Efficient Lazy Elimination for Averaged One-Dependence Estimators

naive Bayes, semi-naive Bayes, attribute independence assumption

Abstract

Semi-naive Bayesian classifiers seek to retain the numerous strengths of naive Bayes while reducing error by weakening the attribute independence assumption. Backwards Sequential Elimination (BSE) is a wrapper technique for attribute elimination that has proved effective at this task. We explore a new efficient technique, Lazy Elimination (LE), which eliminates highly related attribute-values at classification time without the computational overheads inherent in wrapper techniques. We analyze the effect of LE and BSE on Averaged One-Dependence Estimators (AODE), a state-of-the-art semi-naive Bayesian algorithm. Our extensive experiments show that LE significantly reduces bias and error without undue additional computation, while BSE significantly reduces bias but not error, with high training time complexity. In the context of AODE, LE has a significant advantage over BSE in both computational efficiency and error.

1. Introduction

Naive Bayes (NB) is a simple, efficient and effective approach to classification learning built on the assumption of conditional independence between the attributes given the class. Although the assumption is unrealistic in many practical scenarios, NB has exhibited competitive accuracy with other learning algorithms. There are many attempts to explain NB's surprising degree of competitiveness, and to develop semi-naive Bayes techniques that further improve its accuracy by alleviating the attribute interdependence problem while at the same time retaining NB's simplicity and efficiency (Kittler, 1986; Kononenko, 1991; Langley, 1993; Langley & Sage, 1994; Kohavi, 1996; Pazzani, 1996; Sahami, 1996; Singh & Provan, 1996; Friedman et al., 1997; Webb & Pazzani, 1998; Keogh & Pazzani, 1999; Zheng et al., 1999; Zheng & Webb, 2000; Webb, 2001; Frank et al., 2003; Webb et al., 2005; Jing et al., 2005; Cerquides & Mántaras, 2005; Zhang et al., 2005b).

Many semi-naive Bayes techniques identify and repair harmful inter-dependencies by a simple heuristic wrapper approach that seeks to minimize error on the training set (Kittler, 1986; Langley & Sage, 1994; Pazzani, 1996; Kohavi, 1996; Keogh & Pazzani, 1999; Zheng & Webb, 2000). Backwards Sequential Elimination (BSE) (Kittler, 1986) achieves this by eliminating attributes using leave-one-out cross validation error on the target learning algorithm as the elimination criterion. This approach has proved to be beneficial in domains with highly correlated attributes. However, BSE has high computational overheads, especially on learning algorithms with high classification time complexity, as it applies the algorithms themselves repeatedly until there is no accuracy improvment. In this paper we present a new type of semi-naive Bayesian operation, a Lazy Elimination (LE) technique that utilizes the tables of probability estimates formed at training time to efficiently detect and address a special form of dependency between two attribute-values at classification time. Such dependencies can degrade NB's accuracy. This technique identifies at classification time attribute-values pairs such that one is a generalization of the other. The technique deletes the generalization, which we show is the theoretically correct adjustment for such an inter-dependence relationship.

Previous research (Zheng & Webb, 2005) indicates that Averaged One-Dependence Estimators (AODE) (Webb et al., 2005) has a significant advantage in error over many other semi-naive Bayesian algorithms, with the exceptions of Lazy Bayesian rules (LBR) (Zheng & Webb, 2000) and SuperParent Tree Augment Naive Bayes (SP-TAN) (Keogh & Pazzani, 1999). It shares similar levels of error with these two algorithms without the prohibitive training time of SP-TAN or test time of LBR. As AODE substantially improves upon the error of NB without incurring undue computa-

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

tional overheads, it can be used as an alternative to NB in many cases and has attracted substantial interest (Frank et al., 2003; Nikora, 2005; Jing et al., 2005; Cerquides & Mántaras, 2005; Zhang et al., 2005a; Zhang et al., 2005b; Su & Zhang, 2005). We investigate the effect of LE and BSE on AODE using bias-variance decomposition, a key tool for understanding machine learning algorithms. LE imposes no extra training time overheads on AODE and at most modest test time overheads, while BSE imposes very high training time overheads accompanied by varying decreases in classification time overheads. Our extensive experimental comparison of performance on 56 UCI data sets shows that the accuracy of AODE can be significantly improved by the addition of LE, but not BSE. In the context of AODE, BSE has a significant advantage in bias over LE, while LE has a significant advantage in variance and error over BSE. Like other wrapper techniques, BSE is not suited to incremental learning. LE delays computation until classification time, and hence does not affect AODE's capacity for incremental learning.

2. AODE

The Bayesian classifier (Duda & Hart, 1973) classifies an example $\mathbf{x} = \langle x_1, \ldots, x_n \rangle$ by selecting

$$\operatorname*{argmax}_y \left(P(y \mid x_1, \ldots, x_n) \right), \tag{1}$$

where x_i is the value of the *ith* attribute, and $y \in c_1, \ldots c_k$ are the k classes. Under the attribute independence assumption, this equals:

$$\operatorname*{argmax}_y \left(P(y) \prod_{i=1}^{n} P(x_i \mid y) \right). \tag{2}$$

Naive Bayes uses this formula for classification. Domingos and Pazzani (1996) point out that interdependences between attributes will not affect NB's accuracy performance, so long as it can generate the correct ranks of conditional probabilities for the classes. However, the success of semi-naive Bayesian methods show that appropriate weakening of the attribute independence assumption is effective (Kittler, 1986; Langley & Sage, 1994; Kohavi, 1996; Pazzani, 1996; Sahami, 1996; Singh & Provan, 1996; Friedman et al., 1997; Webb & Pazzani, 1998; Keogh & Pazzani, 1999; Zheng & Webb, 2000; Webb, 2001; Webb et al., 2005).

One approach to weakening the attribute independence assumption is to use a one-dependence classifier (Sahami, 1996), such as TAN (Friedman et al., 1997), in which each attribute depends upon the class and at most one other attribute. AODE (Webb et al., 2005) selects a limited class of 1-dependence classifiers and aggregates the predictions of all qualified classifiers within this class. A single attribute is selected as the parent of all other attributes in each 1-dependence classifier. There is no model selection, which may minimize the variance component of a classifier's error (Hastie et al., 2001). In order to avoid unreliable base probability estimates, the original AODE excludes models where the frequency of the value for classified object of the parent attribute in the training data is fewer than limit m=30, a widely used minimum on sample size for statistical inference purposes. However, subsequent (unpublished) research shows that this constraint actually increases error and hence the current research uses m=1. As AODE makes a weaker attribute independence assumption, and avoids model selection, it has substantially lower bias than NB with a very small increase in variance.

From the definition of conditional probability we have

$$P(y \mid \mathbf{x}) = P(y, \mathbf{x}) / P(\mathbf{x}) \propto P(y, \mathbf{x}), \tag{3}$$

and for any attribute value x_i,

$$P(y, \mathbf{x}) = P(y, x_i) P(\mathbf{x} \mid y, x_i). \tag{4}$$

This equality holds for every x_i. Therefore, for any $I \subseteq \{1, \ldots, n\}$,

$$P(y, \mathbf{x}) = \frac{\sum_{i \in I} P(y, x_i) P(\mathbf{x} \mid y, x_i)}{|I|}. \tag{5}$$

Thus,

$$P(y, \mathbf{x}) = \frac{\sum_{i:1 \le i \le n \wedge F(x_i) \ge m} P(y, x_i) P(\mathbf{x} \mid y, x_i)}{|\{i : 1 \le i \le n \wedge F(x_i) \ge m\}|}, \tag{6}$$

where $F(x_i)$ is the frequency of attribute-value x_i in the training sample.

To this end, AODE classifies by selecting:

$$\operatorname*{argmax}_y \left(\sum_{i:1 \le i \le n \wedge F(x_i) \ge m} P(y, x_i) \prod_{j=1}^{n} P(x_j \mid y, x_i) \right). \tag{7}$$

At training time AODE generates a three-dimensional table of probability estimates for each attribute-value, conditioned by each other attribute-value and each class. The resulting space complexity is $O(k(nv)^2)$, where v is the mean number of values per attribute.

The time complexity of forming this table is $O(tn^2)$, where t is the number of training examples, as an entry must be updated for every training case and every combination of two attribute-values for that case. Classification requires the tables of probability estimates formed at training time of space complexity $O(k(nv)^2)$. The time complexity of classifying a single example is $O(kn^2)$ as we need to consider each pair of qualified parent and child attribute within each class.

3. Related attribute-values and Lazy Elimination (LE)

In many real world problems the attribute independence assumption is violated. Correlations among attributes are common. When two attributes are related, NB may place too much weight on the influence from the two attributes, and too little on the other attributes, which can result in classification bias. Deleting one of these attributes may have the effect of alleviating the problem. Note that the dependence between two attributes comes from the relationship of their values.

One extreme type of interdependence is the specialization–generalization relationship. For two attribute values x_i and x_j, if $P(x_j \mid x_i) = 1.0$ then x_j is a generalization of x_i and x_i a specialization of x_j.

Theorem. *If x_j is a generalization of $x_i, 1 \leq i \leq n, 1 \leq j \leq n, i \neq j$ then $P(y \mid x_1, \ldots, x_n) = P(y \mid x_1, \ldots, x_{j-1}, x_{j+1}, \ldots x_n)$.*

Proof. Note, $\forall Z$, if $P(x_j \mid x_i) = 1.0$, then $P(x_i, x_j, Z) = P(x_i, Z)$. Hence,

$$
\begin{aligned}
& P(y \mid x_1, \ldots, x_n) \\
&= \frac{P(y, x_1, \ldots, x_n)}{P(x_1, \ldots, x_n)} && (8) \\
&= \frac{P(y, x_1, \ldots, x_{j-1}, x_{j+1}, \ldots x_n)}{P(x_1, \ldots, x_{j-1}, x_{j+1}, \ldots x_n)} && (9) \\
&= P(y \mid x_1, \ldots, x_{j-1}, x_{j+1}, \ldots x_n) && (10)
\end{aligned}
$$

□

Hence, deleting the generalization x_j from a Bayesian classifier should not be harmful, and if that classifier makes unwarranted assumptions about the relationship of x_j to the other attributes, such as NB's independence assumption, it may be positive.

To illustrate this, consider the data presented in Table 1 for hypothetical example with the three attributes *Gender*, *Pregnant* and *MaleHormone* and the class *Normal*. *Pregnant=yes* is a specialization of *Gender=female* and *Gender=male* is a specialization of *Pregnant=no*. These two attributes are highly related. Given a test instance $\langle Gender{=}male, Pregnant{=}no, MaleHormone{=}3\rangle$, which occurred in the training data, NB misclassifies it as *Normal=no*. The reason is that NB in effect double counts the evidence from *Pregnant=no*. The new object can be correctly classified as *Normal=yes* by deleting attribute-value *Pregnant=no*.

Table 1. An example

Gender	*Pregnant*	*MaleHormone*	*Normal*
male	no	3	yes
female	yes	3	yes
female	yes	2	yes
female	yes	2	yes
male	no	1	no
female	no	3	no
female	no	4	no
female	yes	4	no

However, if *Pregnant=no*, we cannot make any definite conclusion of the value of *Gender*, nor about the value of *Pregnant* if $Gender = female$. Deleting one of the attribute-values *Pregnant=no* and $Gender{=}female$ will lose information. Hence, we use both if neither attribute-value is a generalization of the other. Such dependence is determined by the values of two attributes.

Note that the generalization relation is transitive. If $P(x_j \mid x_i) = 1.0$, and $P(x_h \mid x_j) = 1.0$, then $P(x_h \mid x_i) = 1.0$. Hence, we can delete two attribute-values x_j and x_h. Another two cases are illustrated in Figure 1. The parent nodes are specializations of the child nodes.

Figure 1. (a) *Multiple children*, (b) *Multiple parents*

In the former case, we delete two attribute-values x_j and x_h. In the latter case, we delete attribute value x_j. Eliminating attribute-values in this way, we only use the nodes without parents in the graph to classify an object.

However, the transitive property does not hold for a near generalization relation. For instance, if $P(x_j \mid x_i) = 0.9$ and $P(x_h \mid x_j) = 0.9$, we can not infer that $P(x_h \mid x_i) = 0.9$. In an extreme case, $P(x_h \mid x_i)$ may equal zero. Hence, we only consider the perfect

generalization relation, which is a common relation in real world problems.

It is superficially attractive to pre-check the generalization relation at training time. The complexity of creating the dependency matrix is $O(n^2v^2)$, as it requires each pair of attributes, every pairwise combination of their respective values to be considered. At classification time, scanning the dependency matrix to delete attributes has time complexity of $O(n^2)$. However, if we check attribute-value pairs for generalization relationships at classification time (so there is no additional computation for dependency matrix at training time), time complexity is $O(n^2)$ as well. Hence, we infer the generalization relation at classification time by utilizing the tables of probability estimates formed at training time. We call the technique Lazy Elimination (LE), as it delays the computation of elimination until classification time, and deletes different attributes depending upon which attribute values are instantiated in the object being classified.

4. LE for AODE

AODE has very competitive prediction accuracy, low variance, and high computational efficiency. We explore the effect of LE on AODE, which deletes generalization attribute-values if a specialization is detected, and aggregates the predictions of all qualified classifiers using resulting attribute-values. For brevity and clarity, we call the resulting classifier LE, and AODE without LE as NE (No Elimination).

Classification consists of two steps:

1. Check for dependence between each pair of attribute values. If $P(x_j \mid x_i) = 1.0$, delete x_j. The resulting attribute value set is denoted as $Atts = \{x_{i_1}, \ldots, x_{i_p}\}$.

2. Classify the instance by selecting:

$$\operatorname*{argmax}_y \left(\sum_{i:i_1 \le i \le i_p \wedge F(x_i) \ge m} P(y, x_i) \prod_{j=i_1}^{i_p} P(x_j \mid y, x_i) \right). \tag{11}$$

LE requires a criterion for inferring from sample data when $P(x_j \mid x_i) = 1.0$. Clearly it would be dangerous to infer that this condition holds if the data contains only a small number of examples of x_j. We use the criterion that x_i for all cases for which x_j, and that x_i for at least 30 cases, 30 being a widely used minimum on sample size for statistical inference purposes. We use m=1 as the frequency limit to accept a conditional probability estimate of 1-dependence classifiers as AODE does.

LE has identical time and space complexity to AODE. At training time it behaves identically to AODE, simply computing the required three dimensional joint frequency table mentioned in Section 2. At classification time, it must check all attribute-value pairs for generalization relationships, an additional operation of time complexity $O(n^2)$. However, the time complexity of AODE at classification time is $O(kn^2)$ and so this additional computation does not increase the time complexity. LE inherits AODE's capacity for incremental learning, as updating the classifier with evidence from a new example requires only incrementing the relevant entries in the tables of probability estimates.

5. BSE for AODE

Backwards Sequential Elimination (BSE) (Kittler, 1986) selects a subset of attributes using leave-one-out cross validation error as a selection criterion. Starting from the full set of attributes, BSE successively eliminates the attribute whose elimination most improves accuracy, until there is no further accuracy improvement. In the context of AODE, BSE uses leave-one-out cross validation error on AODE as deleting criterion, and averages the predictions of all qualified classifiers using resulting attribute set. The subset of selected attributes is denoted as $Atts = \{X_{i_1}, \ldots, X_{i_q}\}$. BSE Classifies the instance by selecting

$$\operatorname*{argmax}_y \left(\sum_{i:i_1 \le i \le i_q \wedge F(x_i) \ge m} P(y, x_i) \prod_{j=i_1}^{i_q} P(x_j \mid y, x_i) \right). \tag{12}$$

The same frequency limit m=1 is used as for AODE. At training time BSE generates a three-dimensional table of probability estimates, as AODE does. It must also store the training data, with additional space complexity $O(tn)$, to perform leave-one-out cross validation on AODE. The resulting space complexity is $O(tn + k(nv)^2)$. Deleting attributes has time complexity of $O(tkn^4)$, as a single leave-one-out cross validation is order $O(tkn^2)$ and it is performed at most $O(n^2)$ times. BSE has identical time and space complexity with AODE at classification time. However, it does not support incremental learning, as it has to update the classifier by re-performing leave-one-out cross validation using all examples available in the past.

6. Bias and variance

Bias-variance decomposition provides valuable insights into the components of the error of classifiers learned by learning algorithms. Bias denotes the systematic component of error, which describes how closely the

Table 2. Data sets

No.	Domain	Cases	Atts	Class
1	Abalone	4,177	9	3
2	Adult	48,842	15	2
3	Annealing	898	39	6
4	Audiology	226	70	24
5	Autos Imports-85	205	26	7
6	Balance Scale	625	5	3
7	Breast Cancer (Wisconsin)	699	10	2
8	Chess	551	40	2
9	Connect-4 Opening	67,557	43	3
10	Credit Approval	690	16	2
11	Diabetes	768	9	2
12	Echocardiogram	131	7	2
13	German	1,000	21	2
14	Glass Identification	214	10	3
15	Heart	270	14	2
16	Heart Disease (cleveland)	303	14	2
17	Hepatitis	155	20	2
18	Horse Colic	368	23	2
19	House Votes 84	435	17	2
20	Hungarian	294	14	2
21	Hypothyroid	3,163	26	2
22	Hypothyroid(Garavan Institute)	3,772	30	4
23	Ionosphere	351	35	2
24	Iris Claasification	150	5	3
25	King-rook-vs-king-pawn	3,196	37	2
26	Labor negotiations	57	17	2
27	LED	1,000	8	10
28	Letter Recognition	20,000	17	26
29	Liver Disorders (bupa)	345	7	2
30	Lung Cancer	32	57	3
31	Lymphography	148	19	4
32	mfeat-mor	2,000	7	10
33	Mushrooms	8,124	23	2
34	Nettalk(Phoneme)	5,438	8	46
35	New-Thyroid	215	6	3
36	Optical Digits	5,620	50	10
37	Page Blocks	5,473	11	5
38	Pen Digits	10,992	17	10
39	Pima Indians Diabetes	768	9	2
40	Postoperative Patient	90	9	3
41	Primary Tumor	339	18	22
42	Promoter Gene Sequences	106	58	2
43	Satellite	6,435	37	6
44	Segment	2,310	20	7
45	Sign	12,546	9	3
46	Solar Flare	1,389	10	2
47	Sonar Classification	208	61	2
48	Splice-junction Gene Sequences	3,190	62	3
49	Syncon	600	61	6
50	Thyroid Disease(Garavan Institute)	3,772	30	2
51	Tic-Tac-Toe Endgame	958	10	2
52	Vehicle	846	19	4
53	Waveform-5000	5,000	41	3
54	Wine Recognition	178	14	3
55	Vowel	990	14	11
56	Zoo	101	18	7

learner is able to describe the decision surfaces for a domain. Variance describes the component of error that stems from sampling, which reflects the sensitivity of the learner to variations in the training sample (Kohavi & Wolpert, 1996). There is a bias-variance tradeoff such that bias typically increases when variance decreases and vice versa. In general, the better the learner is able to fit the training data, the lower the bias. However, closely fitting the training data may result in greater changes in the model formed from sample to sample, and hence higher variance.

There are a number of different bias-variance decomposition methods. In the current research, we use the repeated cross-validation bias-variance estimation method proposed by Webb (2000). This is preferred to the default method in Weka as it results in the use of substantially larger training sets. In order to maximize the variation in the training data from trial to trial we use two-fold cross validation. The training data are randomly divided into two folds. Each fold is used as a test set for a classifier generated from the other fold. Hence, each available example is classified once for each two-fold cross-validation. Bias and variance are estimated by fifty runs of two-fold cross-validation in order to give a more accurate estimation of the average performance of an algorithm. The advantage of this technique is that it uses the full training data as the training set and test set, and every case in the training data is used the same number of times.

7. Experimental results

The fifty-six natural domains from the UCI Repository of machine learning used in our experiments are shown in Table 2. The experiments were performed in the Weka workbench (Witten & Frank, 2000) on a dual-processor 1.7 GHz Pentium 4 Linux computer with 2 Gb RAM, and all data were discretized using MDL discretization (Fayyad & Irani, 1993). The base probabilities were estimated using Laplace estimation. We compare the performance of these three algorithms using the method mentioned in Section 6.

Table 3. Mean for NE, LE and BSE

	NE	**LE**	**BSE**
Mean error	0.1882	0.1849	0.1875
Mean bias	0.1358	0.1315	0.1310
Mean variance	0.0525	0.0534	0.0565

Table 3 shows the mean error, bias and variance across all the data sets for each of NE, LE and BSE. Figure 2 graphs the relative error, bias and variance of

Table 4a. Win/Draw/Loss: LE vs. alternative

	NE		**BSE**	
	W/D/L	p	W/D/L	p
Error	29/13/14	0.0158	35/2/19	0.0201
Bias	40/13/3	<0.0001	17/2/37	0.0045
Variance	10/14/32	0.0005	44/2/10	<0.0001

Table 4b. Win/Draw/Loss: BSE vs. alternative

	NE		**LE**	
	W/D/L	p	W/D/L	p
Error	21/3/32	0.0845	19/2/35	0.0201
Bias	47/2/7	<0.0001	37/2/17	0.0045
Variance	7/3/46	<0.0001	10/2/44	<0.0001

the three classifiers. The values on the y-axis are the outcome for BSE divided by that for NE. The values of the x-axis are the outcome for LE divided by that for NE. Each point on the graph represents one of the 56 data sets. Points on the left of the vertical line at LE/NE=1 in each subgraph are those of which LE has better results than NE. Points below the horizontal line at BSE/LE=1 indicate that BSE wins in those domains compared with NE. Points above the line X=Y represent that LE has lower values than those of BSE.

Table 4a presents the win/draw/loss records for LE against the alternative algorithms on fifty-six data sets. The win/draw/loss records for BSE against the alternative algorithms is shown in Table 4b. The p value is the outcome of a one-tailed binomial sign test. We assess a difference as significant if $p \leq 0.05$.

Considering first the error outcomes, LE achieves the lowest mean error. The win/draw/loss record indicates that LE has a significant advantage over NE and BSE. However, there is no significant error difference between NE and BSE. From the error graph in Figure 2, we can see that the majority of the points are on the left of the vertical line at LE/NE=1, and above the line X=Y. The error ratios of LE and BSE over NE on King-rook-vs-king-pawn (the point at the bottom of the graph) are 0.8478 and 0.6237 respectively. The error ratio of BSE over LE is 0.7357. That is, both LE and BSE reduce the error of NE considerably, while BSE has substantial lower error than LE. BSE might identify wider range of dependencies than LE, which can only detect a special dependency relationship. Hence, the error of BSE might be greatly affected in King-rook-vs-king-pawn, in which there are strong dependencies between the presence and posi-

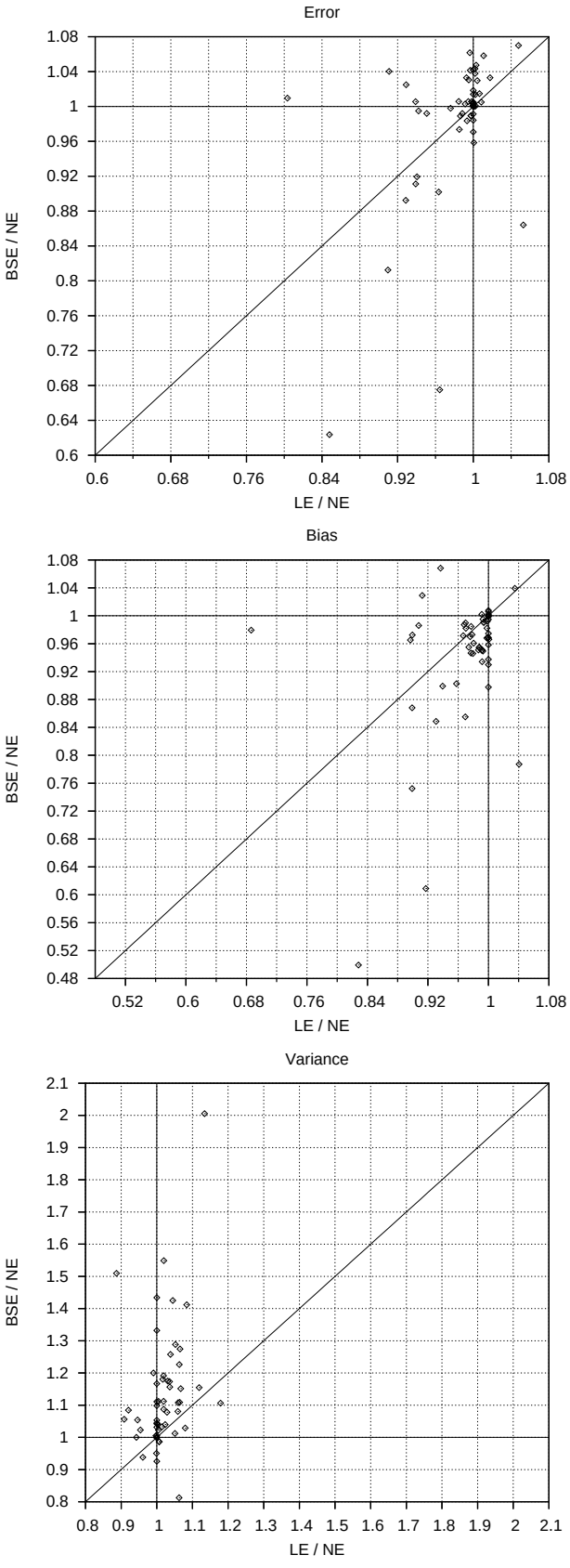

Figure 2. Comparison of Error, Bias and Variance

tion of pieces on the board. However, it appears that BSE does not scale well to the data sets with many attributes, such as Audiology. The error ratio of LE and BSE over NE on Audiology are 0.8034 and 1.0096 respectively, and the error ratio of LE over BSE is 0.7957. For the 10 data sets with more than 38 attributes, LE has better error rates than BSE on 9 data sets, except for Splice-junction Gene Sequences, on which LE has no effect for NE, and BSE has a small reduction on error.

With respect to bias and variance, BSE exhibits the lowest mean bias and highest mean variance. The win/draw/loss records for bias show that the advantage of LE and BSE is significant compared to NE. BSE has a significant advantage over LE in bias. Most of points are in the area indicates by the vertical line at LE/NE=1 and horizontal line at BSE/NE=1 in the bias graph. The advantage of NE and LE in variance is significant compared with BSE. The variance graph shows that NE wins in most cases. The variance ratio of LE and BSE over NE on Hungarian (the point located at the top of the variance graph) are 1.1341 and 2.0056 respectively. On this data set BSE has the lowest bias but highest variance and error, while LE has the lowest error.

BSE has lower bias, higher variance and higher error than NE in the 26 data sets, while LE has lower bias, higher variance and higher error than NE in 10 data sets. Among these 10 data sets, 9 data sets are in the 26 data sets for BSE, with the exception of Echocardiogram, on which BSE has higher bias, variance and error than NE. BSE has lower bias, higher variance and higher error than LE in 19 data sets. These results suggest that LE performs less aggressive attribute elimination than BSE.

8. Conclusion

Of many semi-naive Bayes approaches to deal with the inter-dependencies problem, most use a wrapper to identify irrelevant and redundant attributes, such as BSE. We have proposed a novel technique, LE, to efficiently eliminate highly correlated attribute values at classification time. We investigate the effect of LE and BSE on AODE. Extensive experimental results show that both of them have substantially lower bias, but higher variance than AODE. However, LE reduces the error of AODE considerably without computational burden, while BSE has no consistent error reduction on AODE with very high training time complexity. The advantage of LE in error and computation efficiency over BSE is significant in the context of AODE. Note, however, that the failure of BSE to consistently improve upon AODE runs counter to the experience of BSE applied to NB (Langley & Sage, 1994) and of appropriate selection of parent attributes for AODE (Yang et al., 2005), and it is plausible that appropriate refinement of the technique might substantially improve its performance. We believe that the appropriate conclusion to draw from our results is that LE is effective, rather than that it is necessarily superior to the BSE strategy in the AODE context.

As LE eliminates highly dependent attribute values in a lazy manner, it does not interfere with AODE's capacity for incremental learning. However, it is only applicable to algorithms without model selection, such as NB[1] and AODE. BSE is more widely applicable, but does not support incremental learning.

References

Cerquides, J., & Mántaras, R. L. D. (2005). Robust bayesian linear classifier ensembles. *Proc. 16th European Conf. Machine Learning, Lecture Notes in Computer Science* (pp. 70–81).

Domingos, P., & Pazzani, M. J. (1996). Beyond independence: Conditions for the optimality of the simple Bayesian classifier. *Proc. 13th Int. Conf. Machine Learning* (pp. 105–112). Morgan Kaufmann.

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

Fayyad, U. M., & Irani, K. B. (1993). Multi-interval discretization of continuous-valued attributes for classification learning. *Proc. 13th Int. Joint Conf. Artificial Intelligence (IJCAI-93)* (pp. 1022–1029). Morgan Kaufmann.

Frank, E., Hall, M., & Pfahringer, B. (2003). Locally weighted naive Bayes. *Proc. 19th Conference in Uncertainty in Artificial Intelligence (UAI 2003)* (pp. 249–256). Morgan Kaufmann.

Friedman, N., Geiger, D., & Goldszmidt, M. (1997). Bayesian network classifiers. *Machine Learning*, *29*, 131–163.

Hastie, T., Tibshirani, R., & Friedman, J. (2001). *Elements of statistical learning: Data mining, inference and prediction*. New York: Springer.

Jing, Y., Pavlović, V., & Rehg, J. M. (2005). Efficient discriminative learning of bayesian network classifier via boosted augmented naive bayes. *Proc. 22nd*

[1] In experiments, not presented, LE proves effective at reducing the error of NB, but not as effective as AODE with which LE+NB shares similar computational complexity.

Int. Conf. Machine learning (ICML 2005) (pp. 369–376). ACM Press.

Keogh, E. J., & Pazzani, M. J. (1999). Learning augmented Bayesian classifers: A comparison of distribution-based and classification-based approaches. *Proc. Int. Workshop on Artificial Intelligence and Statistics* (pp. 225–230).

Kittler, J. (1986). Feature selection and extraction. In T. Y. Young and K.-S. Fu (Eds.), *Handbook of pattern recognition and image processing*. New York: Academic Press.

Kohavi, R. (1996). Scaling up the accuracy of naive-Bayes classifiers: a decision-tree hybrid. *Proc. 2nd ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining* (pp. 202–207).

Kohavi, R., & Wolpert, D. (1996). Bias plus variance decomposition for zero-one loss functions. *Proc. 13th Int. Conf. Machine Learning* (pp. 275–283). San Francisco: Morgan Kaufmann.

Kononenko, I. (1991). Semi-naive Bayesian classifier. *Proc. 6th European Working Session on Machine learning* (pp. 206–219). Berlin: Springer-Verlag.

Langley, P. (1993). Induction of recursive Bayesian classifiers. *Proc. 1993 European Conf. Machine Learning* (pp. 153–164). Berlin: Springer-Verlag.

Langley, P., & Sage, S. (1994). Induction of selective Bayesian classifiers. *Proc. 10th Conf. Uncertainty in Artificial Intelligence* (pp. 399–406). Morgan Kaufmann.

Nikora, A. P. (2005). Classifying requirements: Towards a more rigorous analysis of natural-language specifications. *Proc. 16th IEEE Int. Symposium on Software Reliability Engineering (ISSRE'05)* (pp. 291–300).

Pazzani, M. J. (1996). Constructive induction of Cartesian product attributes. *ISIS: Information, Statistics and Induction in Science*, 66–77.

Sahami, M. (1996). Learning limited dependence Bayesian classifiers. *Proc. 2nd Int. Conf. Knowledge Discovery in Databases* (pp. 334–338). Menlo Park, CA: AAAI Press.

Singh, M., & Provan, G. M. (1996). Efficient learning of selective Bayesian network classifiers. *Proc. 13th Int. Conf. Machine Learning* (pp. 453–461). Morgan Kaufmann.

Su, J., & Zhang, H. (2005). Representing conditional independence using decision trees. *Proc. 20th National Conf. Artificial Intelligence (AAAI 2005)* (pp. 874–879). AAAI Press.

Webb, G. I. (2000). Multiboosting: A technique for combining boosting and wagging. *Machine Learning*, *40*, 159–196.

Webb, G. I. (2001). Candidate elimination criteria for lazy Bayesian rules. *Proc. 14th Australian Joint Conf. Artificial Intelligence* (pp. 545–556). Berlin:Springer.

Webb, G. I., Boughton, J., & Wang, Z. (2005). Not so naive Bayes: Aggregating one-dependence estimators. *Machine Learning*, *58*, 5–24.

Webb, G. I., & Pazzani, M. J. (1998). Adjusted probability naive Bayesian induction. *Proc. 11th Australian Joint Conf. Artificial Intelligence* (pp. 285–295). Berlin:Springer.

Witten, I. H., & Frank, E. (2000). *Data mining : practical machine learning tools and techniques with java implementations*. San Francisco, CA: Morgan Kaufmann.

Yang, Y., Korb, K., Ting, K.-M., & Webb, G. I. (2005). Ensemble selection for superparent-one-dependence estimators. *Proc. 18th Australian Joint Conf. Artificial Intelligence* (pp. 102–111). Springer.

Zhang, H., Jiang, L., & Su, J. (2005a). Augmenting naive bayes for ranking. *Proc. 22nd Int. Conf. Machine Learning (ICML 2005)* (pp. 1025–1032). ACM Press.

Zhang, H., Jiang, L., & Su, J. (2005b). Hidden naive Bayes. *Proc. 20th National Conf. Artificial Intelligence (AAAI 2005)* (pp. 919–924). AAAI Press.

Zheng, F., & Webb, G. I. (2005). A comparative study of semi-naive Bayes methods in classification learning. *Proc. 4th Australasian Data Mining conference (AusDM05)* (pp. 141–156).

Zheng, Z., & Webb, G. I. (2000). Lazy learning of Bayesian rules. *Machine Learning*, *41*, 53–84.

Zheng, Z., Webb, G. I., & Ting, K. M. (1999). Lazy Bayesian rules: A lazy semi-naive Bayesian learning technique competitive to boosting decision trees. *Proc. 16th Int. Conf. Machine Learning (ICML 1999)* (pp. 493–502). Morgan Kaufmann.