Learning Bayesian Networks Using Feature Selection

Gregory M. Provan*
Institute for Decision Systems Research
4984 El Camino Real, Los Altos, CA, 94022
< provan@camis.stanford.edu >

Moninder Singh
Dept. of Computer and Information Science
University of Pennsylvania
Philadelphia, PA 19104-6389
< msingh@gradient.cis.upenn.edu >

Abstract

This paper introduces a novel enhancement for learning Bayesian networks with a bias for small, high-predictive-accuracy networks. The new approach selects a subset of features which maximizes predictive accuracy prior to the network learning phase. We examine explicitly the effects of two aspects of the algorithm, feature selection and node ordering. Our approach generates networks which are computationally simpler to evaluate and which display predictive accuracy comparable to that of Bayesian networks which model all attributes.

1 INTRODUCTION

Bayesian networks are being increasingly recognized as an important representation for probabilistic reasoning. For many domains, the need to specify the probability distributions for a Bayesian network is considerable, and learning these probabilities from data using an algorithm like K2 [8][1] could alleviate such specification difficulties.

We describe an extension to the Bayesian network learning approaches introduced in K2. Rather than use all database features (or attributes) for constructing the network, we select a subset of features that maximize the predictive accuracy of the network. Then the learning process uses only the selected features as nodes in learning the Bayesian network. Our goal is to construct networks which are simpler to evaluate, but which still have high predictive accuracy relative to networks which model all features. We examine explicitly the effects of two aspects of the algorithm: (a) feature selection, and (b) node ordering.

Our experimental results verify that this approach generates networks which are computationally simpler to evaluate and which display predictive accuracy comparable to that of Bayesian networks which model all features. Our results, similar to those observed by other studies of feature selection in learning [6, 13, 17, 18], demonstrate that feature selection provides comparable predictive accuracy using smaller networks. For example, by selecting as few as 15% of the features for the gene-splice domain, we obtained a predictive accuracy of 94.8% (as opposed to 96.8% with all features). However, the reduction of features is not always as significant (e.g. 71% for the letter recognition domain), which implies that feature selection should be used advisedly, and its effect is domain- and data-dependent.

The remainder of the paper is organized as follows. Section 2 introduces the Bayesian network learning algorithm which we modify, the K2 algorithm. Section 3 describes our new learning approach. Section 4 outlines the experimental design, and Section 5 summarizes the experimental results. Section 6 compares and contrasts our approach with other related work. Finally, we summarize our contributions in Section 7.

2 Bayesian Network Learning

We now define the learning algorithm used in K2.[2] Assume that we have a database D of m cases, where each case contains an instantiation for each of a set Z of n discrete features. B_S denotes a belief network structure representing the features in Z. In a belief network, a node represents a feature, and the absence of an arc between two nodes denotes the independence of the two nodes given the remaining network structure. The posterior prob-

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

[1] K2 is a Bayesian reformulation of the Kutato learning algorithm [12].

[2] We use the nomenclature used in the papers on K2 by Herskovits and Cooper [12, 8].

ability of a network given the data, $P(B_S|D)$, is proportional to the joint probability, so networks can be ranked according to their joint probabilities. The single most likely network is given by

$$B_{S_{max}} = \text{argmax}_{B_S}[P(B_S|D)].$$

K2 constructs a Bayesian network from a set of features as follows. K2 selects the network (out of a set of possible networks exponential in the number of network nodes) which maximizes the network's posterior probability, $P(B_S, D)$. K2 requires an ordering on the features from which the network will be constructed. Given an ordering $n_1, n_2, ..., n_m$ of the m features, K2 takes each successive feature in the ordering, adds it as a node n_i in the network, and creates parents for n_i in a greedy fashion: rather than evaluate all subsets of network nodes $n_1, n_2, ..., n_{i-1}$ as parent nodes, K2 selects as a parent node the the *single* node in $\{n_1, n_2, ..., n_{i-1}\}$ which most increases the posterior probability of the network structure. New parent nodes are added incrementally to n_i as long as doing so increases the posterior probability of the network given the data.

Our new approach proceeds in two phases. The first phase computes a subset $\Delta \subseteq Z$ of features that generates the network B_ζ with highest predictive accuracy, where B_ζ denotes the network formed from the subset $\zeta \subseteq Z$ of features. The second phase computes the network (from the set of features Δ) which maximizes the predictive accuracy over the test data.

The learning algorithm that we use, called CB, is a modified version of K2 [24]. Whereas K2 assumes a node ordering, CB uses conditional independence (CI) tests to generate a "good" node ordering, and then uses the K2 algorithm to generate the Bayesian network from the database D using this node ordering. The CB algorithm starts by using CI tests of order 0 and keeps constructing networks for increasing orders of CI tests as long as the predictive accuracy of the generated network keeps increasing. Since CB uses the K2 algorithm to generate the Bayesian network from a particular ordering, CB is correct in the same sense that K2 is [24]. Singh and Valtorta show the importance of deriving a good node ordering [24], given the $n!$ possible node orderings on n features.

3 Feature Selection Algorithm

We implemented the Feature Selection Algorithm using the CB algorithm in both the node selection as well as the network construction phase. We call this approach K2-AS, since it uses the basic K2 algorithm allied with *A*ttribute *S*election in the node selection phase. The algorithm we use is what has been described as a *wrapper model* [13], in that "the feature subset selection algorithms conducts a search for a good subset using the induction algorithm itself as part of the evaluation function" [13, page 124].

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. 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.

The algorithm used for the node selection phase is a forward selection algorithm, in that it starts with an empty set of features and adds features 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, K2-AS chooses the set of attributes Δ, $\Delta \subseteq Z$ (Z is the set of all attributes) from which the final network is constructed. The algorithm starts with the initial assumption that Δ consists of only the class variable $class_{var}$. It then adds incrementally that attribute (from $Z - \Delta$) whose addition results in the maximum increase in the predictive accuracy of the network constructed from the resulting set of nodes. When there is no single attribute whose addition increases predictive accuracy, the algorithm stops adding attributes. We define $\Phi(\Delta)$ to be the predictive accuracy of the set Δ of attributes, and $\mathcal{G}(M)$ to be the network constructed from the set M of nodes. This phase can be described as follows:

```
$\Delta \leftarrow \{class_{var}\}$
$\Phi_{old} \leftarrow \Phi(class_{var})$
NotDone ← True
while NotDone do
    $\forall\, x \in Z - \Delta$, let $B_{S_x} \leftarrow \mathcal{G}(\Delta \cup \{x\})$
    $\Phi_{new} \leftarrow \max_x \Phi(B_{S_x})$
    $z = \arg\max_x\{\Phi_x\}$
    if $\Phi_{new} > \Phi_{old}$ then
        $\Phi_{old} \leftarrow \Phi_{new}$
        $\Delta \leftarrow \Delta \cup \{z\}$
    else NotDone ← false
end {while};
```

- **network construction phase:** K2-AS uses the final set of nodes Δ selected in the node selection phase to construct a network using training data. Once again, the CB algorithm generates networks for increasing orders of CI tests as long as the predictive accuracy (on evaluation data) keeps increasing and stops when there is no further increase. The network

Table 1: Summary of databases and learning approaches. The first two columns summarize the total number of features and the nodes selected by K2-AS. The last four columns summarize information about how we split the databases.

	Features	Nodes Selected	Cases (TOTAL)	Cases in *training* set	Cases in *evaluation* set	Cases in *test* set
Gene-splice	61	10	3175	1000	1000	1175
Soybean	36	12	640	290	340	340
Chess	37	6	3196	1000	1000	1196
Letter	17	12	20000	10000	6000	4000

corresponding to the maximum predictive accuracy is the final network.

- **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 Experimental Design

We divide the database into three parts. The first two parts are used for the node selection phase: the first part, the *training* data, is used for learning the network (from the current subset of nodes and the node under consideration); the second part, the *evaluation* data, is used to test it for predictive accuracy (to decide whether to add the new node to the set of selected nodes). Once we have finished with the selection of the subset of nodes, we use the first part of the database to learn the network using the CB algorithm (the network construction phase). The third part of the database, the *test* data, is then used for determining the predictive accuracy of the network derived from the network construction phase. We performed inference on the networks using the Lauritzen-Spiegelhalter inference algorithm as implemented in the HUGIN [3] system.[3]

The K2-AS approach trades off the time required to construct a network from the full feature set (as done in K2) with precomputing a feature subset and subsequently constructing a network with this feature subset. The node selection phase in K2-AS adds a modest amount of computational expense to the network induction. At each iteration of the node selection phase, K2-AS constructs a network (for each of the nodes not in the current set of selected nodes) by adding each node to the current set of selected nodes and then perform inference using the constructed network. At each stage, since a very small subset of the features is used, the network generated is very small and so the network construction as well as the inference phase is very fast (a matter of seconds for the databases considered).

Note that the node selection phase depends on selecting a relatively small number of features to create Bayesian networks that are efficient to perform inference on. Our results confirm the success of this approach, in that no more than a dozen features were selected in each of four domains studied.

5 Results

We have performed a set of experiments to compare the networks generated by our approach with those created by CB. We tested this method on four databases acquired from the University of California, Irvine Repository of Machine Learning databases [21], namely Michalski's Soybean database, Slate's Letter Recognition database, the Gene-Splicing database due to Towell, Noordewier, and Shavlik,[4] and Shapiro's Chess Endgame database.

Table 2: Comparison of predictive accuracy for the basic CB approach and for K2-AS.

	Basic CB	K2-AS
Chess	95	94.65
Gene-splice	97	94.81
Soybean	86.2	93.83
Letter recognition	82.5	82.85

Table 1 summarizes the four databases used in terms of number of cases and features, and compares the number of features with the number of nodes selected.[5] The K2-AS approach always selects fewer features to be nodes in the network.

[3] We gratefully acknowledge that HUGIN has been kindly supplied to the second author for doctoral training.

[4] We used the database used to recognize genes in DNA Sequences (which we call gene-splicing database), as created by Towell, Noordewier, and Shavlik.

[5] In the case of the Chess database and the Gene-splice database, we mixed up the cases in the database prior to splitting it because the database had all cases for one value of the class variable first, then all cases for the next value and so on. In the case of the Soybean database, due to less number of cases, both the *evaluation* and the *test* parts of the database consisted of the same 340 cases.

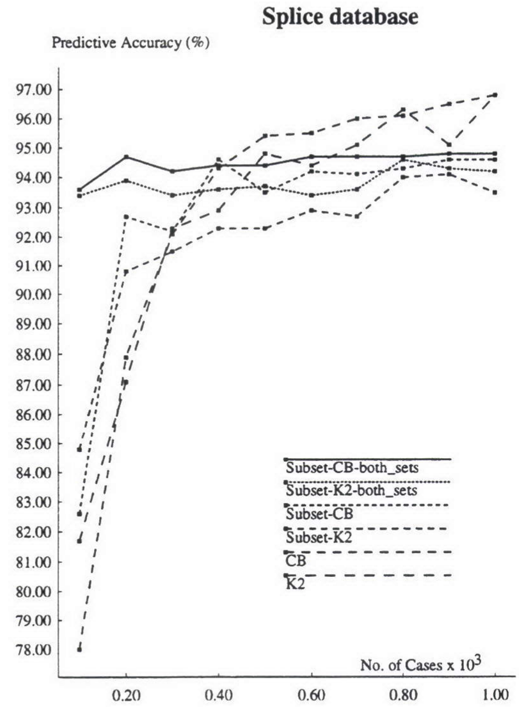

Figure 1: Learning curves for Gene-splice database

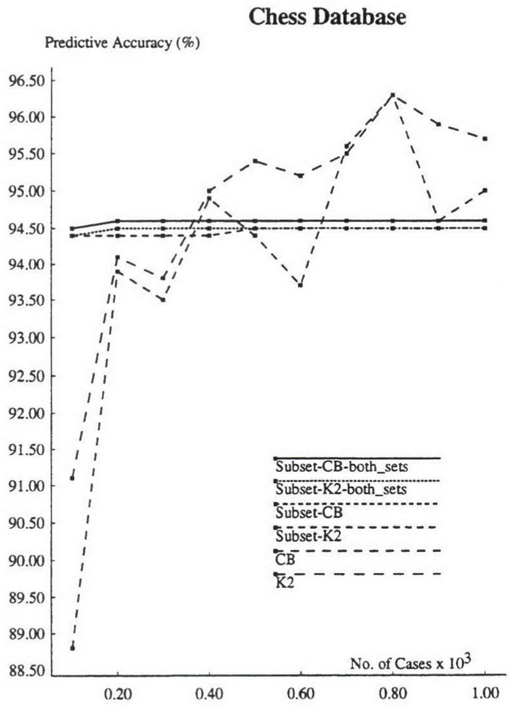

Figure 2: Learning curves for Chess database

The ratio of nodes selected (excluding the node for the class variable) to total database features ranges from 15% for the gene-splice domain to 71% for the letter recognition domain.

Table 2 compares the predictive accuracy obtained for the basic CB approach and for the K2-AS approach. In 2 out of the 4 cases, the K2-AS approach created a network with *higher* predictive accuracy than the CB approach.[6]

In the Gene-splice case the predictive accuracy was smaller by just over 2%, but the network which had 2% lower predictive accuracy was one sixth the size of the full network created by the CB approach: i.e., of the 61 features, 10 were deemed to be important to the predictive accuracy, resulting in a network one-sixth the size of the network generated using all 61 features. Note also that the time required for performing inference on the networks induced using K2-AS was substantially less than than required for the networks induced using CB. This was particularly true for the network for the chess domain, which is very densely connected.

Figures 1 and 2 show the learning curves for the Splice and Chess databases.[7] In these graphs K2 was given the node ordering selected by CB which resulted in a network with the best predictive accuracy, which we call the *pseudo-optimal*, and CB derived an ordering from the data. We denote the feature selection algorithms using the "subset" prefix.[8]

Since the number of attributes used by the subset-selection algorithm for these two domains is small relative to the total number of attributes (approximately 16%), few cases are needed to learn the networks, and the learning curve is almost flat. In addition, the predictive accuracy of the reduced-feature networks is within 2% of the networks with all attributes. Hence, in these domains K2-AS significantly reduced network size with little loss of predictive accuracy.

Figures 3 and 4 show the learning curves for the Soybean and Letter-recognition databases. The number of attributes used by the subset-selection algorithm for these two domains is 33% and 71% respectively. Since the reduced-attribute networks are closer in size to the full-attribute networks, the learning rates are comparable, and a modest reduc-

[6] The best results we ever got for the Soybean domain with CB were 86%. Herskovits [11], even with his multiscore algorithm (using multiple networks for inference), got about 86%. As a point of comparison, in the chess endgame domain decision trees are able to obtain 99% accuracy.

[7] A learning curve plots predictive accuracy versus number of test cases.

[8] The curves with suffix "both-sets" refer to predictive accuracies obtained by using the first two parts of the database for prior probabilities, as opposed to using just the first part.

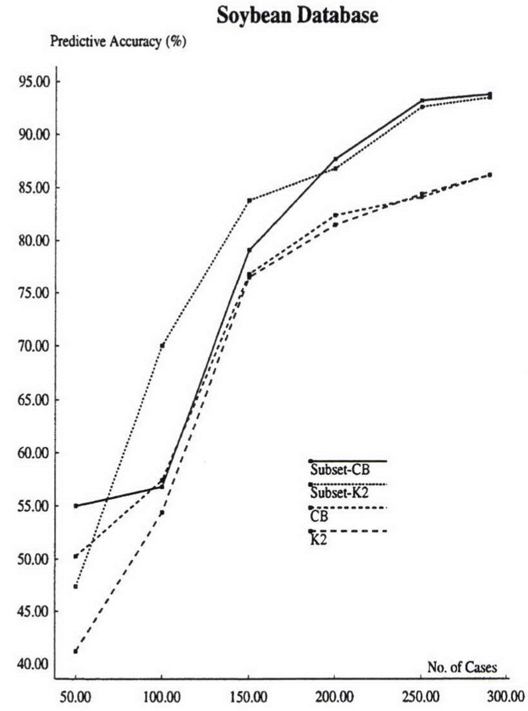

Figure 3: Learning curves for Soybean databases

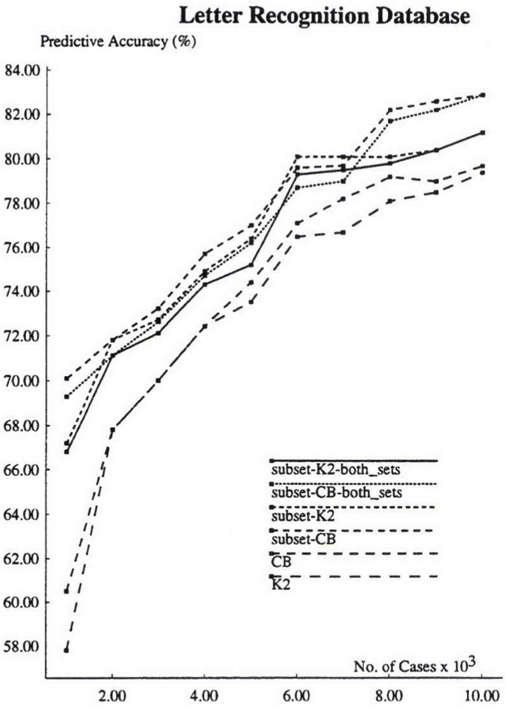

Figure 4: Learning curves for Letter-recognition database

tion in network size is possible.

Figure 5 shows the learning curves for the soybean database for a variety of node orderings: five random orderings, the average of these five orderings, the ordering computed by CB (subset-CB), and the pseudo-optimal ordering for K2 (subset-K2). This figure shows that although there is some variability in predictive accuracy and learning rate for random orderings, the final predictive accuracy is relatively insensitive to the ordering chosen: the predictive accuracy results for all but one of the random orders are within 2% of the predictive accuracy of the pseudo-optimal ordering.

We also compared the nodes selected by these eight orderings (random, CB-derived and pseudo-optimal), as shown in Table 3.

Table 3: Nodes always selected (regardless of ordering) over multiple random orderings

	Nodes	Nodes Chosen	Nodes Always Chosen
Gene-splice	61	10	7
Soybean	36	12	7
Chess	37	6	5

Table 3 shows that each ordering contained an identical set of nodes; we call these the relevant nodes. For example, for the soybean database, 7 of the 12 nodes selected were common over all the runs using random orderings. In addition, of the networks nodes which are not always selected, there was a subset that was selected in most runs using random orderings; we call these weakly relevant nodes. This approach thus provides an empirical method for determining relevance of nodes in a Bayesian network.

6 Related Work

Feature selection has been widely used in statistics and pattern recognition, and its use within the computational learning community has become quite widespread within the last few years. In statistics, research on feature selection has focused primarily on selecting a subset of features within linear regression. Techniques developed include sequential backward selection [20], branch&bound [22], and search algorithms [23, 25]. A 1993 meeting of the Society of AI and Statistics was dedicated to papers on "Selecting Models from Data" [7], and contains a large number of papers on feature selection. This statistical approach to subset selection shares many principles with other statistical notions of information minimality, like MDL. For example, Dawid discusses the close relation between subset selection and the MDL principle in [9].

The computer vision community has studied feature selection for over 20 years [10], and has formed a

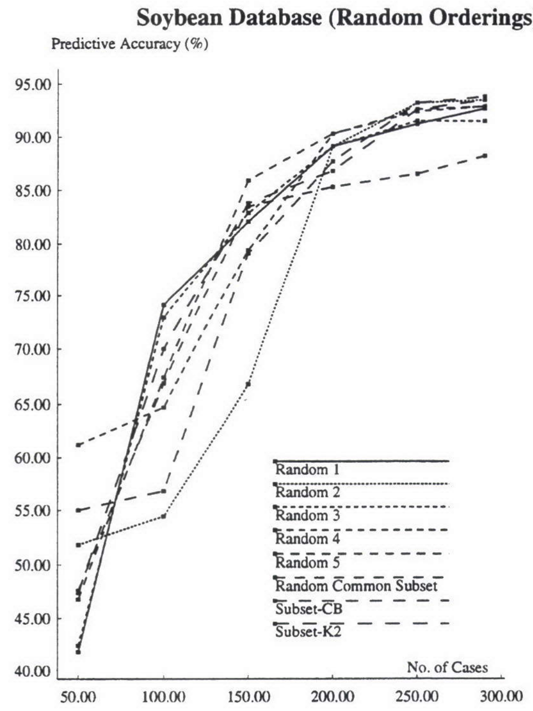

Figure 5: Learning curves for Soybean database testing the effect of node orderings

sub-community within computational vision called pattern recognition. The mainstream vision community typically does not use statistical feature selection to identify relevant features, but makes assumptions about what the features should be present to represent, for example, a class of objects.

Feature selection has received considerable attention in the last few years within the computational learning community, using both filter-based and wrapper-based approaches [13]. A filter model filters out less relevant features using an algorithm different from the induction algorithm used for the learning, and a wrapper model uses induction algorithm itself for feature selection. Three filter-model approaches that have been taken are: the FOCUS algorithm [2] the Relief algorithm [14, 15] (which Kononenko has extended in [16]), and an extended nearest-neighbor algorithm [5]. Wrapper-based approaches have been studied in [13, 6, 18], among others.[9]

A growing consensus in this research is that the success of feature selection is strongly correlated to the data itself, as well as to the algorithm employed. Many domains studied (for example, the domains described in the University of California, Irvine database) have a relatively small number of features, namely the features have been pre-selected for their relevance. It is expected that in such domains feature selection may not make a significant impact. One exception is the study of cloud classification by Aha and Bankert [1], in which a set of 204 attributes were significantly pruned, leading the greatly improved performance.[10] Better understanding of data sets and of domains may lead to a deeper understanding of the role of feature selection, and improved performance from feature selection algorithms.

This paper does not attempt to achieve the accuracy found on the four databases studied using other approaches, such as decision trees. Instead, the purpose of this paper is *not* to identify the Bayesian network with the highest predictive accuracy, but to identify a parsimonious model with *good* predictive accuracy. It is possible to compute multiple models and average over them (e.g. as proposed in [19, 4]) to obtain the best predictive accuracy, and we hope to take this approach in future work. In addition, we restrict our attention to Bayesian networks. To fairly compare the best possible predictive accuracy of other approaches to the predictive accuracy obtained using reduced-attribute Bayesian networks is not useful; rather, the averaged-model approach should provide a predictive accuracy for such comparison.

7 Conclusion

This paper introduces a feature-selection approach for learning Bayesian networks using a greedy search (i.e. K2-based) algorithm called CB. Selecting a subset of features prior to learning the networks significantly improves the inference efficiency of the resulting networks, and achieves a predictive accuracy comparable to networks learned using the full set of attributes.

We believe that the benefits of having a simpler network (which greatly reduces the inference time) outweigh the slight reduction in predictive accuracy and the one-time cost incurred during network construction, especially in cases where the network generated using the entire set of features may be too large to even allow inference.

In addition, we have showed that the predictive accuracy of the learned Bayesian network is relatively insensitive to node ordering, and that this approach can identify the most relevant subsets of nodes in a Bayesian network.

[9] Langley [17] presents a thorough review of feature selection approaches studied within the Machine Learning literature.

[10] In this domain it is likely that the features were not pre-selected for relevance, as there was no *a priori* knowledge of relevance and irrelevance.

References

[1] D.W. Aha and R.L. Bankert. Feature selection for case-based classification of cloud types. In *AAAI Workshop on Case-based Reasoning*, pages 106–112, Seattle, WA, 1994. AAAI Press.

[2] H. Amuallim and T.G. Dietterich. Learning with Many Irrelevant Features. In *Proc. Conf. of the AAAI*, pages 547-552. AAAI Press, Menlo Park, CA, 1991.

[3] 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.

[4] W.L. Buntine and T. Niblett. A further comparison of splitting rules for decision-tree induction. *Machine Learning*, 7, 1992.

[5] C. Cardie. Using Decision Trees to Improve Case-based Learning. In *Proc. Machine Learning*, pages 25–32. Morgan Kaufmann, 1993.

[6] R. Caruana and D. Freitag. Greedy attribute selection. In W. Cohen and H. Hirsch, editors, *Proc. Machine Learning*, pages 28–36. Morgan Kaufmann, 1994.

[7] P. Cheeseman and W. Oldford, editors. *Selecting Models from Data: AI and Statistics IV.* Springer-Verlag, 1994.

[8] G.F. Cooper and E. Herskovits. A Bayesian Method for the Induction of of Probabilistic Networks from Data. In *Machine Learning 9*, pages 54–62, Kluwer, 1992.

[9] A.P. Dawid. Prequential Analysis, Stochastic Complexity and Bayesian Inference. In J.M. Bernardo, J. Berger, A. Dawid, and A. Smith, editors, *Bayesian Statistics 4*, pages 109–125. Oxford Science Publications, 1992.

[10] P. Dejviver and J. Kittler. Pattern Recognition: A Statistical Approach. Prentice-Hall, 1982.

[11] E. Herskovits. Computer-based probabilistic-network construction. Doctoral dissertation, Medical Information Sciences, Stanford University, Stanford, CA., 1991

[12] 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.

[13] G. John, R. Kohavi, and K. Pfleger. Irrelevant features and the subset selection problem. In W. Cohen and H. Hirsch, editors, *Proc. Machine Learning*, pages 121–129. Morgan Kaufmann, 1994.

[14] K. Kira and L. Rendell. A practical approach to feature selection. In *Proc. Machine Learning*, pages 249–256, Aberdeen, Scotland, 1992. Morgan Kaufmann.

[15] K. Kira and L. Rendell. The Feature Selection Problem: Traditional Methods and a New Algorithm. In *Proc. AAAI*, pages 129–134, Minneapolis, MN, 1992. AAAI Press.

[16] I. Kononenko. Estimating attributes: Analysis and extension of relief. In *Proc. European Conf. on Machine Learning*, pages 171–182. Springer Verlag, 1994.

[17] P. Langley. Selection of relevant features in machine learning. In R. Greiner, editor, *Proc. AAAI Fall Symposium on Relevance.* AAAI Press, 1994.

[18] P. Langley and S. Sage. Induction of selective bayesian classifiers. In *Proc. Conf. on Uncertainty in AI*, pages 399–406. Morgan Kaufmann, 1994.

[19] D Madigan, A. Raftery, J. York, J. Bradshaw, and R. Almond. Strategies for Graphical Model Selection. In *Proc. International Workshop on AI and Statistics*, pages 331–336, 1993.

[20] T. Marill and D. Green. On the effectiveness of receptors in recognition systems. *IEEE Trans. on Information Theory*, 9:11–17, 1963.

[21] 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.

[22] M. Narendra and K. Fukunaga. A branch and bound algorithm for feature subset selection. *IEEE Trans. on Computers*, C-26(9):917–922, 1977.

[23] W. Siedlecki and J. Sklansky. On automatic feature selection. *Itnl. J. of Pattern Recognition and Artificial Intelligence*, 2(2):197–220, 1988.

[24] M. Singh and M. Valtorta. Bayesian Network Structures from Data. In *Proc. Conf. Uncertainty in Artificial Intelligence*, pages 259–265, Morgan-Kaufmann Publishers, 1993.

[25] L. Xu, P. Yan, and T. Chang. Best-first strategy for feature selection. In *Proc. Ninth International Conf. on Pattern Recognition*, pages 706–708. IEEE Computer Society Press, 1989.