Averaged One-Dependence Estimators: Preliminary Results

Geoffrey I. Webb
School of Computer Science and Software Engineering
Monash University
Vic. 3800, Australia

Janice Boughton
School of Computer Science and Software Engineering
Monash University
Vic. 3800, Australia

Zhihai Wang
School of Computer Science and Software Engineering
Monash University
Vic. 3800, Australia

webb@infotech.monash.edu.au

ABSTRACT

Naive Bayes is a simple, computationally efficient and remarkably accurate approach to classification learning. These properties have led to its wide deployment in many online applications. However, it is based on an assumption that all attributes are conditionally independent given the class. This assumption leads to decreased accuracy in some applications. AODE overcomes the attribute independence assumption of naive Bayes by averaging over all models in which all attributes depend upon the class and a single other attribute. The resulting classification learning algorithm for nominal data is computationally efficient and achieves very low error rates.

1. INTRODUCTION

Naive Bayes has gained widespread application due to its simplicity, computational efficiency, direct theoretical basis, and competitive classification accuracy. It is particularly attractive for interactive applications due to the speed with which it can be applied. This speed is derived from its use of a simplifying attribute-independence assumption.

Naive Bayes has a long history of application in information retrieval [12] and has gained some popularity in the machine learning community. It has numerous desirable features. It is extremely efficient. It is provably optimal bar only for two explicit assumptions, the attribute independence assumption and the assumption that the estimates based on frequency are accurate. Finally, for many classification tasks its accuracy is extremely competitive with much more computationally intensive techniques [8; 11], especially when sample sizes are small [21].

However, where the attribute independence assumption is violated, its accuracy may suffer. It should be noted that not all violations of the attribute independence assumption matter. Naive Bayes will remain an optimal classifier so long as its probability estimate for the most probable class is higher than its probability estimate for any other class [3]. Nonetheless, harmful violations of the attribute independence assumption appear to be frequent in real-world application domains. In consequence, there is a sizeable body of literature addressing measures that can be applied to maintain the desirable features of naive Bayes while reducing the harmful effects of the attribute independence assumption [4; 6; 7; 9; 10; 11; 13; 14; 15; 18; 17; 21; 22].

Of these approaches, two in particular have demonstrated remarkable prediction accuracy, LBR [21] and TAN [4]. However, both these algorithms have high computational requirements reducing their utility in many data mining applications. We present a new algorithm that results from our efforts to attain the prediction accuracy of LBR and TAN without the computational overheads.

2. CLASSIFICATION BY CONDITIONAL PROBABILITY ESTIMATION

We assume a joint probability distribution X, Y, a sample X^*, Y^* drawn from X, Y, where each $x \in X$ is an object described by a vector of k attribute values $\langle x_1, x_2, \ldots x_k \rangle$ and each $y \in Y$ is a class label. The classification task is to assign a $y \in Y$ to a given $x \in X$.

Assuming that x is drawn at random from X, Y, error can be minimized by selecting $argmax_y P(y \mid x)$. As $P(y \mid x)$ is not generally known, one approach is to estimate it from X^*, Y^*. In general, $P(Z \mid W) = P(W \wedge Z)/P(W)$. If X^*, Y^* is a representative sample of X, Y then for any predicates W and Z, $P(W \wedge Z) \approx F(W \wedge Z, X^*, Y^*)$ and $P(W) \approx F(W, X^*, Y^*)$, where $F(Z, X, Y)$ denotes the frequency with which Z occurs in the reference distribution X, Y. However, to guard against problems associated with sampling error, when estimating population frequency from the sample frequency the exact sample frequency is usually adjusted providing a function on the sample frequency, $f(Z, X^*, Y^*)$, such as the Laplace error estimate [5] or an m-estimate [1]. In consequence,

$$\begin{aligned} P(y \mid x) &= P(x \wedge y)/P(x) &(1)\\ &\approx f(x \wedge y, X^*, Y^*)/f(x, X^*, Y^*). &(2)\end{aligned}$$

However, the accuracy of this approximation will deteriorate as the values of $P(x \wedge y)$ and $P(x)$ decrease. Where k is large, the probabilities $P(x)$ and $P(x \wedge y)$ are likely to be extremely low, and hence the approximation in (2) will be poor. This problem can be circumvented with respect to estimating $P(x)$ by observing that $P(x) = \sum_{y \in Y} P(x \wedge y)$ and hence that it is necessary only to estimate the numerator for each class and the denominator in (1) can be calculated therefrom. Alternatively, if we wish only to identify the most probable class, we can observe that $P(x)$ is invariant across classes and hence for any x, $P(y \,|\, x) \propto P(y)P(x \,|\, y)$ thus relieving us from the need to estimate $P(x)$. Nonetheless, it remains necessary to estimate $P(x \wedge y)$. In this context it is necessary to use less direct approximations for $P(y \,|\, x)$.

One approach is to use Bayes theorem:

$$P(Z \,|\, W) = \frac{P(Z)P(W \,|\, Z)}{P(W)}. \tag{3}$$

Bayes theorem holds irrespective of whether one adopts a Bayesian or a frequentist definition of probability. Hence its use need not imply a commitment to one or the other theoretical approach to probability. When estimating probabilities from data it can be used to replace the term $P(Z \,|\, W)$ with $\frac{P(Z)P(W \,|\, Z)}{P(W)}$ in contexts where the base terms in the latter can be estimated more reliably or accurately than direct estimation of the former.

However, using Bayes theorem to replace the estimation of $P(y \,|\, x)$ by the estimation of $P(y)P(x \,|\, y)/P(x)$ does not directly resolve the problem. If direct estimation of $P(y \,|\, x)$ is inaccurate then direct estimation of $P(x \,|\, y)$ will also be inaccurate because both require estimation of $P(x \wedge y)$.

Naive Bayes resolves this problem by assuming that the attributes are conditionally independent given the class. In consequence, the following equality is assumed:

$$P(x \,|\, y) = \prod_{i=1}^{k} P(x_i \,|\, y). \tag{4}$$

This allows the following approximation to be applied to estimate the conditional probabilities:

$$P(y \,|\, x) \approx \frac{f(y, X^*, Y^*) \prod_{i=1}^{k} \frac{f(x_i \wedge y, X^*, Y^*)}{f(y, X^*, Y^*)}}{\sum_{y' \in Y} \left(f(y', X^*, Y^*) \prod_{i=1}^{k} \frac{f(x_i \wedge y', X^*, Y^*)}{f(y', X^*, Y^*)} \right)} \tag{5}$$

where the denominator may be omitted when the objective is only to identify the most probable class, as it is invariant across classes.

3. LBR AND TAN

LBR and TAN reduce harmful effects of the attribute independence assumption by allowing models to be formed that represent limited attribute interdependencies. For each x to be classified, LBR selects a subset w of the values in x and assumes only that the remaining attributes are independent given $w \wedge y$. Thus, each x is classified using

$$P(y \,|\, x) \approx \frac{\frac{f(y \wedge w, X^*, Y^*)}{f(w, X^*, Y^*)} \prod_{i=1}^{k} \frac{f(x_i \wedge y \wedge w, X^*, Y^*)}{f(y \wedge w, X^*, Y^*)}}{\sum_{y' \in Y} \left(\frac{f(y' \wedge w, X^*, Y^*)}{f(w, X^*, Y^*)} \prod_{i=1}^{k} \frac{f(x_i \wedge y' \wedge w, X^*, Y^*)}{f(y' \wedge w, X^*, Y^*)} \right)} \tag{6}$$

This process is lazy, using a wrapper approach at classification time to select the $w \subseteq x$ that appears to best reduce error on the training data.

The models that LBR forms have all attributes dependent upon the same group of attributes. In contrast, TAN forms models in which each attribute depends on at most one other attribute. For attribute-value v we will denote the attribute upon which its attribute depends by $parent(v)$. TAN classifies each x using

$$P(y \,|\, x) \approx \frac{f(y, X^*, Y^*) \prod_{i=1}^{k} \frac{f(x_i \wedge x_{parent(x_i)} \wedge y, X^*, Y^*)}{f(x_{parent(x_i)} \wedge y, X^*, Y^*)}}{\sum_{y' \in Y} \left(f(y', X^*, Y^*) \prod_{i=1}^{k} \frac{f(x_i \wedge x_{parent(x_i)} \wedge y', X^*, Y^*)}{x_{parent(x_i)} \wedge f(y', X^*, Y^*)} \right)} \tag{7}$$

TAN selects the parent of each attribute at training time. A variant of TAN with high accuracy uses a wrapper to select the parents that appear to best reduce error [6]. We utilize this variant of TAN in the following work.

LBR and TAN have demonstrated comparable classification accuracy [16]. In separate evaluation, LBR has demonstrated classification accuracy comparable to that of boosting decision trees [22]. However, their very strong classification accuracy comes at considerable computational cost. While the lazy strategy adopted by LBR has negligible training costs, the cost to classify each object at classification time is relatively high. In contrast, TAN is relatively efficient at classification time, but model selection imposes substantial computational overheads at training time.

4. EFFICIENT CLASSIFICATION

We seek to retain the accuracy of LBR and TAN while reducing the computational overheads. These overheads result from two types of activity. The first is model selection, performed at training time by TAN and at classification time by LBR. The second is estimation of the required conditional probabilities.

One way to tackle the latter problem is to restrict the allowed interdependencies to each attribute depending upon the class and one other attribute, in other words to the use of probabilities $P(x_i|x_j \wedge y)$. That is, we restrict the space of models that we will consider to 1-dependence Bayesian classifiers [14], the space of TAN models. 1-dependence conditional probabilities can be estimated efficiently by using joint frequencies stored in a three dimensional table, indexed in one dimension by x_i, in another by x_j and in the third dimension by y. This table can be formed at training time in a single scan through the data. Utilizing greater interdependencies requires higher dimensional tables, potentially resulting in infeasible memory requirements.

By the product rule, for any $x_i \in x$,

$$P(x \wedge y) = P(y \wedge x_i)P(x \,|\, y \wedge x_i). \tag{8}$$

Hence,

$$P(y \,|\, x) = \frac{P(y \wedge x_i)P(x \,|\, y \wedge x_i)}{P(x)}. \tag{9}$$

Utilizing (9) instead of (1) allows us to make a weaker, and hence potentially less harmful, attribute independence assumption than (4),

$$P(x \,|\, y \wedge x_i) = \prod_{j=1}^{k} P(x_j \,|\, y \wedge x_i). \tag{10}$$

This allows the approximation,

$$P(y\,|\,x) \approx \frac{f(x_i \wedge y, X^*, Y^*) \prod_{j=1}^{k} \frac{f(x_j \wedge x_i \wedge y, X^*, Y^*)}{f(x_i \wedge y, X^*, Y^*)}}{\sum_{y' \in Y} \left(f(x_i \wedge y', X^*, Y^*) \prod_{j=1}^{k} \frac{f(x_j \wedge x_i \wedge y', X^*, Y^*)}{f(x_i \wedge y', X^*, Y^*)} \right)}. \qquad (11)$$

(10) is potentially less harmful than (4) because it assumes independence between fewer attributes (x_i is not considered independent of the other attributes) and independence is assumed under strong constraints (the remaining attributes are independent given y and x_i). The estimate (11) can only be less accurate than the estimate (5) if the estimates of the individual probabilities are less accurate. The only systematic effect that might cause this to occur is that the frequencies will be lower and hence derived from fewer examples. We can reduce the impact of this effect by restricting the application of (11) to attribute values x_i which occur with sufficient frequency for us to have confidence that estimation accuracy will not be significantly affected. Note that this restriction implies the use of lazy learning, as the value of an attribute is only determined with respect to a given object that we wish to classify.

Use of (5) reduces the problem of model selection to selecting between up to k models, the models in which every attribute depends upon the class and the same single attribute such that the value of that attribute x_i occurs with sufficient frequency for us to have confidence in the accuracy of its estimation. This leaves us with the problem of how to select the appropriate model. We note that there are two problems associated with model selection. The first is that it can be expected to lead to variance. The second is that it must entail computational overheads, the performance of the necessary computations to determine which is the preferred model. This reasoning led us to seek models that allowed for attribute interdependencies without a process of selecting which interdependencies to represent. We achieve this by averaging across all models of the form (11) such that x_i occurs at least 100 times in the training data. Note that averaging across all such models is justified as (9) holds for any value of i. Hence,

$$P(y\,|\,x) = \frac{\sum_{i=1}^{k} \frac{P(y \wedge x_i) P(x\,|\,y \wedge x_i)}{P(x)}}{k}. \qquad (12)$$

When we replace each $P(y \wedge x_i)P(x\,|\,y \wedge x_i)$, averaging multiple estimates of this quantity with different values of i offers the opportunity to average out estimation errors. If the estimation errors are normally distributed with mean 0, the greater the number of values of i included, the lower the expected estimation error.

The resulting system classifies using the following.

$$P(y\,|\,x) \approx \frac{\sum_{i: C(x_i) \geq 100} \frac{f(x_i \wedge y, X^*, Y^*) \prod_{j=1}^{k} \frac{f(x_i \wedge x_j \wedge y, X^*, Y^*)}{f(x_i \wedge y, X^*, Y^*)}}{\sum_{y' \in Y} \left(f(x_i \wedge y', X^*, Y^*) \prod_{j=1}^{k} \frac{f(x_i \wedge x_j \wedge y', X^*, Y^*)}{f(x_i \wedge y', X^*, Y^*)} \right)}}{|\{i : C(x_i, X^*, Y^*) \geq 100\}|} \qquad (13)$$

where $C(x_i)$ is the number of training objects that have attribute value x_i.

As $\sum_{y' \in Y} \left(f(x_i \wedge y', X^*, Y^*) \prod_{j=1}^{k} \frac{f(x_i \wedge x_j \wedge y', X^*, Y^*)}{f(x_i \wedge y', X^*, Y^*)} \right)$ is invariant across classes, to select the class that maximizes the estimate of $P(y\,|\,x)$ it is necessary only to select the class that maximizes the following.

$$\frac{\sum_{i: C(x_i, X^*, Y^*) \geq 100} f(x_i \wedge y, X^*, Y^*) \prod_{j=1}^{k} \frac{f(x_i \wedge x_j \wedge y, X^*, Y^*)}{f(x_i \wedge y, X^*, Y^*)}}{|\{i : C(x_i, X^*, Y^*) \geq 100\}|} \qquad (14)$$

Table 1: Training time algorithm

```
INPUTS: training set $X^*, Y^*$,
        number of attributes $k$, and
        number of classes $m$
OUTPUTS: joint frequency vector $freq$,
         class frequency vector $cfreq$,
         attribute frequency vector $afreq$,
         attribute-value frequency vector $vfreq$, and
         item count $count$

Initialize frequencies
$count \leftarrow 0$
Initialize all elements of $freq$, $cfreq$, $afreq$, and $vfreq$ to 0

Accumulate frequencies
FOR EACH $x, y \in X^*, Y^*$
  $count \leftarrow count + 1$
  $cfreq[y] \leftarrow cfreq[y] + 1$
  FOR $i \leftarrow 1$ TO $k$
    IF $x_i$ is known
      $afreq[i] \leftarrow afreq[i] + 1$
      $vfreq[x_i] \leftarrow vfreq[x_i] + 1$
      FOR $j \leftarrow 1$ TO $k$
        IF $x_j$ is known
          $freq[y, x_i, x_j] \leftarrow freq[y, x_i, x_j] + 1$
        END IF
      END FOR
    END IF
  END FOR
END FOR
```

5. AVERAGED ONE-DEPENDENCE ESTIMATORS

Averaged One-Dependence Estimators (AODE) use (14) for classification. At training time they produce a three dimensional joint frequency table, indexed in one dimension by the values of the class and in the remaining two dimensions by the values of the attributes. The algorithm for this process is presented in Table 1. Note that we allow for missing attribute values but not for missing class values. To apply the algorithm to allow for missing class values it is possible to add a preprocess that removes all such objects.

At classification time we utilize the joint frequency table and (14) for classification as follows. We use the m-estimate [1] to produce conservative estimates of conditional probabilities from joint frequencies. To estimate $P(x_i|x_j \wedge y)$ we use an m-estimate with weight 0.5 and a prior defined by the frequency of the attribute value in the data as a whole. We allow for the possibility that some attribute values are missing, resulting in the following.

$$\hat{P}(x_i|x_j \wedge y) = \frac{freq[y, x_i, x_j] + 0.5}{freq[y, x_j, x_j] + 0.5 \times vfreq[x_i]} \qquad (15)$$

To estimate $P(x_i \wedge y)$ we use an m-estimate with weight 0.5 and a prior defined by the product of the frequencies of the two terms.

$$\hat{P}(x_i \wedge y) = \frac{freq[y, x_i, x_i] + 0.5}{afreq[i] + 0.5 \times vfreq[x_i] \times cfreq[y]} \tag{16}$$

The algorithm to return a vector containing probability estimates $\hat{P}(y|x)$ for each $y \in Y$ is presented in Table 2.

AODE can be viewed as Bayesian averaging with uniform priors over all plausible models that have all attributes depend upon a single attribute and the class. Another perspective from which AODE can be viewed is as ensemble learning.

Note, that while there is some superficial similarity between AODE and exact model averaging with naive Bayesian classifiers [2], the differences are very substantial. Both predict by averaging the estimated class probabilities of multiple models. Exact model averaging averages over naive Bayes (zero-dependence) models formed with different subsets of the available attributes. Hence, it cannot successfully model attribute inter-dependencies. In contrast, AODE averages over models that directly model attribute inter-dependencies and hence has the potential to overcome the attribute independence problem. Exact model averaging aggregates models formed by feature subset selection. AODE aggregates models formed by modelling alternative attribute inter-dependencies.

5.1 Computational complexity

The time complexity of training is $O(nk^2)$, where n is the number of training objects and k is the number of attributes. Further, the operations performed in the inner loops involve minimal computation, resulting in minor computational overheads for training for the numbers attributes normally encountered (up to hundreds of attributes). Training time complexity is linear with respect to training set size. Thus, AODE is an efficient algorithm for classification from large datasets.

The time complexity of classifying an object is $O(mk^2)$, where m is the number of classes and k the number of attributes. Again the operations performed within the inner loops involve little computation, resulting in computationally efficient classification for the numbers of classes and attributes normally encountered.

The main space requirement for the algorithm is for the table containing the joint frequencies, which is $O(mv^2)$, where m is the number of classes and v is the number of attribute values. For typical learning problems involving up to tens of classes and hundreds of attributes this entails relatively minor space requirements. Note that training can be performed by a single sequential scan through the data. There is no need to retain training data in memory.

5.2 Sensitivity to large numbers of attributes

Abstract analysis suggests that the performance of AODE will decline as the numbers of attributes increase. One dimension of the algorithm's sensitivity to large numbers of attributes is the time and space complexity of the algorithm. Another dimension is that the benefit of the algorithm's reduction in the attribute independence assumption may be diluted as the number of attributes increases. Consider a situation where there is a pairwise interdependence between two attributes A and B. The probability estimates in the

Table 2: Probability estimation algorithm

```
INPUTS: object $x$,
        number of attributes $k$,
        number of classes $m$,
        vector of the number of values for each attribute $v$,
        joint frequency vector $freq$,
        class frequency vector $cfreq$,
        count of objects for which a value is known
            for an attribute $afreq$,
        attribute-value frequency vector $vfreq$, and
        item count $count$
OUTPUT: conditional probability vector $prob$,

Initialize values
FOR $i \leftarrow 1$ TO $m$, $prob[i] \leftarrow 0.0$
$sum \leftarrow 0.0$
Calculate probabilities
FOR $y \leftarrow 1$ TO $m$
    $prob[y] \leftarrow 0.0$
    $attcount \leftarrow 0$
    FOR $i \leftarrow 1$ TO $k$
        IF $x_i$ is known AND $vfreq[x_i] \geq 100$
            $attcount \leftarrow attcount + 1$
            $p \leftarrow \hat{P}(x_i \wedge y)$
            FOR $j \leftarrow 1$ TO $k$
                IF $x_j$ is known, $p \leftarrow p \times \hat{P}(x_j|x_i \wedge y)$
            END FOR
            $prob[y] \leftarrow prob[y] + p$
        END IF
    END FOR
    If no attribute value occurs with sufficient frequency,
    revert to naive Bayes
    IF $attcount = 0$
        $prob[y] \leftarrow NB(x, y)$
    ELSE
        $prob[y] = prob[y]/attcount$
    END IF
    $sum \leftarrow sum + prob[y]$
END FOR
Normalize to obtain probabilities
FOR $y \leftarrow 1$ TO $m$, $prob[y] \leftarrow prob[y]/sum$
```

two submodels formed when all attributes depend on either A or B will be correct with respect to the interdependency. However, in the other submodels, each formed with all attributes depending upon another attribute C, the probability estimates will treat A and B as independent, except in so far as their interdependence is captured by their mutual interdependencies with C. In many cases interdependencies between two variables will be captured in part by their mutual interdependencies with other variables. For example, if we assume that *senility* and *nocturia* are both correlated with *age*. In this case *senility* and *nocturia* will be interdependent. However, for any given value of *age* they might be independent. Nonetheless, as the number of other attributes increases and hence the number of submodels averaged increases, the advantage gained from undoing individual interdependencies in individual submodels might be diluted.

6. EXPERIMENTS

To evaluate the performance of AODE, it was implemented in Weka [19] and compared to existing Weka implementations of naive Bayes (NB), LBR, TAN, and C4.5 (J48). Note that the implementations of LBR and TAN are our own implementations, created for previous research. The implementations of naive Bayes and C4.5 are part of the Weka distribution.

All four algorithms were applied to 39 data sets. These data sets are formed around a core of twenty-nine data sets used in previous related research [3; 21] augmented by a variety of larger data sets. These larger datasets were added as the original twenty-nine datasets were all relatively small and the techniques of LBR and TAN have greatest scope to improve upon NB when more data is available. Note that three further large datasets, Musk, Census-income, and Cover-type, are not included in these results because at least one of LBR or TAN could not complete processing for them within a one-fortnight time limit.

Each algorithm was applied to each data set using ten-fold cross validation on a dual-processor 1.7Ghz Pentium 4 Linux computer with 2Gb RAM. However, due to the long compute times of TAN and LBR, a small number of computations had to be performed on an alternative machine. This has not affected the error results obtained. These computations are identified and excluded from time analyses.

Numeric attributes were discretized using ten-bin discretization. Note that the decision tree learner is applied to the discretized data, rather than being allowed to select its own cut-points. Its error would be lower if it were allowed to select its own cut-points. However, we believe that the error of the probabilistic techniques could also be reduced by application of alternative discretization techniques [20]. All algorithms have been applied to the same discretized data in order to compare their performance on nominal data.

6.1 Relative Error

Table 3 presents the data sets used, the number of objects in each data set, the number of attributes by which each object is described, and the mean error of each algorithm on the data set. At the foot of the table the mean error across all data sets is provided together with the geometric mean of the error ratio obtained by dividing the error of AODE by the error of the alternative algorithm. AODE achieves the lowest mean error across all data sets. However, this is at best a gross measure of performance, as error rates across data sets are incommensurable. The error ratio corrects for this. The geometric mean is the appropriate approach to averaging ratio values such as the error ratio. A value less than 1.0 indicates an advantage to AODE. On this measure AODE registers very large advantage compared with NB and J48 and sizeable advantage compared to TAN and LBR. Note that we do not perform statistical tests of significance on differences in performance on individual data sets as the large number of such tests that would be entailed would result in extremely high levels of expected type 1 error. Instead we perform statistical tests on the win-draw-loss records, as presented in Table 4. This table presents the number of data sets for which AODE obtains lower error, equal error (measured to one decimal place), and higher error, with respect to each alternative algorithm. The outcome of a binomial sign test is also presented. This is the probability that the observed ratio of wins to losses or higher would be obtained by chance if both were equi-probable. As can be seen, AODE enjoys highly significant ratios of wins to losses against NB and J48. While AODE wins more often than it loses against TAN and LBR, these ratios are not significant at the 0.05 level.

Table 4: Win–Draw–Loss Records

	NB	**TAN**	**LBR**	**J48**
AODE WDL	23–8–8	18–6–15	20–4–15	27–2–10
p	0.005	0.364	0.250	0.004

Our abstract analysis of AODE identified a potential vulnerability of the algorithm when processing data sets utilizing large numbers of attributes. The two data sets with the most attributes are syncon and sonar. For the former AODE achieves substantially lower error than any of the other algorithms. For the latter AODE achieves lower error than all alternatives other than TAN. The next seven data sets in descending order of number of attributes are promoters, lung-cancer, chess, anneal, satellite, pioneer, and ionosphere. For one of these data sets, satellite, AODE achieves lower error than all alternatives. For two, promoters, and lung-cancer, AODE shares the lowest error with NB, LBR, and, in one case, TAN. For none of the remaining four data sets does the error of AODE reach the level of the highest of the alternatives. These outcomes suggest that the expected vulnerability of AODE to large numbers of attributes is not apparent for the numbers of attributes investigated in these experiments. The question of whether the expected vulnerability becomes a problem for larger numbers of attributes is an important area for future investigation.

6.2 Relative Compute Time

Table 5 presents the total compute time of each algorithm by data set. These times are the total time to run Weka to complete the cross-validation task and hence include data input as well as learning and classification. Note that due to the long compute times of LBR and TAN some computations were performed on an alternative computer and hence are excluded as they are incommensurable with the times listed. Note also that compute times on the compute server used are highly variable from run to run. Further, our implementations of AODE, TAN, and LBR are not optimized. Finally, the structure of this task is unfavorable to LBR,

Table 3: Mean error

	objects	atts	AODE	NB	TAN	LBR	J48
waveform	100000	22	3.6	6.9	4.4	3.7	3.9
adult	48842	15	16.7	18.0	16.0	15.0	15.7
letter-recognition	20000	17	13.2	30.0	16.5	16.0	19.4
sign	12546	9	29.3	38.6	26.6	20.9	20.4
pendigits	10992	17	2.2	12.9	3.6	3.3	9.8
pioneer	9150	37	4.1	9.8	3.5	4.8	4.3
satellite	6435	37	11.3	18.9	12.4	13.3	15.3
hypothyroid	3163	26	2.9	2.9	2.9	2.8	2.5
segment	2310	20	5.7	11.1	6.3	6.4	6.5
mfeat-mor	2000	7	30.2	30.7	30.0	30.0	30.1
german	1000	21	24.4	24.6	24.8	24.7	30.4
led	1000	8	26.4	26.2	25.9	26.2	27.1
ttt	958	10	26.1	29.5	28.6	14.6	15.1
anneal	898	39	5.0	5.5	4.0	4.1	9.2
vehicle	846	19	27.9	39.5	31.3	31.4	30.3
breast-cancer-wisc.	699	10	2.7	2.6	2.6	2.6	5.7
crx	690	16	13.6	15.1	14.4	14.6	15.5
balance-scale	625	5	9.9	8.6	8.6	8.6	36.6
syncon	600	61	1.0	3.0	3.0	2.5	23.0
chess	551	40	11.4	12.7	10.0	11.1	8.3
house-votes-84	435	17	6.4	9.9	6.7	7.1	4.1
horse-colic	368	22	20.4	20.1	18.5	19.0	15.0
ionosphere	351	35	9.4	9.1	9.4	9.1	13.7
bupa	345	7	36.5	36.8	39.7	36.8	39.7
primary-tumor	339	18	48.1	49.0	49.9	49.9	57.8
cleveland	303	14	19.1	16.5	16.5	16.5	21.5
hungarian	294	14	15.7	15.3	15.7	16.0	20.4
heart	270	14	15.6	15.2	15.2	15.2	23.7
new-thyroid	215	6	7.0	8.4	7.4	8.4	7.0
glass	214	10	24.8	25.2	24.3	25.7	23.8
sonar	208	61	24.5	25.5	23.6	26.0	32.2
wine	178	14	3.4	3.4	3.4	3.4	21.4
hepatitis	155	20	15.5	16.1	16.1	14.8	16.8
iris	150	5	6.7	6.7	6.0	6.7	4.0
echocardiogram	131	7	27.5	27.5	28.2	28.2	35.9
promoters	106	58	8.5	8.5	8.5	8.5	21.7
post-operative	90	9	28.9	28.9	30.0	30.0	28.9
labor-neg	57	17	3.5	3.5	3.5	10.5	29.8
lung-cancer	32	57	46.9	46.9	50.0	46.9	53.1
Mean error			16.3	18.4	16.6	16.3	20.5
Geometric mean error ratio			1.00	0.82	0.96	0.95	0.72

which is relatively efficient for small test sets and relatively inefficient for large test sets. Hence, these results should be regarded as broadly indicative only. The mean CPU time across all data sets is also presented. Finally, we present the geometric mean of the time for AODE divided by the time for each alternative algorithm. These latter values are presented for all algorithms excluding results for waveform and pioneer as times are not available for all algorithms on these data sets. They are also presented including all results for AODE, NB, and J48 only.

Where there are large numbers of attributes the compute time of AODE increases significantly from that of naive Bayes. This is especially apparent for pioneer. However, while up to 12 times slower than naive Bayes, the compute times are still low, in no case exceeding two minutes to complete ten-fold cross validation. For most data sets the compute time is lower than the decision tree learner. For no data set does the compute time of AODE exceed that of TAN or LBR. For most data sets it is dramatically faster.

The geometric mean time ratios indicate that naive Bayes holds a substantial advantage over AODE, but that AODE holds a substantial advantage over the decision tree algorithm and a massive advantage over TAN and LBR.

7. CONCLUSION

AODE is a new classification learning algorithm that weakens the attribute independence assumption of naive Bayes without undue increase in computational complexity. In these preliminary investigations, this algorithm is demonstrated to deliver accuracy at least comparable to the state-of-the-art LBR and TAN algorithms without their high computational cost. The training time of the algorithm is linear with respect to data set size. Training requires only a single sequential scan of the data. These features make the algorithm highly attractive for processing very large data sets.

Our theoretical analysis of the algorithm leads us to expect that it is more suited to data sets described by fewer rather than more attributes. However, no vulnerability to large numbers of attributes was apparent in our experiments using data sets with as many as 61 attributes.

We believe that AODE is successful in achieving our aim of significantly and substantially reducing the error of naive Bayes without substantially increasing compute time. We believe that the resulting algorithm is extremely well suited to applications that require efficient computation, such as on-line applications, as well as to applications requiring processing of very large training sets.

8. REFERENCES

[1] B. Cestnik, I. Kononenko, and I. Bratko. ASSISTANT 86: A knowledge-elicitation tool for sophisticated users. In I. Bratko and N. Lavrač, editors, *Progress in Machine Learning*, pages 31–45. Sigma Press, Wilmslow, 1987.

[2] D. Dash and G. F. Cooper. Exact model averaging with naive Bayesian classifiers. In *Proceedings of the Nineteenth International Conference on Machine Learning, ICML 2002*, pages 91–98, Sydney, July 2002. Morgan Kaufmann.

[3] P. Domingos and M. Pazzani. Beyond independence: Conditions for the optimality of the simple Bayesian classifier. In *Proceedings of the Thirteenth International Conference on Machine Learning*, pages 105–112, Bari, Italy, 1996. Morgan Kaufmann.

[4] N. Friedman and M. Goldszmidt. Building classifiers using Bayesian networks. In *AAAI-96*, pages 1277–1284, 1996.

[5] I. J. Good. *Probability and the Weighing of Evidence.* Charles Griffin and Co. Ltd., London, 1950.

[6] E. Keogh and M. Pazzani. Learning augmented Bayesian classifiers: A comparison of distribution-based and classification-based approaches. In *International Workshop on Artificial Intelligence and Statistics*, pages 225–230, 1999.

[7] R. Kohavi. Scaling up the accuracy of naive-Bayes classifiers: A decision-tree hybrid. In *KDD-96*, Portland, Or, 1996.

[8] I. Kononenko. Comparison of inductive and naive Bayesian learning approaches to automatic knowledge acquisition. In B. Wielinga, J. Boose, B. Gaines, G. Schreiber, and M. van Someren, editors, *Current Trends in Knowledge Acquisition.* IOS Press, Amsterdam, 1990.

[9] I. Kononenko. Semi-naive Bayesian classifier. In *ECAI-91*, pages 206–219, 1991.

[10] P. Langley. Induction of recursive Bayesian classifiers. In *Proceedings of the 1993 European Conference on Machine Learning*, pages 153–164, Vienna, 1993. Springer-Verlag.

[11] P. Langley and S. Sage. Induction of selective Bayesian classifiers. In *Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence*, pages 399–406, Seattle, WA, 1994. Morgan Kaufmann.

[12] D. D. Lewis. Naive Bayes at forty: The independence assumption in information retrieval. In *ECML-98: Proceedings of the Tenth European Conference on Machine Learning*, pages 4–15, Chemnitz, Germany, April 1998. Springer.

[13] M. J. Pazzani. Constructive induction of Cartesian product attributes. In *ISIS: Information, Statistics and Induction in Science*, pages 66–77, Melbourne, Aust., August 1996. World Scientific.

[14] M. Sahami. Learning limited dependence Bayesian classifiers. In *Proceedings 2nd International Conference on Knowledge Discovery and Data Mining*, pages 334–338. AAAI Press, 1996.

[15] M. Singh and G. M. Provan. Efficient learning of selective Bayesian network classifiers. In *Proceedings of the 13th International Conference on Machine Learning*, pages 453–461, Bari, 1996. Morgan Kaufmann.

[16] Z. Wang and G. I. Webb. Comparison of lazy bayesian rule and tree-augmented bayesian learning. In *To appear in Proceedings of the IEEE International Conference on Data Mining, ICDM-2002*, 2002.

Table 5: Compute time in CPU seconds

	objects	atts	AODE	NB	TAN	LBR	J48
waveform	100000	22	108	33	*	1548801	702
adult	48842	15	19	10	4792	183699	522
letter-recognition	20000	17	34	8	8710	147454	302
sign	12546	9	4	3	170	5376	112
pendigits	10992	17	16	4	2714	12395	73
pioneer	9150	37	98	8	*	*	18
satellite	6435	37	25	4	20902	15176	1
hypothyroid	3163	26	6	2	474	2937	6
segment	2310	20	4	2	409	1384	8
mfeat-mor	2000	7	1	1	9	95	4
german	1000	21	2	1	39	100	2
led	1000	8	1	1	8	145	1
ttt	958	10	1	1	10	31	1
anneal	898	39	5	1	1323	1001	6
vehicle	846	19	2	1	128	60	3
breast-cancer-wisconsin	699	10	1	1	2	13	1
crx	690	16	1	1	26	71	2
balance-scale	625	5	1	1	1	3	1
syncon	600	61	6	1	510	235	7
chess	551	40	1	1	414	227	2
house-votes-84	435	17	1	1	28	15	1
horse-colic	368	22	1	1	82	19	2
ionosphere	351	35	2	1	58	16	2
bupa	345	7	1	1	2	2	1
primary-tumor	339	18	1	1	18	145	1
cleveland	303	14	1	1	4	4	1
hungarian	294	14	1	1	7	6	1
heart	270	14	1	1	3	4	1
new-thyroid	215	6	1	1	1	1	1
glass	214	10	1	1	3	2	1
sonar	208	61	2	1	330	18	2
wine	178	14	1	1	2	2	1
hepatitis	155	20	1	1	7	4	1
iris	150	5	1	1	1	1	1
echocardiogram	131	7	1	1	1	1	1
promoters	106	58	1	1	52	5	1
post-operative	90	9	1	1	1	1	1
labor-neg	57	17	1	1	2	1	1
lung-cancer	32	57	1	1	44	3	1
Mean CPU time$^+$			4	2	1116	10018	29
Geometric mean time ratio$^+$			1.00	1.51	0.05	0.04	0.68
Mean CPU time			9	3	*	*	46
Geometric mean time ratio			1.00	1.64	*	*	0.69

* indicates computation performed on alternative computer and hence included.
$^+$ Values exclude waveform and pioneer for which some values are not available.

[17] G. I. Webb. Candidate elimination criteria for Lazy Bayesian Rules. In *Proceedings of the Fourteenth Australian Joint Conference on Artificial Intelligence*, pages 545–556, Adelaide, December 2001. Springer.

[18] G. I. Webb and M. J. Pazzani. Adjusted probability naive Bayesian induction. In *Proceedings of the Eleventh Australian Joint Conference on Artificial Intelligence*, pages 285–295, Brisbane, Australia, 1998. Springer.

[19] I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations.* Morgan Kaufmann, San Francisco, CA, 1999.

[20] Y. Yang and G. I. Webb. Proportional k-interval discretization for naive-Bayes classifiers. In *12th European Conference on Machine Learning (ECML'01)*, pages 564–575. Springer, 2001.

[21] Z. Zheng and G. I. Webb. Lazy learning of Bayesian Rules. *Machine Learning*, 41(1):53–84, 2000.

[22] Z. Zheng, G. I. Webb, and K. M. Ting. Lazy Bayesian Rules: A lazy semi-naive Bayesian learning technique competitive to boosting decision trees. In *Proceedings of the Sixteenth International Conference on Machine Learning (ICML-99)*, pages 493–502, Bled, Slovenia, 1999. Morgan Kaufmann.