Searching for Attribute Dependencies in Bayesian Classifiers

Michael Pazzani
Department of Information and Computer Science
University of California, Irvine
Irvine, CA 92717
pazzani@ics.uci.edu
phone: (714) 824-5888 fax (714) 824-4056

1 Introduction

The Bayesian classifier (Duda & Hart, 1973) is a probabilistic method for classification. It can be used to determine the probability that an example j belongs to class C_i given values of attributes of the example:

$$P(C_i|A_1=V_{1_j} \ \& \ ...A_n=V_{n_j})$$

If the attributes are independent, this probability can be calculated by Equation 1.

$$P(C_i) \prod_k \frac{P(C_i|A_k=V_{k_j})}{P(C_i)} \qquad [1]$$

In this form, it is well suited for learning from data, since the probabilities $\hat{P}(C_i)$ and $\hat{P}(C_i|A_k=V_{k_j})$ may be estimated from the training data. A classifier created in this manner is sometimes called a simple (Langley, 1993) or naive (Kononenko, 1990) Bayesian classifier.

On many problems, the accuracy of the naive Bayesian classifier is equal to or greater than that of more sophisticated machine learning algorithms. For example, Pazzani, Merz, Murphy, Ali, Hume and Brunk (1994) found that the accuracy of the Bayesian classifier was 55.8% at predicting the severity of heart disease from training sets of 200 examples, and 75.2% at diagnosing diabetes from training sets of 440 examples, while the accuracy of ID3, a simple decision tree algorithm (Quinlan, 1986), was 50.0% and 70.2% on the same problems. However, on other problems, the reverse pattern was observed. The naive Bayesian classifier was 95.1% accurate at identifying poisonous mushrooms (from training sets of 500 examples), while the decision tree was 99.5% accurate.[1] When learning from 600 examples of telephone network troubleshooting (Danyluk and Provost, 1993), the naive Bayesian classifier was 30.1% accurate while the decision tree was 37.3% accurate.

One possible explanation for the poor performance of the Bayesian classifier on these last two problems is that the major assumption of the classifier, that the attributes are independent does not hold. In this paper, we address this issue by searching for dependencies among pairs of attributes. For example, if there are three independent attributes, the numerator of Equation 1 would be as follows:

$$P(C_i|A_1=V_{1_j})P(C_i|A_2=V_{2_j})P(C_i|A_3=V_{3_j}) \qquad [2]$$

However, if it is known that A_1 and A_3 are not independent, then Equation 3 should be used instead:

$$P(C_i|A_1=V_{1_j} \ \& \ A_3=V_{3_j})P(C_i|A_2=V_{2_j}) \qquad [3]$$

1. All of the differences in accuracy reported here at significant at least at the .05 level using a paired t-test. The databases are available from the UCI Repository of Machine Learning Databases (Murphy & Aha, 1994).

We will say that that the two attributes A_1 and A_3 are *joined* if Equation 3 is used instead of Equation 2. A more accurate classifier can result from joining the right groups of attributes. For example, if the concept to be learned was an *exclusive-or* ($A_1 \oplus A_3$), then $P(True|A_1=0)$, $P(True|A_1=1)$, $P(True|A_3=0)$, and $P(True|A_3=1)$ would all equal 0.5. However, $P(True|A_1=0 \& A_3=0)$ and $P(True|A_1=1 \& A_3=1)$ would equal 0, while $P(True|A_1=0 \& A_3=1)$ and $P(True|A_1=1 \& A_3=0)$ would equal 1. This simple example demonstrates that there can be a major advantage in joining attributes. However, if independent attributes are joined, and the probabilities of joined attributes are estimated from training data, a less accurate classifier may result because the estimates of the joined attributes are less reliable than the estimates of individual attributes.

Kononenko (1991) uses the term "semi-naive Bayesian classifier" for systems that use equations such as Equation 3. However , in order to deal with the fact that the estimates of $P(C_i|A_1=V_{1_j})$ and $P(C_i|A_3=V_{3_j})$ are more reliable than the estimate of $P(C_i|A_1=V_{1_j} \& A_3=V_{3_j})$, it is not sufficient to simply assume that two attributes are independent if $P(C_i|A_1=V_{1_j} \& A_3=V_{3_j})$ does not equal $P(C_i|A_1=V_{1_j})P(C_i|A_3=V_{3_j})$. Kononenko proposes a method for identifying that features are not independent based upon a statistical test that determines the probability that two attributes are not independent. The algorithm joins two attributes if there is a greater than 0.5 probability that the attributes are not independent. Experimentation results with this method were disappointing. On two domains the semi-naive Bayesian classifier had the same accuracy as the naive Bayesian classifier, and on two domains the semi-naive Bayesian classifier was one percent more accurate, but it is not clear whether this difference is statistically significant.

In this paper, we explore an alternate approach to determining whether it is useful to join two attributes when constructing a Bayesian classifier. We also give experimental results on parity functions, an artificial set of functions that are particularly difficult for naive Bayesian classifiers, and results on three naturally occurring data sets.

2 Searching for Attribute Dependencies

In our work, we empirically determine when it is beneficial for a Bayesian classifier to join two attributes. First, the accuracy of the naive Bayesian classifier is found by using cross-validation on the training data. Next, for each pair of attributes, we use cross-validation to measure the accuracy of a Bayesian classifier that joins this pair of attributes. If joining two attributes results in an improved accuracy, we join the pair of attributes that results in the highest accuracy as measured by cross-validation. The process of selecting the two best attributes to join repeats until no such joining results in an improvement in accuracy. Note that it is possible that two joined attributes are later joined with additional attributes.

The complexity of joining attributes is at $O(EA^3)$ where E is the number of examples and A is the number of attributes. We have exploited several opportunities for making the process of measuring the accuracy by cross-validation more efficient. In particular, we use leave-one-out testing on the training data. This allows a single Bayesian classifier to be constructed on the entire training set. To classify each example, the contribution of that example to the probability estimates is subtracted out (Langley, 1993). In addition, it is not necessary to re-estimate the probabilities of attributes that are not joined when evaluating the impact of joining two attributes.

We have also experimented with removing attributes from the classifier. Such removal may improve accuracy if the attributes are irrelevant or, as in Langley and Sage (1994), if one attribute is correlated with another. We use the selective backward elimination (SBE) method (Kittler, 1986) for removing irrelevant attributes after first joining attributes. This process deletes the attribute whose deletion most increases accuracy (as measured by cross-validation) and terminates when no such deletion increases accuracy.

3. Experimental Results

In the first experiment, we consider learning *exclusive-or* of two attributes, a function that is very difficult for a naive Bayesian classifier. We tested *exclusive-or* functions with 2, 4, 6, and 8 irrelevant features included in the example description. For each function we ran 35 trials in which 50% of the examples were randomly selected for training and the remaining 50% were used to test the accuracy. On each training set, we tested a naive Bayesian classifier (called Independent in the legend), a naive Bayesian classifier using selective backward elimination to remove attributes (SBE), a Bayesian classifier that joins attributes (Dependency), and a Bayesian classifier that joins attributes and then removes irrelevant attributes (Dependency+SBE). Figure 1 shows the mean accuracy of the four algorithms, plotted as a function of the number of irrelevant features. Paired t-tests at the .001 level indicate that the algorithms that search for feature dependencies are significantly more accurate at each point than the corresponding algorithms that assume features are independent.

A potential weakness of the algorithm for joining attributes is that in order to join more than two features, it must first join two of the attributes and later join other attributes with

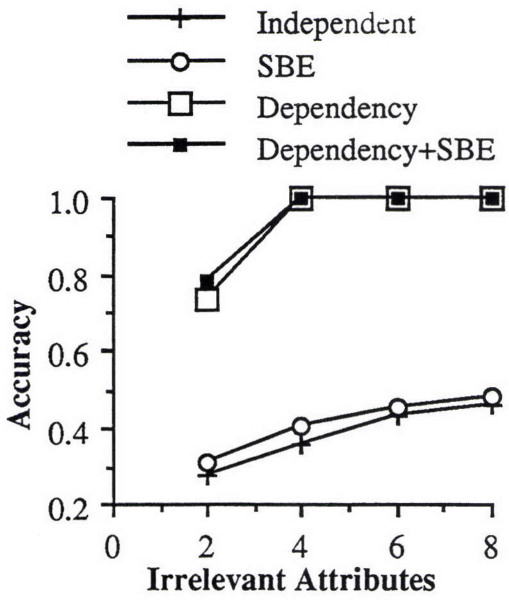


Figure 1. Learning the exclusive-or function with irrelevant attributes.

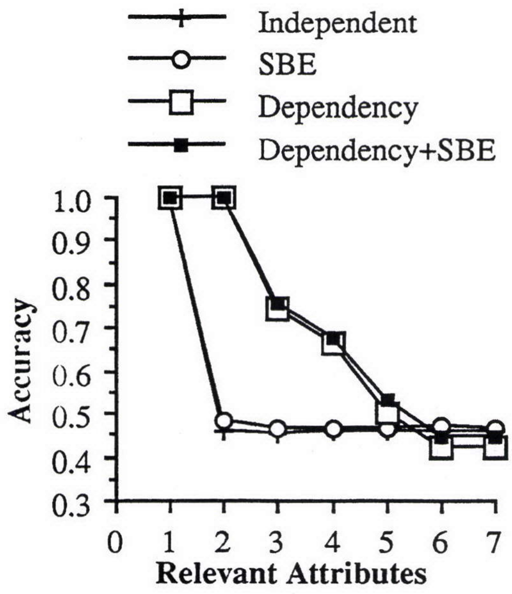


Figure 2. Learning parity functions.

the original two. This will not occur unless forming the first pair results in an increase in accuracy. In the second experiment, we evaluate the ability of the four algorithms to learn various parity function. A parity function is true iff an odd number of the attributes have a value of true. We varied the number of relevant attributes from 1 to 7 and included irrelevant attributes so that each example had a total of 10 attributes. We ran 35 trials of each function, choosing 512 examples for training and 512 examples for testing. The results displayed in Figure 2 show that the advantage of using the dependency detection algorithm does decrease when the parity function has more than two elements. However, dependency detection combined with selective backward elimination is still significantly more accurate than the naive Bayesian classifier for parity of 2, 3, 4 and 5 variables at least at the .01 level.

Next, we turn our attention to three naturally occurring databases: telephone network troubleshooting[2], identifying mushrooms, and identifying the party of a member of congress from their voting record. Figure 3 plots the the mean accuracy (N = 20) of several sized training sets of each database. To avoid clutter, we compare only the standard naive Bayesian classifier to the Bayesian classifier with dependency detection followed by selective backward elimination. In all three databases, at the maximum number of training examples tested, detecting attribute dependencies and selective backward elimination significantly improves accuracy at the .001 level. On each of these three problem, the naive Bayesian classifier was substantially less accurate than a decision tree learner. The Bayesian classifier with dependency detection followed by selective backward elimination is as accurate or more accurate than a decision tree learner.

2. The telephone troubleshooting domain contains numeric data, and the dependency detection algorithm requires nominally valued attributes. We converted the numeric data to nominal data by partitioning each numeric value into 5 partitions.

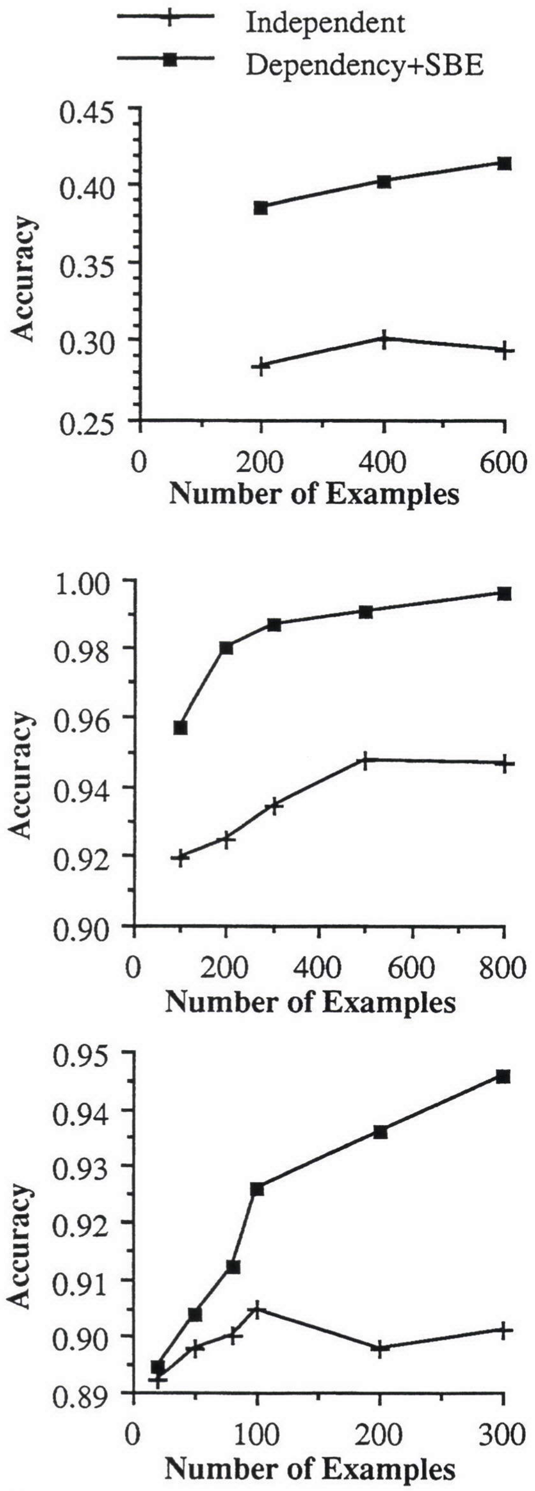


Figure 3. Accuracy on the NYNEX telephone troubleshooting problem (top), classifying mushrooms (middle) and Congressional voting (bottom).

On some problems, one would expect attempting to detect dependencies would make little or no difference. Indeed, the first point plotted in Figure 2 (parity of a single variable) shows that this is the case. We have observed this behavior on some naturally occurring data sets. Figure 4 compares the Bayesian classifier with dependency detection to the naive Bayesian classifier on the lymphography problem (each point is averaged over 20 trials). None of the differences between the algorithms are statistically significant.

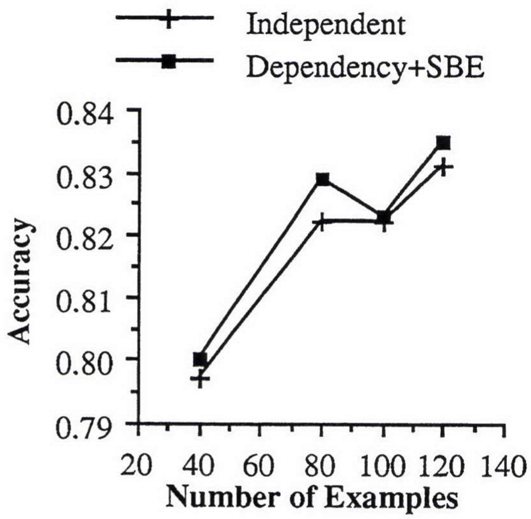
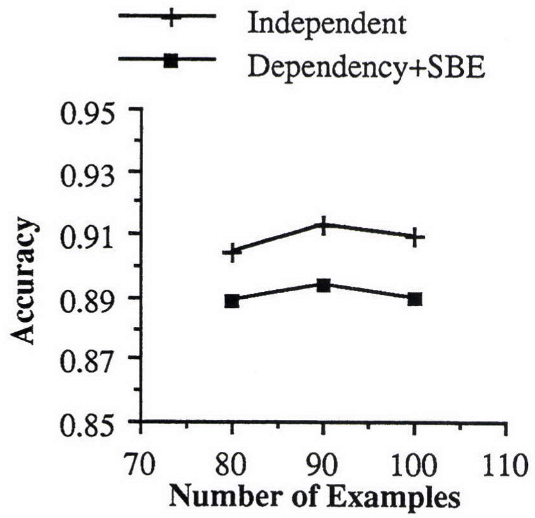


Figure 4. Accuracy on the lymphography problem

It would be gratifying if we could guarantee that detecting dependencies will never hurt the accuracy of the learner. However, we cannot offer such guarantees in general (Schaffer, 1994). In practice we have observed two domains in which finding dependencies decreases the accuracy of the classifier. Figure 5 compares the Bayesian classifier with dependency detection to the naive Bayesian classifier on the promoter problem and a foreign trade negotiation problem (averaged over 20 trials). On both of these problems, there are many attributes (57 for promoters, 43 for trade negotiations) and few examples. Under these circumstances, the chances that cross-validation detects a "spurious" dependency may be high, since it considers 1596 and 903 two feature dependencies, respectively.



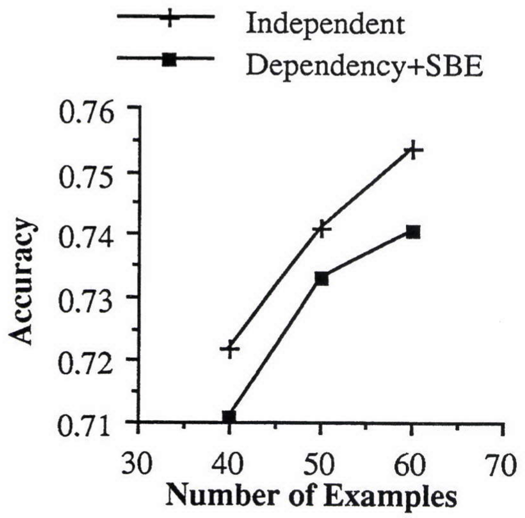


Figure 5. Accuracy on the promoters (top) and trade negotiations problems (lower).

4. Related Work

Although Kononenko (1991) has advocated detecting dependencies in Bayesian classifiers, his published results do not indicate significant or substantial increases in accuracy. Langley & Sage (1994) have used a forward selection method for eliminating attributes from Bayesian classifiers. Although they advocate using the method for correlated attributes, we feel that joining a pair of correlated attributes is more useful than deleting one of a pair of correlated attributes. The algorithm they propose would not be able to improve the accuracy on exclusive-or or parity functions.

Bayesian networks (Pearl, 1988) offer a more complex way of representing attribute dependencies. Although algorithms exist for learning Bayesian networks from data (Cooper & Herskovits, 1992) it has not yet been demonstrated that inducing Bayesian networks results in more accurate classifiers than naive Bayesian classifiers.

6. Conclusion

We have shown that when learning Bayesian classifiers from data searching for dependencies among attributes results in significant increases in accuracy. This suggests that the attributes used in some common databases are not independent and that the violations of the independence assumption that affect the accuracy of the classifier can be detected from training data.

Acknowledgements
The research reported here was supported in part by NSF grant IRI-9310413 and ARPA grant F49620-92-J-0430 monitored by AFOSR. I'd like to thank Pedro Domingos, Pat Langley and Kamal Ali for advice on Bayesian classifiers.

References

Cooper, G. & Herskovits, E. (1992). A Bayesian method for the induction of probabilistic networks from data. *Machine Learning, 9,* 309–347.

Danyluk, A. & Provost, F. (1993). *Small disjuncts in action: Learning to diagnose errors in the telephone network local loop.* Machine Learning Conference, pp 81-88.

Duda, R. & Hart, P. (1973). *Pattern classification and scene analysis.* New York: John Wiley & Sons.

Kittler, J. (1986). Feature selection and extraction. In Young & Fu, (eds.), *Handbook of pattern recognition and image processing*. New York: Academic Press.

Kononenko, I. (1990). Comparison of inductive and naive Bayesian learning approaches to automatic knowledge acquisition. In B. Wielinga (Eds..), *Current trends in knowledge acquisition.* Amsterdam: IOS Press.

Kononenko, I. (1991). Semi-naive Bayesian classifier. *Proceedings of the Sixth European Working Session on Learning.* (pp. 206–219). Porto, Portugal: Pittman.

Langley, P. (1993). Induction of recursive Bayesian classifiers. *Proceedings of the 1993 European Conference on Machine Learning.* (pp. 153–164). Vienna: Springer-Verlag.

Langley, P. & Sage, S. (1994). Induction of selective Bayesian classifiers. *Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence.* Seattle, WA

Murphy, P. M. , & Aha, D. W. (1994). UCI Repository of machine learning databases [Machine-readable data repository]. Irvine: University of California, Department of Information & Computer Science.

Pazzani, M., Merz, C., Murphy, P., Ali, K., Hume, T., and Brunk, C. (1994). Reducing Misclassification Costs. *Proceedings of the Eleventh International Conference on Machine Learning*. New Brunswick, NJ.

Pearl, J. (1988). *Probabilistic reasoning in intelligent systems: Networks of plausible inference.* San Mateo, CA: Morgan Kaufmann.

Schaffer, C. (1994). A conservation law of generalization performance *Proceedings of the Eleventh International Conference on Machine Learning*. New Brunswick, NJ.

Quinlan, J.R. (1986). Induction of decision trees. *Machine Learning, 1,* 81–106.