm 2.15	88.34 \pm 1.80
K2-AS	94.09 \pm 0.34	95.32 \pm 1.12	87.45 \pm 2.62
K2-AS<	95.47 \pm 1.18	95.18 \pm 0.82	89.43 \pm 2.08
CB	94.66 \pm 1.07	94.95 \pm 0.85	90.45 \pm 1.67
CIG-AS	96.13 \pm 1.00	94.91 \pm 0.96	90.27 \pm 1.85
CGR-AS	96.08 \pm 0.89	95.53 \pm 1.14	90.41 \pm 1.52
CDC-AS	96.26 \pm 0.84	95.55 \pm 1.05	90.37 \pm 1.48

Table 3: Predictive accuracies for the different algorithms (cont.).

	Soybean	Splice	Mushroom
Naive-ALL	88.67 \pm 2.05	95.41 \pm 0.69	92.87 \pm 0.51
Naive-AS	87.43 \pm 3.57	94.56 \pm 0.63	98.30 \pm 0.36
K2-AS	88.15 \pm 3.16	94.94 \pm 0.60	98.51 \pm 0.23
K2-AS<	90.50 \pm 3.11	–	99.27 \pm 0.49
CB	89.57 \pm 2.58	96.35 \pm 0.52	99.64 \pm 0.31
CIG-AS	84.00 \pm 4.39	94.58 \pm 0.63	99.33 \pm 0.39
CGR-AS	90.85 \pm 2.02	95.03 \pm 0.54	99.16 \pm 0.53
CDC-AS	89.15 \pm 2.58	95.02 \pm 0.56	99.20 \pm 0.55

Table 4: Comparison of CIG-AS with naive-ALL and naive-AS

	naive-ALL	naive-AS
Chess	-9.42 (-30.34) –	-2.94 (-9.98) –
Voting	-5.22 (-21.14) –	-1.43 (-3.65) –
Voting1	-3.19 (-8.91) –	-1.93 (-5.47) –
Soybean	4.67 (5.03) +	3.43 (3.95) +
Mushroom	-6.46 (-49.05) –	-1.03 (-9.50) –
Splice	0.84 (5.36) +	0.02 (0.12)

5.3 Comparison with *naive* Bayesian classifiers

Results of the two-tailed paired *t*-test comparing Info-AS with the *naive* Bayesian classifiers are shown in Tables 4, 5 and 6. Each result shows the average difference in predictive accuracies (negative value for an algorithm indicates that Info-AS using the corresponding metric performed better than it on the data-set under consideration), the value of the *t*-statistic and whether the difference is statistically significant or not. Significance is shown by means of the symbols ‘+’ and ‘–’. A ‘–’ for a given algorithm means that the Info-AS algorithm (using the corresponding metric) achieved a higher predictive accuracy than this algorithm on the data-set under consideration, and that this difference is significant at the 95% confidence level. Likewise, a ‘+’ indicates that the accuracy was significantly less for Info-AS (using the corresponding metric) when compared to the algorithm concerned on the data-set under consideration.

Table 5: Comparison of CGR-AS with naive-ALL and naive-AS

	naive-ALL	naive-AS
Chess	-9.37 (-26.53) –	-2.89 (-9.73) –
Voting	-5.84 (-24.26) –	-2.06 (-5.45) –
Voting1	-3.33 (-9.59) –	-2.07 (-7.83) –
Soybean	-2.18 (-5.47) –	-3.42 (-4.78) –
Mushroom	-6.29 (-41.17) –	-0.86 (-7.76) –
Splice	0.39 (2.43) +	-0.47 (-3.48) –

Table 6: Comparison of CDC-AS with naive-ALL and naive-AS

	naive-ALL	naive-AS
Chess	-9.56 (-28.15) –	-3.07 (-10.05) –
Voting	-5.86 (-23.74) –	-2.07 (-6.18) –
Voting1	-3.29 (-9.69) –	-2.03 (-6.37) –
Soybean	-0.48 (-0.98)	-1.72 (-2.31) –
Mushroom	-6.34 (-41.11) –	-0.91 (-7.93) –
Splice	0.41 (2.49) +	-0.46 (-3.43) –

As can be seen from Tables 4–6, Info-AS generally outperforms both naive-ALL as well as naive-AS. On 4 of the databases (Chess, Mushroom, Voting and Voting1), Info-AS always achieves higher predictive accuracy than both naive-ALL and naive-AS, and this difference is statistically significant. On the remaining two databases, the three metrics varied in their performance. On the splice database, all three methods were significantly poorer than naive-ALL; however, both CGR-AS and CDC-AS were significantly better than naive-AS while the difference between CIG-AS and naive-AS was not statistically significant. On the soybean database, CGR-AS was significantly better than both naive classifiers, CIG-AS was significantly poorer than both naive classifiers whereas CDC-AS was significantly better than naive-AS but was not statistically different from naive-ALL.

Table 7: Comparison of CIG-AS with K2-AS, K2-AS< and CB

	K2-AS	K2-AS<	CB
Chess	-2.04 (-11.44) -	-0.66 (-2.34) -	-1.47 (-4.78) -
Voting	0.41 (1.70)	0.27 (1.41)	0.04 (0.19)
Voting1	-2.82 (-6.40) -	-0.84 (-2.24) -	0.19 (0.55)
Soybean	4.15 (4.14) +	6.50 (9.68) +	5.57 (6.00) +
Mushroom	-0.82 (-11.98) -	-0.06 (-0.56)	0.31 (4.82) +
Splice	0.36 (2.21) +	--	1.77 (12.83) +

Table 8: Comparison of CGR-AS with K2-AS, K2-AS< and CB

	K2-AS	K2-AS<	CB
Chess	-1.99 (-13.63) -	-0.61 (-2.62) -	-1.41 (-4.22) -
Voting	-0.21 (-1.06) -	-0.36 (-1.94)	-0.58 (-3.70) -
Voting1	-2.96 (-6.82) -	-0.98 (-2.39) -	0.04 (0.16)
Soybean	-2.70 (-4.18) -	-0.35 (-0.59)	-1.28 (-2.53) -
Mushroom	-0.65 (-7.50) -	0.11 (0.90)	0.48 (5.22) +
Splice	-0.09 (-0.71)	-	1.32 (10.22) +

5.4 Comparison with Bayesian Network classifiers

We also compared Info-AS with non-selective Bayesian networks learned using the CB algorithm as well as with selective Bayesian networks learned using K2-AS and K2-AS<. The results of these experiments are shown in Tables 7–9. As in section 5.3, these tables show the average difference in predictive accuracies, the value of the t -statistic and whether the difference is statistically significant or not.

As can be seen from Tables 7–9, Info-AS is generally either better than or as good as both K2-AS and K2-AS<. CGR-AS performs significantly better than K2-AS on 5 of the 6 databases and significantly better than K2-AS< on 2 of the 5 databases tested. On the remaining databases, there was no significant difference. CDC-AS was significantly better than both K2-AS and K2-AS< on 3 databases, significantly poorer than K2-AS< on 1 database while there was no significant difference on the rest. CIG-AS performed significantly better than K2-AS on 3 databases, significantly

Table 9: Comparison of CDC-AS with K2-AS, K2-AS< and CB

	K2-AS	K2-AS<	CB
Chess	-2.17 (-14.43) -	-0.79 (-3.22) -	-1.60 (-5.62) -
Voting	-0.23 (-1.21)	-0.37 (-2.48) -	-0.60 (-3.70) -
Voting1	-2.92 (-6.43) -	-0.94 (-2.18) -	0.09 (0.27)
Soybean	-1.00 (-1.91)	1.35 (2.22) +	0.42 (1.01)
Mushroom	-0.69 (-7.64) -	0.07 (0.55)	0.43 (4.81) +
Splice	-0.08 (-0.62)	-	1.33 (10.28) +

poorly on 2 and as well as K2-AS on 1 database. As far as K2-AS< was concerned, CIG-AS was significantly better on 2, significantly worse on 1 and same as K2-AS< on 2 of the databases.

As compared to Bayesian networks induced using all attributes by the CB algorithm, CGR-AS and CDC-AS perform significantly better on 3 and 2 databases, respectively, significantly worse on 2 and perform as well on the remaining databases. CIG-AS on the other hand is significantly outperformed by CB on 3 of the 6 databases but performs better on 1 and as well on the remaining 2 databases.

Table 10: Comparison with C4.5.

	Chess	Voting	Voting1
C4.5	98.37 \pm 0.62	92.21 \pm 5.04	86.87 \pm 2.64
CIG-AS	96.33 \pm 0.90 -	94.87 \pm 0.82 +	89.94 \pm 2.08 +
CGR-AS	96.23 \pm 0.87 -	95.38 \pm 1.22 +	90.30 \pm 1.53 +
CDC-AS	96.42 \pm 0.83 -	95.36 \pm 1.08 +	90.11 \pm 1.44 +

Table 11: Comparison with C4.5 (cont.).

	Soybean	Mushroom	Splice
C4.5	93.10 \pm 1.79	99.51 \pm 0.36	94.00 \pm 0.94
CIG-AS	84.50 \pm 3.90 -	99.43 \pm 0.27	94.64 \pm 0.56 +
CGR-AS	90.60 \pm 2.02 -	99.15 \pm 0.52 -	95.10 \pm 0.34 +
CDC-AS	89.20 \pm 2.81 -	99.22 \pm 0.55 -	95.08 \pm 0.39 +

5.5 Comparison with Decision Tree classifiers

For the sake of comparison, we also compared the performance of Info-AS with that of a decision tree classifier, C4.5 (as implemented in the IND version 2.1 software). Tables 10 and 11 show the predictive accuracies (averaged over 20 trials) of the various methods as well as the statistical significance of the results using the paired t -test. A + sign indicates that Info-AS (using the corresponding metric) was significantly better than C4.5 on the database under consideration. Likewise, a – sign indicates that C4.5 was better. CGR-AS was significantly better than C4.5 on 3 databases and significantly worse on 2 databases. On one database, there was no significant difference. Both CIG-AS and CDC-AS were significantly better on 3 databases. However, they were also significantly worse on 3 databases.

5.6 Discussion

Our experimental results show that Info-AS performs significantly better than both naive-AS as well as naive-ALL on almost all the databases considered, demonstrating that the extra modeling power of Info-AS over the naive-classifier induction methods actually makes a difference in practice. This improved performance is due to the different conditional independence assumptions that the two approaches make. The naive Bayesian classifier assumes that the attributes are independent of each other, given the class variable. However, this assumption is not normally valid in the real world. Many attributes in a given domain will not be independent, and these violations will degrade the accuracy of the naive Bayesian classifier. On the other hand, Bayesian networks take into account these dependencies, and should have a better performance than the naive classifiers, especially in domains where the attributes are highly correlated.

As regards to the other Bayesian network induction methods, the results show that Info-AS performs better or at least as well as these methods on most of the databases considered. The important point to note here with respect to K2-AS is the tremendous improvement in the time-complexity of the two algorithms. Whereas

the node-selection phase of the K2-AS algorithm is computationally intensive because Bayesian inference is performed repeatedly at every stage in order to decide which attribute to select next, the node-selection phase of Info-AS is polynomial in the number of attributes.

Moreover, Info-AS generates smaller networks that display predictive accuracy comparable to that of Bayesian networks which model all attributes. By selecting a subset of attributes prior to learning the network, Info-AS greatly enhances the inference efficiency (which is a \mathcal{NP} -hard problem) of the resulting networks.

As far as the three metrics themselves are concerned, there was almost no difference statistically between CGR and CDC (except on the soybean database in which CGR was statistically better). CIG performed as well as the others on some of the databases tested, while being significantly poorer on the rest. This implies that the metrics based on ratios (CGR and CDC) perform better than the metric based purely on the gain (CIG).

We also studied various modifications of the Info-AS algorithm proposed in this paper.

One of the modifications we studied was to learn a *naive* Bayesian classifier from the set of nodes selected during the node-selection phase of Info-AS. However, the accuracy of the *naive* classifiers was significantly less than that of Bayesian network classifiers learned from the selected attributes.

Another strategy we examined was to use forward sequential search using the information metric to guide the search process. In other words, at each stage we considered adding the node which maximized the information metric to the set of already selected nodes. This node was added only if it resulted in an increase in the predictive accuracy of the network constructed from the resulting set of nodes. We considered two stopping criteria - stopping when the addition of no new attribute improves the classification accuracy, or stopping when the addition of any attribute would degrade accuracy. Our experimental results indicate that this method has very poor performance as compared to Info-AS.

Yet another strategy we studied was to select a subset of the attributes using

Info-AS and then use backward sequential elimination to eliminate some of the selected attributes. This is similar to *pruning* in decision tree induction. However, our experimental results show that this method also resulted in a decreased predictive accuracy as compared to Info-AS.

6 Future Work

Though our results are fairly encouraging, a number of things must be addressed. Firstly, as pointed out earlier, the CB algorithm may not be the best algorithm since we are interested in maximizing predictive accuracy and not just fitting the data on hand. Cowell et. al. [4] describe “global monitors” to measure the predictive quality of a Bayesian network. We intend to extend that approach to learn Bayesian networks so as to maximize their predictive accuracy.

Another interesting issue stems from what has been known as the ‘curse of dimensionality’. As the set of selected nodes grows larger, the number of possible instantiations of the nodes in this set increases exponentially; however, the actual number of instantiations is limited by the size of the database. The result is that the contingency table has a large number of cells with very low frequency counts. As such, as more nodes are selected, the values of the information metrics become more and more unreliable. Note that the Info-AS algorithm selects keeps selecting nodes as long as there is at least one node which gives us some information about the class node. Therefore, as the set of selected nodes grows larger, it may select attributes unnecessarily (due to unreliable values), thereby degrading performance. One way to avoid this problem, and to improve the quality of the generated classifier would be to use a statistical test (e.g. χ^2) as a stopping criterion.

Another very interesting issue is that of ‘joining’ of nodes. As pointed out earlier, Pazzani [13] has achieved a lot of success in improving the accuracy of *naive* Bayesian classifiers by joining attributes to take into account the dependencies between them. It may be possible to join attributes in the Bayesian network learned by Info-AS to yield high accuracy, *naive* Bayesian classifiers.

7 Conclusion

This paper introduces an attribute-selection approach for learning Bayesian networks using information-based metrics. We experimented with three different conditional information metrics based on Quinlan’s gain and gain ratio and Mantaras’s distance metrics. Our results show that the ratio based metrics (gain ratio and distance) perform better than the metric based purely on gain, with conditional gain ratio proving to be slightly superior on the databases tested.

We compared this selective information-based algorithm to selective and non-selective naive Bayesian approaches as well as to a non-selective Bayesian algorithm (CB) and a selective Bayesian approach (K2-AS). The Info-AS approach performs significantly better than the naive Bayesian approaches on almost all databases tested. This demonstrates that induced Bayesian networks are an improvement over naive Bayesian induction methods. This is due to the fact that Info-AS generates Bayesian networks, which model dependencies between the variables, as against the *naive* Bayesian classifiers which assume independence of the attributes (though many of them may be correlated). Note, however, that although the differences between Bayesian and naive Bayesian classifiers are statistically significant, they are not dramatic. Given the relative simplicity of naive Bayesian classifiers, it is important to note how well they perform.

One of our initial motivations for studying selective Bayesian networks is computational: inference using Bayesian networks is an \mathcal{NP} -hard problem. Our results show that by selecting a subset of attributes prior to learning the networks not only significantly improves the inference efficiency of the resulting networks, but also achieves a predictive accuracy comparable to Bayesian networks learned using the full set of attributes. Similar results were obtained with the K2-AS algorithm [15, 16]. However, the Info-AS algorithm has a polynomial subset-selection phase as compared to the exponential time complexity of the corresponding phase in K2-AS. Info-AS learns Bayesian networks which are statistically as good as or better (as far as predictive accuracies are concerned) than those learned by K2-AS, but at a much lower cost.

The Info-AS algorithm thus gives us an efficient method of learning selective Bayesian networks which seem to be a good compromise between naive Bayesian classifiers and non-selective Bayesian networks.

Acknowledgements

The first author gratefully acknowledges the use of HUGIN provided by Hugin Expert A/S, Denmark for his PhD research. We also thank Wray Buntine and RIACS / NASA Ames Research Center for making IND Version 2.1 available to us.

References

- [1] S.K. Andersen, K.G. Olesen, F.V. Jensen, and F. Jensen. HUGIN—a Shell for Building Belief Universes for Expert Systems. In *Proc.IJCAI*, pages 1080–1085, 1989.
- [2] W.L. Buntine and T. Niblett. A further comparison of splitting rules for decision-tree induction. *Machine Learning*, 7, 1992.
- [3] G.F. Cooper and E. Herskovits. A Bayesian Method for the Induction of Probabilistic Networks from Data. In *Machine Learning 9*, pages 54–62, Kluwer, 1992.
- [4] Cowell, R.G., Dawid, P. and Spiegelhalter, D.J. (1993). Sequential model criticism in probabilistic expert systems. *IEEE Transactions of Pattern Analysis and Machine Intelligence*, **15**(3), 209–219.
- [5] Heckerman, D., Geiger, D. and Chickering, M. (1994). Learning Bayesian networks: the combination of knowledge and statistical data. Technical report MSR-TR-94-09, Microsoft research, Redmond, WA.
- [6] E. Herskovits. Computer-based probabilistic-network construction. Doctoral dissertation, Medical Information Sciences, Stanford University, Stanford, CA., 1991
- [7] E. Herskovits and G.F. Cooper. KUTATO: An Entropy-Driven System for Construction of Probabilistic Expert Systems from Databases. In *Proc. Conf. Uncertainty in Artificial Intelligence*, pages 54–62, 1990.
- [8] I. Kononenko. Comparison of Inductive and noise Bayesian Learning Approaches to automatic knowledge acquisition. In B. Wielinga et al. *Current Trends in Knowledge Acquisition*, Amsterdam, IOS Press, 1990.
- [9] P. Langley. Induction of Recursive Bayesian Classifiers In *Proc. European Conf. on Machine Learning*, pages 153–164. Springer Verlag, 1993.

- [10] P. Langley and S. Sage. Induction of selective Bayesian classifiers. In *Proc. Conf. on Uncertainty in AI*, pages 399–406. Morgan Kaufmann, 1994.
- [11] R. Lopez de Mantaras. A distance-based attribute selection measure for decision tree induction. *Machine Learning*, 6, 81–92, 1991.
- [12] P.M. Murphy and D.W. Aha. UCI Repository of Machine Learning Databases. Machine-readable data repository, Dept. of Information and Computer Science, Univ. of California, Irvine.
- [13] M. Pazzani. Don't be so naive: Detecting dependencies when learning Bayesian Classifiers. Submitted for publication.
- [14] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. San Mateo, CA: Morgan Kaufmann, 1988.
- [15] G. Provan and M. Singh. Learning Bayesian Networks using Feature Selection. To appear in *Lecture Notes in Statistics*, (ed. Fisher, D. and Lenz, H.), Springer Verlag, New York.
- [16] G. Provan and M. Singh. Learning Bayesian Networks using Feature Selection. In *Working Notes Fifth International Workshop on Artificial Intelligence and Statistics*, pages 450-456, 1995.
- [17] J. Quinlan. Induction of decision trees. *Machine Learning*, 1, 81–106, 1986.
- [18] M. Singh and G. Provan. A Comparison of Induction Algorithms for Selective and non-Selective Bayesian Classifiers. In *Proceedings of the 12th International Conference on Machine Learning*, 497-505.
- [19] M. Singh and M. Valtorta. Construction of Bayesian Network Structures from Data: a Brief Survey and an Efficient Algorithm. *International Journal of Approximate Reasoning*, 12, 111–131, 1995.

- [20] M. Singh and M. Valtorta. An Algorithm for the Construction of Bayesian Network Structures from Data. In *Proc. Conf. Uncertainty in Artificial Intelligence*, pages 259–265, Morgan-Kaufmann Publishers, 1993.
- [21] A. White and W. Liu. Bias in information-based measures in decision tree induction. *Machine Learning*, 321–329, 1994.

Chess Database

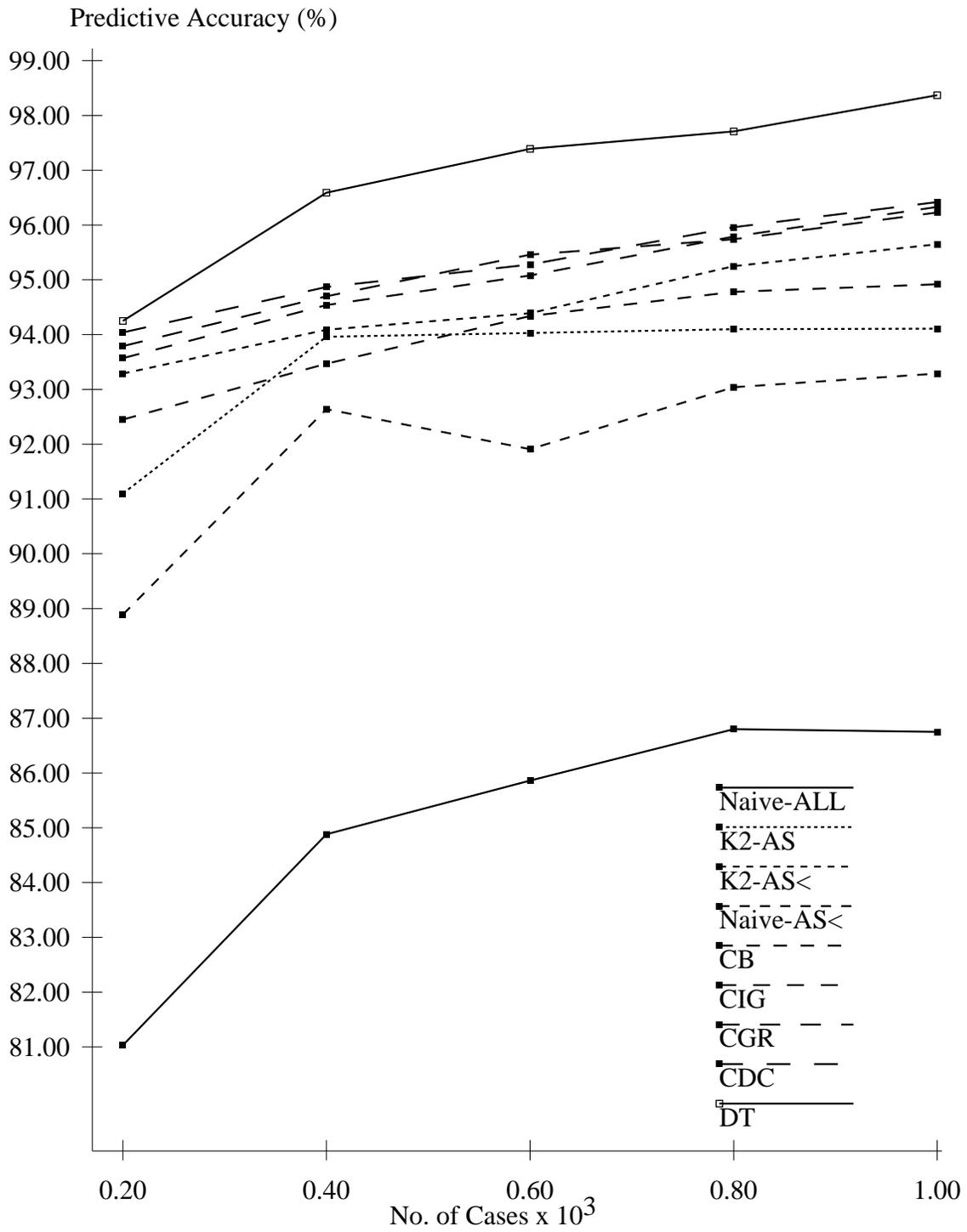


Figure 1: Learning curves for the Chess database.

Mushroom Database

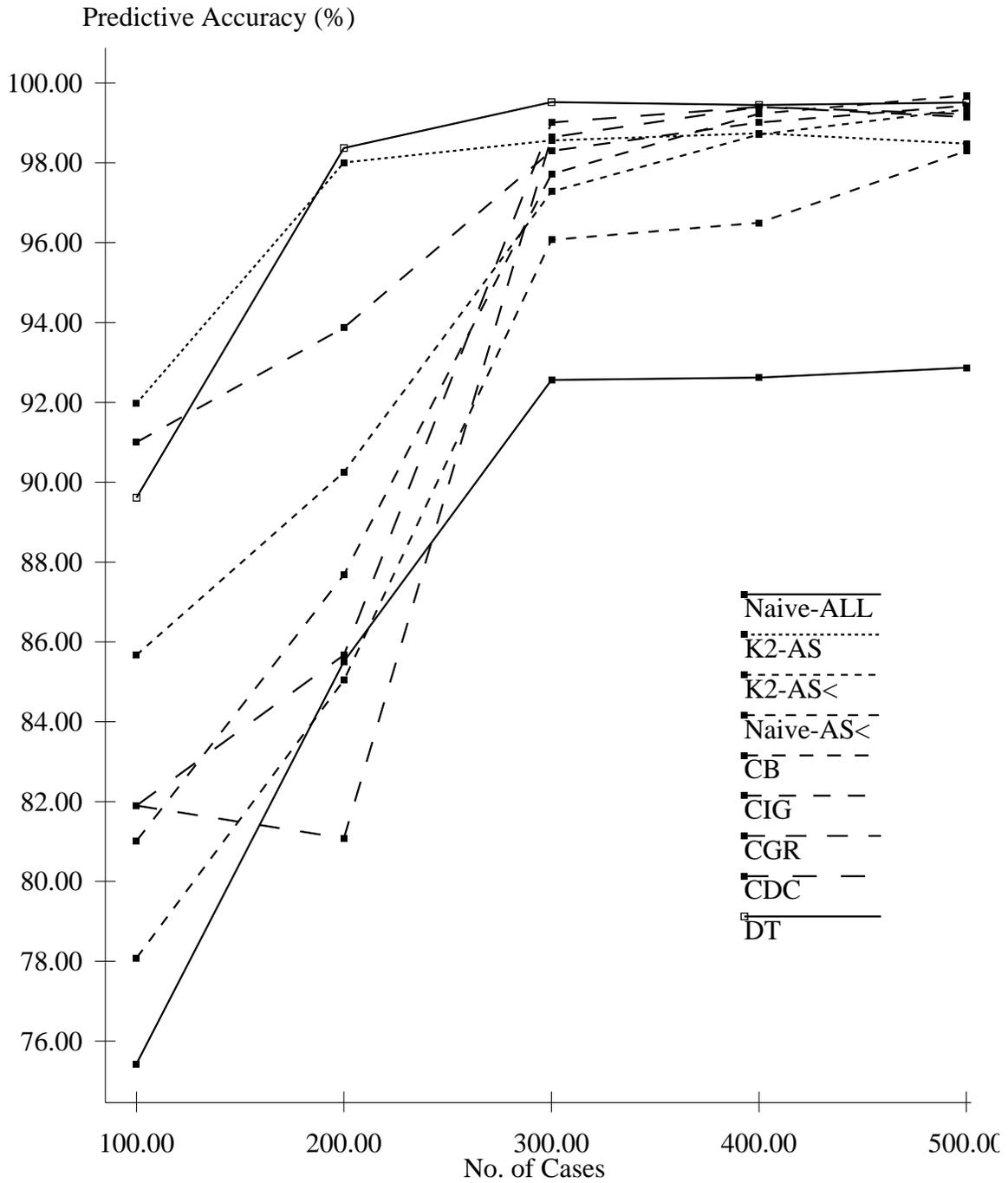


Figure 2: Learning curves for the Mushroom database.

Soybean Database

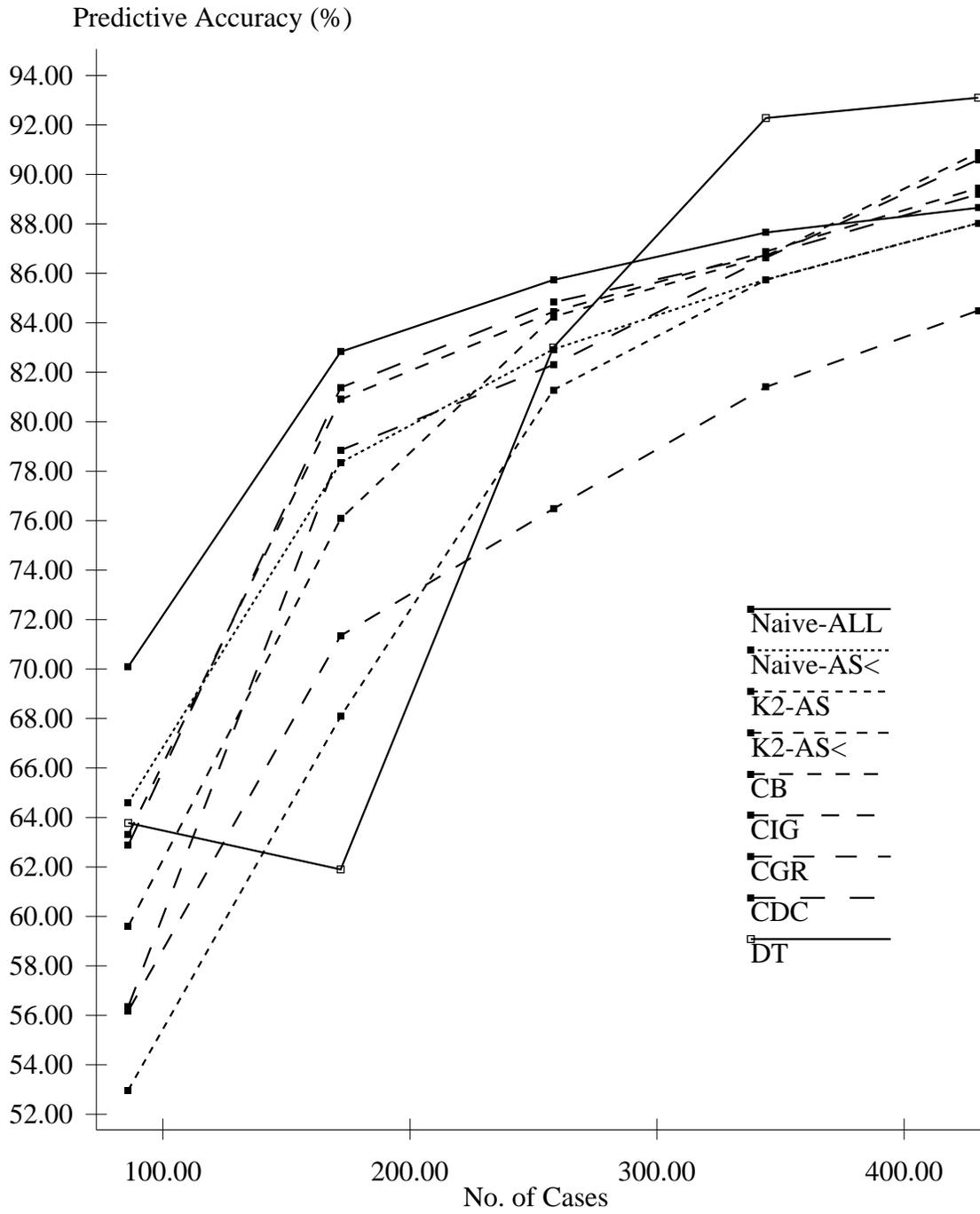


Figure 3: Learning curves for the Soybean database.

Voting Database

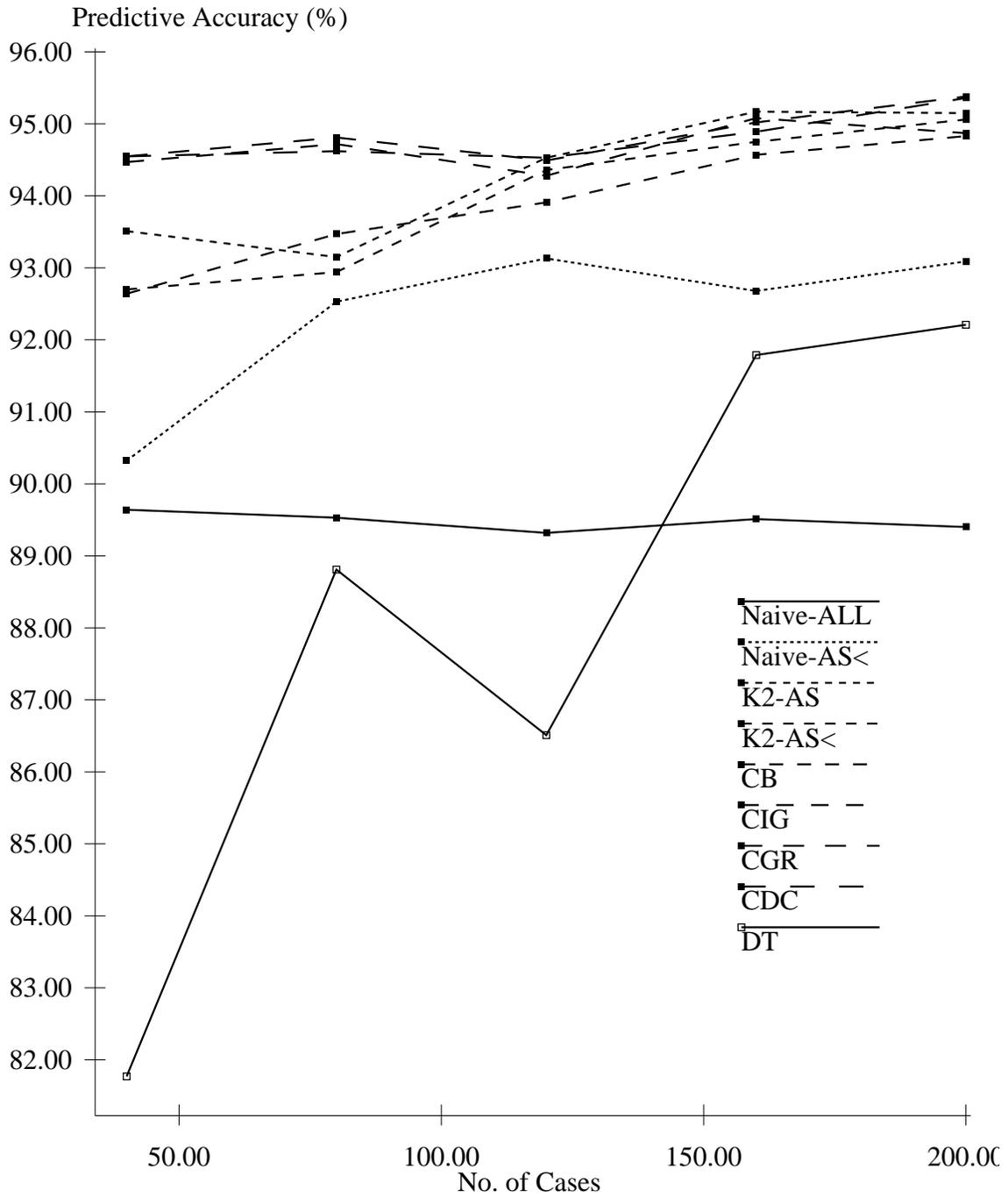


Figure 4: Learning curves for the Voting database.