Noname manuscript No.
(will be inserted by the editor)

Selective AnDE for large data learning: a low-bias memory constrained approach

Shenglei Chen · Ana M. Martínez · Geoffrey I. Webb · Limin Wang

Received: Jun 06, 2015 /Revised: Jan 17, 2016 /Accepted: Mar 06, 2016

Abstract Learning from data that are too big to fit into memory poses great challenges to currently available learning approaches. Averaged n-Dependence Estimators (AnDE) allows for a flexible learning from out-of-core data, by varying the value of n (number of super parents). Hence AnDE is especially appropriate for learning from large quantities of data. Memory requirement in AnDE, however, increases combinatorially with the number of attributes and the parameter n. In large data learning, number of attributes is often large and we also expect high n to achieve low bias classification. In order to achieve the lower bias of AnDE with higher n but with less memory requirement, we propose a memory constrained selective AnDE algorithm, in which two passes of learning through training examples are involved. The first pass performs attribute selection on super parents according to available memory, whereas the second one learns an AnDE model with parents only on the selected attributes. Extensive experiments show that the new selective AnDE has considerably lower bias and prediction error relative to An'DE, where $n' = n - 1$, while maintaining the same space complexity and similar time complexity. The proposed algorithm works well on categorical data. Numerical data sets need to be discretized first.

Shenglei Chen
Department of E-Commerce, Nanjing Audit University, Nanjing, 211815, China
Faculty of Information Technology, Monash University, VIC 3800, Australia
Key Laboratory of Symbolic Computation and Knowledge Engineering of Ministry of Education, Jilin University, Changchun, 130012, China
E-mail: tristan_chen@126.com

Ana M. Martínez
Faculty of Information Technology, Monash University, VIC 3800, Australia
Aalborg University, DK-9220 Aalborg, Denmark

Geoffrey I. Webb
Faculty of Information Technology, Monash University, VIC 3800, Australia

Limin Wang
Key Laboratory of Symbolic Computation and Knowledge Engineering of Ministry of Education, Jilin University, Changchun, 130012, China

Keywords Attribute selection · Bayesian classification · AnDE · Large data

1 Introduction

Many applications in e.g. bioinformatics, IT-security and text-classification come with millions of training examples. Learning from these large data sets poses great challenges to currently available machine learning algorithms, one of which is that the entire dataset probably cannot be loaded into memory. One way to address this problem is to learn from a sample of the dataset, which will potentially lose information implicit in the data as a whole. Another alternative is to learn from out-of-core data [27]. While information will not be lost in this approach, data access will be very expensive. Hence approaches requiring limited number of passes through training data become more desirable.

AnDE is one family of Bayesian learning algorithms that can learn in a single pass through training examples [22, 23]. Developed based on Naïve Bayes (NB) learning [6], AnDE preserves many favorable features of Naïve Bayes, including no model selection and direct prediction of class probabilities [23]. What is more desirable is that AnDE has linear time complexity with respect to the number of training examples, which allows learning from a single pass through training data and hence enables out-of-core learning possible.

Besides the feature of single pass of data access, AnDE provides lower bias than NB, because it allows every attribute to depend on n other attributes, where the depended n attributes are called super parents. It is well known that classification error can be decomposed into bias and variance [14]. The variance component will be lower when learning from large data than when learning from small data sets [2], since probability estimates are made considering a larger number of points. A low value of the bias component of the classification error is highly appealing to large data learning, where generally more complex multivariate relationships must be captured. In AnDE, bias can be made progressively lower as the parameter n increases.

However, memory requirement of AnDE increases combinatorially with the number of attributes, denoted by a, and the parameter n. And it is often the case that a large data set contains not just a large amount of training examples but also a large number of attributes. Additionally, higher n is desired to obtain low bias from large datasets. Thus, this poses conflicting objectives between lower bias and less memory requirement.

In order to achieve the lower bias of AnDE with higher n but with less memory requirement, we propose a memory constrained selective AnDE algorithm which performs attribute selection in AnDE with higher n according to certain memory restrictions. Two passes of learning through training examples are involved in this scheme. The first pass generates the information theoretic statistics required for selecting the parent attributes. The second pass is used to calculate the sample distribution for AnDE, in which only the attributes selected in the previous step are used as super parents. By this means, we choose a very productive subset of all attribute subsets much more efficiently than previous attribute selection techniques [24, 25, 28], which involve a passes learning on the data in the worst case.

As a proof of concept and to show the validity of this setting, we show how this memory constrained selective AnDE obtains similar performance to regular AnDE with the same n and better performance than regular AnDE with lower n. That is, experimental results show that selective AnDE, where memory has been constrained so as to match the memory of An'DE, where $n' = n - 1$, performs similarly to regular AnDE (which requires significantly more memory) and can achieve lower bias and will deliver more accurate classification than regular An'DE, where $n' = n - 1$ (which requires the same memory).

It is worthwhile to note that the proposed algorithm can be used only on categorical data. If we deal with numerical data sets directly, the memory and runtime requirements would be not acceptable as the numerical attribute might take an infinite number of values. As a result, for data sets with numerical attributes, the attributes have to be discretized first.

The paper is organized as follows: we provide a survey of related work in Section 2. Section 3 reviews some preliminaries, including Naïve Bayes and Averaged n-Dependence Estimators. We present how the number of selected parent attributes can be approximated and what metrics can be used for attribute ranking in Section 4, and propose a two passes algorithm for selective AnDE. Section 5 presents experimental evaluation of our proposed approach. Section 6 provides conclusions and directions for future research.

2 Related work

Since AODE was first proposed in 2005 [22], which is a special case of AnDE with n equal to one, there have been many attempts to perform attribute selection in AnDE from different perspectives. Here we give a brief survey of them.

Yang et al [24,25] investigated how to select SPODEs (Super Parent One Dependence Estimator) within AODE. They presented five different metrics to rank SPODEs, including two popular information theoretic metrics, Minimum Description Length (MDL) [7], Minimum Message Length (MML), and three accuracy-based empirical metrics, Leave One Out (LOO), Backward Sequential Elimination (BSE), and Forward Sequential Addition (FSA). For MDL, MML and LOO, they selected those SPODEs whose metric values were lower than the mean value. For BSE and FSA, the process produced a SPODE ensembles (where a is the number of attributes), from size 1 to a; out of these a ensembles, the one with the lowest classification error was selected. They demonstrated that model selection in AODE did make a difference and empirical metrics outperformed information theoretic metrics at the cost of higher training time overhead. Zheng et al [28] also explored the attribute selection problem in AODE in the framework of BSE and FSA, but rather concentrated on the comparison of parent selection and children selection. They suggested that elimination of children was more effective.

These studies agree that BSE is very effective for attribute selection in AODE. However, it involves multiple passes learning on the data, which can amount to a passes in the worst case. This is clearly not a suitable approach for large data learning.

Chen et al [4] proposed a new attribute selection algorithm for AODE, referred to as ASAODE. This algorithm can search in a large model space, while it requires only a single extra pass through the training data, resulting in a computationally efficient two-pass learning algorithm. Experimental results indicate that ASAODE significantly reduces AODE's bias at the cost of a modest increase in training time.

Besides these studies on attribute selection in AODE, there is also some well-known work that improves AODE. Jiang et al [12] proposed weightily AODE based on the observation that in AODE, each One-Dependence Estimator (ODE) is treated equally, while attributes do not play the same role in classification for many real world applications. Weightily AODE uses mutual information between the super-parent and the class as the weight.

Zheng et al [29] proposed a technique called *Subsumption Resolution* for AODE, which identifies pairs of attribute values such that one appears to subsume the other and deletes the generalization. This idea is inspired by the fact that one value of one attribute might be a generalization of one value of the other. For example, consider *Gender* and *Pregnant* as two attributes, then $Pregnant = yes$ implies that $Gender = female$. Therefore, $Gender = female$ is a generalization of $Pregnant = yes$. Likewise, $Gender = male$ implies that $Pregnant = no$, so $Pregnant = no$ is a generalization of $Gender = male$. Where one value x_i is a generalization of another value, x_j, $P(y|x_i, x_j) = P(y|x_j)$. In consequence dropping the more general value from any calculations should not harm any posterior probability estimates, whereas assuming independence between them may.

3 Preliminaries

In this section, we present some preliminaries, including Naïve Bayes and Averaged n-Dependence Estimators.

The classification task is assumed as follows. Given a training sample $\mathcal{T}$ of t classified objects, we are required to predict the probability $\mathrm{P}(y \mid \mathbf{x})$ that a new example $\mathbf{x} = \langle x_1, \ldots, x_a \rangle$ belongs to some class y, where x_i is the value of the attribute $\mathbf{x}_i$ and $y \in \{c_1, \ldots, c_k\}$.

3.1 Naïve Bayes

From the definition of conditional probability, we have

$$\mathrm{P}(y \mid \mathbf{x}) = \mathrm{P}(y, \mathbf{x}) / \mathrm{P}(\mathbf{x}).$$

As $\mathrm{P}(\mathbf{x}) = \sum_{i=1}^{k} \mathrm{P}(c_i, \mathbf{x})$ and $y \in \{c_1, \ldots, c_k\}$, it is reasonable to consider $\mathrm{P}(\mathbf{x})$ as the normalizing constant and estimate only the joint probability $\mathrm{P}(y, \mathbf{x})$ in the remainder of this paper.

If example $\mathbf{x}$ does not appear frequently enough in the training data, we cannot directly derive accurate estimates of $\mathrm{P}(y, \mathbf{x})$ and must extrapolate these estimates from observations of lower-dimensional probabilities in the data [23]. Applying the definition of conditional probabilities again, we have

$$\mathrm{P}(y, \mathbf{x}) = \mathrm{P}(y)\mathrm{P}(\mathbf{x} \mid y).$$

The first term $\mathrm{P}(y)$ on the right side can be sufficiently accurately estimated from the sample frequencies, if the number of classes, k, is not a huge number. For the second term $\mathrm{P}(\mathbf{x} \mid y)$, NB assumes the attributes are independent of each other given the class and calculates by the following formula,

$$\mathrm{P}(\mathbf{x} \mid y) = \prod_{i=1}^{a} \mathrm{P}(x_i \mid y). \tag{1}$$

Consequently NB calculates the joint probability $\mathrm{P}(y, \mathbf{x})$ according to the following formula,

$$\mathrm{P}_{\mathrm{NB}}(y, \mathbf{x}) = \mathrm{P}(y) \prod_{i=1}^{a} \mathrm{P}(x_i \mid y). \tag{2}$$

Thus, NB classifies example $\mathbf{x}$ by selecting

$$\arg\max_{y} \left(\hat{\mathrm{P}}(y) \prod_{i=1}^{a} \hat{\mathrm{P}}(x_i \mid y) \right).$$

Where $\hat{\mathrm{P}}(y)$ and $\hat{\mathrm{P}}(x_i \mid y)$ are estimates of the respective probabilities derived from the frequencies of their respective arguments in the training sample, with possible correction such as Laplace estimate.

We can obtain estimates of the probabilities $\mathrm{P}(y \mid \mathbf{x})$ by normalizing across all possible classes, allowing the classifier to predict not just the class, but the probability of each class [18].

3.2 Averaged n-Dependence Estimators

Although some violations of the attribute independence assumption do not matter [5], it is clear that many do. Consequently there have been and still are increasing interests on developing techniques to alleviate the attribute independence assumption while retaining NB's desirable simplicity and efficiency [9,19,30]. In terms of large data, this requirement becomes as important as ever, since scalable but powerful classifiers are required.

Among these significant developments is AnDE [23], which relaxes the attribute independence assumption and averages over all possible n-dependence estimators, with the aim of reducing the inductive bias in the classifier. To be specific, instead of using (2) to estimate the joint probability $\mathrm{P}(y, \mathbf{x})$, A$n$DE assumes every attribute depends on a subset of attributes of size n known as parent attributes. And in order to simplify the calculation, it assumes that each attribute depends on the same parent attributes set $\mathbf{p}$. Joint probability $\mathrm{P}(y, \mathbf{x})$ for some $\mathbf{p}$ is calculated as following,

$$\mathrm{P}(y, \mathbf{x}) = \mathrm{P}(y, \mathbf{x}_{\mathbf{p}}) \prod_{i=1}^{a} \mathrm{P}(x_i \mid y, \mathbf{x}_{\mathbf{p}}), \tag{3}$$

where $\mathbf{x}_{\mathbf{p}}$ is the set of values of attributes in $\mathbf{p}$ corresponding to example $\mathbf{x}$.

When we try to select a subset $\mathbf{p}$ of size n from a attributes, we have $C(a, n) = a!/(n!(a-n)!)$ possible options. For each possible set of parents, we can build one model. The average across all models gives a final probability. So joint probability in AnDE is calculated by:

$$\mathrm{P}_{\mathrm{A}n\mathrm{DE}}(y, \mathbf{x}) = \frac{1}{C(a, n)} \sum_{\mathbf{p}} \mathrm{P}(y, \mathbf{x}_{\mathbf{p}}) \prod_{i=1}^{a} \mathrm{P}(x_i \mid y, \mathbf{x}_{\mathbf{p}}), \tag{4}$$

where $\mathbf{p}$ ranges over all size-n subsets of attributes.

When training the AnDE classifier, we need to form an $(n + 2)$-dimensional probability table, which contains the observed frequency for each combination of $n + 1$ attribute values and the class labels. The space complexity of the table is $\mathcal{O}(kC(a, n + 1)\bar{v}^{n+1})$, where $\bar{v}$ is the average number of values per attribute. It is worthwhile to note that AnDE can only work well on categorical data. Numerical data sets need to be discretized first. The time complexity of compiling it is $\mathcal{O}(tC(a, n + 1))$, as we need to update each entry for every combination of the $n + 1$ attribute-values for every instance.

It is evident that AnDE has linear time complexity with respect to the number of training examples, which allows single pass learning through training examples and makes out-of-core learning for large data set possible. On the other hand, as every attribute is assumed to depend on its parent attributes, which is more coincident with the characteristics of real data sets, AnDE has lower bias than NB. And as n increases, AnDE achieve lower bias at the cost of higher variance [23]. This low bias characteristic, combined with the single pass learning, makes AnDE well suited to large data learning, where variance is generally low.

However, as we can see from the space complexity, the memory requirement in AnDE increase combinatorially with the number of attributes and the parameter n. Thus, higher n does not only mean lower bias but also higher memory requirement. In the next section we present a new approach that combines the low bias characteristic of AnDE with higher n with the low memory requirement of AnDE with smaller n.

4 Selective AnDE

NB assumes that all attributes are independent of each other, which is not the case in most real-life data sets. AnDE assumes that each attribute depends on its parent attributes which range over all size-n subsets of the entire attribute set. This means AnDE requires large amount of memory for high n and large number of attributes.

Here we propose a trade-off between these two strategies, in which parent attributes (or super parents) range over all size-n subsets of s selected attributes, rather than the entire set of a attributes, where $s \leq a$. This strategy reduces the memory requirement and can resolve the large memory problem brought by increasing n. At the same time, it is able to retain the low bias characteristic of original AnDE. So compared to regular AnDE with lower n, selective AnDE with higher n can be expected to achieve higher accuracy while requiring comparative memory.

There are a number of factors of which we ideally want the attribute selection mechanism to take account. We want the final collection of parent-child pairs to include those that are most predictive of the class as well as those for which the violations of the attribute-independence assumption between parents and children are most harmful to the classifier. The most effective way to assess the latter is through wrapper evaluation. We can obtain a good heuristic approximation to the former by ranking parents and children on their individual capacity to predict the class, which we assess using mutual information.

As a result, we can first use a memory criterion to approximate the number of selected attributes. That is, the memory requirement of selective AnDE with higher n should be comparable to the memory of AnDE with lower n. Then we rank the attributes and select the top s attributes.

The following sections include details in which this process is carried out. Section 4.1 presents an analysis to approximate the number of selected attributes according to AnDE with a lower value of n. Section 4.2 includes three different options to evaluate the classification power of an attribute based on information theoretic measures. Section 4.3 presents the selective AnDE algorithm. The space and time complexity analyses are presented in Section 4.4.

4.1 Approximating the Number of Selected Attributes

In order to calculate s, the number of selected parent attributes, we need to estimate the memory requirements before selecting and after. As it is mentioned afore, the space complexity of regular AnDE is $\mathcal{O}(kC(a,n+1)\bar{v}^{n+1})$, where $C(a,n+1)$ is the binomial coefficient when $n+1$ elements are chosen from a pool of a elements.

When we compute the memory requirement of selective AnDE, we need to consider two parts. One is the memory complexity for s selected attributes. As the probability table contains the observed frequency for each combination of $n+1$ attribute values and the class label, the space complexity of the table is $\mathcal{O}(kC(s,n+1)\bar{u}^{n+1})$, where $\bar{u}$ is the average number of values for the selected attributes. The second is the memory requirement of storing the instance counts for the $a-s$ unselected attributes given the s selected attributes. This is due to the fact that we actually want to select only the parents, but not the children. There should be one dimensional attribute in which all the attributes are included. Hence we should also consider the joint probability of the $a-s$ unselected attributes with other n parent attributes and the class. This requires a memory space of $\mathcal{O}(kC(s,n)*\bar{u}^n*(a-s)*\bar{r})$, where $\bar{r}$ is the average number of values for the unselected attributes. So the overall memory complexity is $\mathcal{O}(k\bar{u}^n(C(s,n+1)\bar{u}+C(s,n)*(a-s)*\bar{r}))$.

We expect this memory to be comparable to the memory of An'DE, where $n'<n$. So we get the following equation,

$$k\bar{u}^n(C(s,n+1)\bar{u}+C(s,n)*(a-s)*\bar{r}) = kC(a,n'+1)\bar{v}^{n'+1}. \tag{5}$$

This gives the criterion to compute s.

Note that in a practical scenario, the number of attributes to be selected can also be calculated according to the total main memory available. The above detailed approach to match versions of AnDE with different values of n is specially used to stress the validity of this approach.

4.2 Metrics for Attribute Ranking

After the number of desired attributes is calculated, we need to rank the attributes and select the top s attributes. Because we want to minimize the accuracy loss resulted from selecting attributes, we should keep those attributes which are more correlated with the class than others. We need to find some metrics to evaluate this correlation.

Previous research findings suggest that correlation metrics based on information theory are more powerful than metrics based on classical linear correlation for classification purposes [26] and are more widely used in classification fields [9, 19].

Here we investigate three different metrics from information theory: mutual information, conditional mutual information and a hybrid combination of both.

4.2.1 Mutual information

Mutual information between two random variables describes how much information one random variable bears on the other [15]. More precisely, mutual information between random variables $\mathbf{X}$ and $\mathbf{Y}$ is defined as:

$$I(\mathbf{X},\mathbf{Y}) = H(\mathbf{X}) - H(\mathbf{X} \mid \mathbf{Y}) = \sum_{y\in\mathbf{Y}}\sum_{x\in\mathbf{X}} P(x,y)log_2\frac{P(x,y)}{P(x)P(y)}, \tag{6}$$

where $H(\mathbf{X}) = -\sum_x P(x)logP(x)$ is the entropy of $\mathbf{X}$, which roughly measures the amount of information carried by $\mathbf{X}$, and $H(\mathbf{X} \mid \mathbf{Y}) = -\sum_y P(y)\sum_x P(x \mid y)logP(x \mid y)$ is the conditional entropy, which measures the entropy of $\mathbf{X}$ when we know the value of $\mathbf{Y}$.

For classification purpose, we consider attributes and class as random variables. As mutual information between attribute $\mathbf{X}_i$ and class $\mathbf{Y}$ measures how much information attribute $\mathbf{X}_i$ provides about class $\mathbf{Y}$, an attribute with higher mutual information value is considered to be more informative to the class. Consequently, as the first approach, we can use mutual information to rank the attributes.

4.2.2 Conditional mutual information

Conditional mutual information between random variables $\mathbf{X}$ and $\mathbf{Y}$ given the value of $\mathbf{Z}$ is defined as:

$$I(\mathbf{X},\mathbf{Y} \mid \mathbf{Z}) = \sum_{x\in\mathbf{X},y\in\mathbf{Y},z\in\mathbf{Z}} P(x,y,z)log_2\frac{P(x,y \mid z)}{P(x \mid z)P(y \mid z)}, \tag{7}$$

Roughly speaking, this function measures the information that $\mathbf{X}$ provides about $\mathbf{Y}$ when the value of $\mathbf{Z}$ is known.

In the classification context, $\mathbf{Z}$ is considered to be another attribute different from $\mathbf{X}$. If we sum $I(\mathbf{X},\mathbf{Y} \mid \mathbf{Z})$ across all possible attributes $\mathbf{Z}$, then we get a second metric between $\mathbf{X}$ and $\mathbf{Y}$.

4.2.3 Direct rank using both mutual information and conditional mutual information

The above two strategies compute some metrics and then rank the attributes. They are quite intuitive, but do not consider the correlation among the selected attributes. Here we give a third ranking approach based on both mutual information and conditional mutual information, which ranks the attributes directly.

At the beginning, the sorted attribute set $\mathcal{A}_s$ is empty. When we select the first attribute, we select the attribute with the largest mutual information with the class and add it to $\mathcal{A}_s$. When we select next attribute from the unsorted attribute set $\mathcal{A}$, we should consider not only the relationship between the attribute and the class, but also the influence of the selected attributes. So we select the attribute from the unsorted attributes which has the largest mutual information with the class conditioned on the selected attributes. In order to achieve this, we loop through each selected attribute, and assess the conditional mutual information between the candidate attribute and the class conditioned on the selected attribute. We take the minimum of these values as an estimate of the conditional mutual information between the candidate and the class conditioned on all the selected attributes. Then from all the minimal conditional mutual information estimates, we select the attribute with the maximal conditional mutual information and add it to $\mathcal{A}_s$. This process continues until the unsorted attribute set $\mathcal{A}$ becomes empty. The sequence of attributes added to the sorted attribute set $\mathcal{A}_s$ gives an attribute rank. The above attributes ranking procedure is summarised in Algorithm 1.

Algorithm 1 Direct attributes ranking algorithm.

```
 1: $\mathcal{A}$: set of unsorted attributes, initialized to contain all attributes
 2: $\mathcal{A}_s$: set of sorted attributes, initialized to be empty
 3: $\mathbf{X}_{max} = \arg\max_{\mathbf{X}\in\mathcal{A}} I(\mathbf{X}, \mathbf{Y})$
 4: Add $\mathbf{X}_{max}$ to $\mathcal{A}_s$
 5: while $\mathcal{A} \neq \emptyset$ do
 6:     for $\mathbf{X} \in \mathcal{A}$ do
 7:         $I(\mathbf{X}) = \min_{\mathbf{Z}\in\mathcal{A}_s} I(\mathbf{X}, \mathbf{Y} \mid \mathbf{Z})$
 8:     end for
 9:     $\mathbf{X}_{max} = \arg\max_{\mathbf{X}\in\mathcal{A}} I(\mathbf{X})$
10:     Add $\mathbf{X}_{max}$ to $\mathcal{A}_s$, AND Remove $\mathbf{X}_{max}$ from $\mathcal{A}$
11: end while
```

4.3 Two Passes Algorithm

As is indicated above, the selective AnDE algorithm consists of two passes on the data. When we try to compute the mutual information, we need the joint distribution of one attribute and the class across all the examples. For the conditional mutual information, we need the joint distribution of each pair of attributes and the class. So before we can compute these metrics, the first pass of learning through the examples should be performed to obtain the joint distribution.

After we select the top s attributes, we need a second pass of learning through the examples to obtain the joint probability distributions on the selected attributes. Algorithm 2 presents the pseudo-code of this second pass in selective A2DE. Note that here we store only the observed count of each combination of 3 attributes and the class label. With these data we can easily compute the frequency of each combination when necessary. This process in selective A3DE is similar. The only difference lies in that we need to store the count of each combination of 4 attributes and the class label. So the overall memory constrained selective AnDE involves two passes of learning through training examples. Algorithm 3 highlights the key steps of the training procedure.

Algorithm 2 Second pass of learning through training data in selective A2DE.

```
1: $\mathcal{U}$ : set of unselected attributes
2: $\mathcal{S}$ : set of selected attributes ordered by one metric from Section 4.2
3: $Count1$ : vector of observed counts of combination of 3 selected attribute values and the
   class label
4: $Count2$ : vector of observed counts of combination of 2 selected attribute values, 1 unse-
   lected attribute value and the class label
5: for instance $inst \in \mathcal{T}$ do
6:     $y$ = value of class label in $inst$
7:     for $\mathbf{X_1} \in \mathcal{S}$ do
8:         $x_1$ = value of attribute $\mathbf{X_1}$ in $inst$
9:         for $\mathbf{X_2} \in \mathcal{S}$ AND $\mathbf{X_2}$ precedes $\mathbf{X_1}$ do
10:            $x_2$ = value of attribute $\mathbf{X_2}$ in $inst$
11:            for $\mathbf{X_3} \in \mathcal{S}$ AND $\mathbf{X_3}$ precedes $\mathbf{X_2}$ do
12:                $x_3$ = value of attribute $\mathbf{X_3}$ in $inst$
13:                increase the element in $Count1$ with index $(\mathbf{X_1}, x_1, \mathbf{X_2}, x_2, \mathbf{X_3}, x_3, y)$ by 1
14:            end for
15:            for $\mathbf{X_3} \in \mathcal{U}$ do
16:                $x_3$ = value of attribute $\mathbf{X_3}$ in $inst$
17:                increase the element in $Count2$ with index $(\mathbf{X_1}, x_1, \mathbf{X_2}, x_2, \mathbf{X_3}, x_3, y)$ by 1
18:            end for
19:        end for
20:    end for
21: end for
```

Algorithm 3 Training algorithm of memory constrained selective AnDE.

```
1: Perform the first pass of learning to compute the joint probability distribution
2: Calculate the number of selected attributes $s$ according to equation (5)
3: Rank the attributes and then select the top $s$ attributes as set $\mathcal{S}$
4: Perform the second pass of learning to compute the probability distribution as in Algorithm
   2
```

4.4 Complexity Analysis

Section 4.1 gives an analysis of the space complexity of selective AnDE, which is $\mathcal{O}(k\bar{u}^n(C(s,n+1)\bar{u} + C(s,n) * (a-s) * \bar{r}))$. It is worthwhile to note that the memory requirement in the first pass of learning can be ignored compared to the memory in the second pass.

In order to compile the probability tables, for every instance, we need to update each entry for every combination of the $n+1$ attribute-values from s attributes and the entry for the unselected $a-s$ attributes given every combination of the n attribute-values from s attributes, so the time complexity of the training procedure is $\mathcal{O}(t(C(s,n+1)+C(s,n)(a-s)))$. When classifying a single example, we need to consider each attribute for every qualified combination of s parent attributes within each class, so the time complexity is $\mathcal{O}(kaC(s,n))$.

5 Empirical Study

In this section, we first describe the empirical setup. Then, in Section 5.2 we evaluate the selective AnDE with different attribute ranking approaches, as well as weighting and subsumption resolution. Section 5.3 compares selective AnDE with NB and AnDE in terms of RMSE and zero-one loss, analyses the bias and variance component of the error results, and presents training and classification time comparisons for different algorithms considered.

5.1 Empirical Setup

As the algorithm is proposed for large data, we undertake an extensive online search to gather a group of large datasets, all of which have more than 100K instances. These are all the publicly available datasets we could find. The detailed sources of all data sets have been indicated in Table 1. From left to right, we present the following characteristics of each data set: name, number of instances, number of attributes, number of classes, source and description. Note that the data sets have been ranked in ascending order of number of instances.

All datasets except `poker-hand`, `uscensus1990` and `splice` contain one or more numeric attributes. 6 datasets contain only numeric attributes: `MITFaceSetA`, `MITFaceSetB`, `MITFaceSetC`, `USPSExtended`, `MSDYearPrediction` and `satellite`. We discretize these numeric attributes using 5-bin equal frequency discretization (EF5). We have observed that EF5 and MDL [7] discretization provide the best results in approximately half of the datasets each. In fact, the discretization method does not matter if the group of data sets is large enough [8]. EF5 has been chosen because it is faster than MDL, and also because it is not supervised and hence does not need to potentially provide the classifier with class information from the holdout data when used for pre-discretization. Using a pre-fixed number of bins gives us another advantage of not having to deal with a huge number of values per attribute as in MDL discretization in some cases. When we discretize one attribute using EF5, we need to store the values of this attribute and sort the values. Then we compute the cut points, which can be used to discretize new instance, either training instance or testing instance. To avoid loading the whole data into memory, only a sample of 100K points is used to define the bins for discretization.

We run the experiments on a C++ system which is specially designed for out-of-core learning. It has the following characteristics:

1) It supports out-of-core learning, which means it can fetch one instance at a time from the disk. This addresses the problem that large data sets can not be loaded into memory entirely.

2) It provides the ability to flexibly set the number of learning passes through the training data.

3) It supports 10-fold cross validation and bias-variance decomposition.

The base probabilities are estimated using m-estimation ($m = 1$) [3]. Missing values have been considered as a distinct value. Note that root mean square error is calculated exclusively on the true class label. This is different from Weka's implementation [10], where all class labels are considered.

Table 1 Data sets used for experiments[1]

No.	Name	♯Inst	♯Att	♯Class	Source	Description
1	localization	164860	5	11	UCI [13]	Recordings of 5 people performing different activities. Each person wore 4 sensors while performing the same scenario 5 times.
2	census-income	299285	41	2	UCI [1]	Weighted census data extracted from the 1994 and 1995 current population surveys conducted by the U.S. Census Bureau.
3	USPS-Extended	341462	676	2	CVM [21]	0/1 digit classification (extended version of the USPS data set).
4	MITFace-SetA	474101	361	2	CVM [21]	Face detection using an extended version of the MIT face database[c]. By adding nonfaces to the original training set.
5	MITFace-SetB	489410	361	2	CVM [21]	Each training face is blurred and added to set A. They are then flipped laterally.
6	MSDYear-Prediction	515345	90	90	UCI [1]	Prediction of the release year of a song from audio features. Songs are mostly western, commercial tracks ranging from 1922 to 2011, with a peak in the year 2000s.
7	covertype	581012	54	7	UCI [1]	Predicting forest cover type from cartographic attributes only (no remotely sensed data).
8	MITFace-SetC	839330	361	2	CVM [21]	Each face in set B is rotated.
9	poker-hand	1025010	10	10	UCI [1]	Each record is an example of a hand consisting of five playing cards drawn from a standard deck of 52. Each card is described using two attributes (suit and rank), for a total of 10 predictive attributes. The class label describes the “Poker Hand”. The order of cards is important.
10	uscensus-1990	2458285	67	4	UCI [1]	Discretized version of the USCensus1990raw dataset, a 1% sample from the full 1990 census. ‘Temp. Absence From Work’ has been selected as class.
11	PAMAP2	3850505	54	19	UCI [17]	Data of 18 different physical activities (such as walking, cycling, playing soccer, etc., the 19th label is transient activities), performed by 9 subjects wearing 3 inertial measurement units and a heart rate monitor.
12	kddcup	5209460	41	40	UCI [1]	Contains a standard set of data to be audited, which includes a wide variety of intrusions simulated in a military network environment: “bad” connections, called intrusions or attacks, and “good” normal connections.
13	linkage	5749132	11	2	[11]	Element-wise comparison of records with personal data from a record linkage setting. The task is to decide from a comparison pattern whether the underlying records belong to one person.
14	satellite	8705159	138	24	[16]	Satellite image time series to predict land cover.
15	splice	54627840	141	2	[20]	Recognising a human acceptor splice site (largest public data for which subsampling is not an effective learning strategy).

[1] The data sets are ranked in ascending order of number of instances and the appendix gives the results for individual datasets for those who wish to consider the effects of different factors on the outcomes.

5.2 Best Configuration of Selective AnDE

In this subsection, we first compare an approximation to Eq. 5. Then we compare the performance of three different ranking approaches along with random ranking in selective A2DE. We also evaluate the influence of weighting and subsumption resolution in selective A2DE. The aim is to obtain the best configuration for selective AnDE.

Zero-one loss and root mean squared error (RMSE) are the most common loss functions to measure the classification performance. As RMSE gives a finer grained measure of the calibration of the probability estimates than zero-one loss, we select the best configuration in terms of RMSE in this section. Table 8 in Appendix A presents the RMSE for each algorithm on all data sets. These results are obtained by 10-fold cross validation.

In order to give the results a more intuitionistic explanation, we present summaries of win/draw/loss records of alternative algorithms, which indicate the number of data sets on which one algorithm has lower, equal or higher outcome relative to the other. Each entry compares the algorithm in the row against the algorithm in the column. The p value following each win/draw/loss record is the outcome of a binomial sign test and represents the probability of observing the given number of wins and losses if each were equally likely. The reported p value is the result of a two-tailed test. We consider a difference to be significant if $p \leq 0.05$. All such p values have been changed to boldface in the table.

5.2.1 Comparison of two approaches to compute the number of selected attributes

It is a bit complex to compute the number of selected attributes using Eq. 5. So we here propose a heuristic to compute the selected attributes approximately. We use $\bar{u}$ also for the $a - s$ unselected attributes as we believe this will not make a significant difference and it will simplify the computation. So the overall memory complexity of selected AnDE is $\mathcal{O}(k(C(s, n+1) + C(s, n) * (a - s))\bar{u}^{n+1})$. This gives the heuristic to compute the number s of selected attributes as follows,

$$k(C(s, n+1) + C(s, n) * (a - s))\bar{u}^{n+1} = kC(a, n' + 1)\bar{v}^{n'+1}. \tag{8}$$

We compare these two approaches while using both mutual information and conditional mutual information to direct rank the attributes. The algorithm using Eq. 5 is abbreviated as SA2DE$_{\text{dRank\_acrt}}$, while the algorithm using Eq.8 is abbreviated as SA2DE$_{\text{dRank}}$. Table 2 provides the win/draw/loss record of SA2DE$_{\text{dRank\_acrt}}$ against SA2DE$_{\text{dRank}}$. We can see that the approximation in Eq. 8 results in performance loss compared to the accurate estimation in Eq. 5. But it is still acceptable. We use Eq. 8 for the rest of the experiments.

Table 2 W/D/L of SA2DE with different approaches to compute the number of selected attributes

	SA2DE$_{\text{dRank}}$	
	W/D/L	p
SA2DE$_{\text{dRank\_acrt}}$	4/10/1	0.375

5.2.2 Comparison of selective A2DE with different attribute ranking approaches

As discussed in Section 4.2, we use mutual information, conditional mutual information, and both of them to rank the attributes respectively. In order to evaluate the effectiveness of ranking, we also run the algorithm which randomly ranks the attributes. These four algorithms are abbreviated as SA2DE$_{\text{MI}}$, SA2DE$_{\text{CMI}}$, SA2DE$_{\text{dRank}}$ and SA2DE$_{\text{Random}}$.

Table 3 presents the win/draw/loss records of these four algorithms. Note that each win/draw/loss entry indicates the result of the row algorithm versus the column algorithm. It is the same for the tables in the rest of the paper. We can see that SA2DE$_{\text{dRank}}$ obtains lower RMSE more often than SA2DE$_{\text{MI}}$, SA2DE$_{\text{CMI}}$ and SA2DE$_{\text{Random}}$, significantly so with respect to SA2DE$_{\text{MI}}$. Note that while SA2DE$_{\text{MI}}$ and SA2DE$_{\text{CMI}}$ also obtain lower RMSE more often than

Table 3 W/D/L of SA2DE with different ranking methods

	$SA2DE_{MI}$		$SA2DE_{CMI}$		$SA2DE_{dRank}$	
	W/D/L	p	W/D/L	p	W/D/L	p
$SA2DE_{CMI}$	4/8/3	1				
$SA2DE_{dRank}$	10/3/2	**0.039**	9/4/2	0.065		
$SA2DE_{Random}$	6/0/9	0.607	6/0/9	0.607	4/0/11	0.118

$SA2DE_{Random}$, these differences are not found to be significant. Relative to $SA2DE_{CMI}$, $SA2DE_{MI}$ achieves lower RMSE almost as often as higher. These results show that dRank is the most favorable attribute ranking approach. This might be explained by the fact that dRank considers not only the correlation between the attribute and the class label, but also the correlation among the selected attributes.

For the rest of the experiments, we consider dRank as the attribute ranking option.

5.2.3 Comparison of weighting and subsumption resolution in selective A2DE

As weighting and subsumption resolution are two most well-known improvements to AODE, we also examine the influence of these techniques in selective A2DE, which are denoted by w and sub in the name.

Table 4 W/D/L of SA2DE with weighting and subsumption resolution

	$SA2DE_{dRank}$		$SA2DE_{dRank\_w}$		$SA2DE_{dRank\_sub}$	
	W/D/L	p	W/D/L	p	W/D/L	p
$SA2DE_{dRank\_w}$	6/7/2	0.289				
$SA2DE_{dRank\_sub}$	4/8/3	1	5/4/6	1		
$SA2DE_{dRank\_w\_sub}$	7/4/4	0.549	4/8/3	1	7/5/3	0.344

From the win/draw/loss records in Table 4, we can see that $SA2DE_{dRank\_w}$ and $SA2DE_{dRank\_w\_sub}$ achieve lower RMSE more often than $SA2DE_{dRank}$, but these differences are not significant. While $SA2DE_{dRank\_sub}$ achieves lower RMSE almost as often as higher than $SA2DE_{dRank}$.

We may conclude that subsumption resolution will not improve selective A2DE while weighting will. The reason might be that subsumption resolution is also a technique to perform attribute selection and the repeated attribute selection will not improve the accuracy further. As the improvement of weighting to selective A2DE is not significant, we will not exploit these two techniques in the following experiments.

5.3 Selective AnDE Compared to AnDE and ASAODE

In order to present a comprehensive comparison of selective AnDE, we also implement two selective A3DE algorithms in this section. $SA3DE_{dRank\_one}$ performs attributes selection based on the memory requirement of AODE, while $SA3DE_{dRank\_two}$ on that of A2DE. We compare these algorithms with NB, AODE and A2DE.

At the same time, we want to obtain a thorough idea on how selective AnDE is compared with other improvements of AODE. As the study in [4] shows that

ASAODE outperforms such improvements of AODE as weightily AODE, AODE with Subsumption Resolution and AODE with BSE, we add ASADOE here to give a thorough comparison.

5.3.1 Comparison in terms of RMSE

The win/draw/loss results of involved algorithms in terms of RMSE are presented in Table 5. Notably, we obtain the results of A2DE on only 13 data sets. So the sum of win/draw/loss records of A2DE with respect to alternative algorithm is 13. Similarly, the sum of win/draw/loss records for SA3DE$_{\mathrm{dRank\_one}}$ and SA3DE$_{\mathrm{dRank\_two}}$ is 14.

Table 5 W/D/L of NB, AODE, ASAODE, SA2DE, SA3DE and A2DE in terms of RMSE

		NB size		AODE size								A2DE size	
		NB		AODE		ASAODE		SA2DE$_{\mathrm{dRank}}$		SA3DE$_{\mathrm{dRank\_one}}$		A2DE	
		W/D/L	p	W/D/L	p	W/D/L	p	W/D/L	p	W/D/L	p	W/D/L	p
AODE size	AODE	12/0/3	**0.035**										
	ASAODE	15/0/0	**<0.001**	14/1/0	**<0.001**								
	SA2DEdRank	14/0/1	**<0.001**	13/0/2	**0.007**	12/0/3	**0.035**						
	SA3DE$_{\mathrm{dRank\_one}}$	12/0/2	**0.013**	12/0/2	**0.013**	10/0/4	0.18	10/0/4	0.18				
A2DE size	A2DE	12/0/1	**0.003**	13/0/0	**<0.001**	6/0/7	1	6/0/7	1	4/0/9	0.267		
	SA3DE$_{\mathrm{dRank\_two}}$	14/0/0	**<0.001**	14/0/0	**<0.001**	12/0/2	**0.013**	14/0/0	**<0.001**	11/3/0	**<0.001**	12/0/1	**0.003**

These algorithms can be divided into three categories. The first category contains only NB. The second category uses models that are of AODE size, including AODE, ASAODE, SA2DE$_{\mathrm{dRank}}$ and SA3DE$_{\mathrm{dRank\_one}}$. The last category uses models that are of A2DE size, including A2DE and SA3DE$_{\mathrm{dRank\_two}}$.

We can see that NB achieves higher RMSE significantly more often than all the other algorithms, which demonstrates the limitations of the independence assumption in NB. We observe that both SA2DE$_{\mathrm{dRank}}$ and SA3DE$_{\mathrm{dRank\_one}}$ reduce RMSE significantly often relative to AODE. They also reduce RMSE relative to ASAODE, although the difference between SA3DE$_{\mathrm{dRank\_one}}$ and ASADOE is not significant. As for algorithms of A2DE size, SA3DE$_{\mathrm{dRank\_two}}$ also reduces RMSE significantly often relative to A2DE.

We can see that selective AnDE can obtain better performance than AnDE of the same size but with lower n. Not only that, SA2DE$_{\mathrm{dRank}}$ achieves lower RMSE almost as often as higher than A2DE. SA3DE$_{\mathrm{dRank\_one}}$ obtains lower RMSE more often than A2DE, although the difference is not significant. Notably, SA3DE$_{\mathrm{dRank\_one}}$ and SA2DE$_{\mathrm{dRank}}$ are of AODE size, which require much less memory than A2DE. These results show that selective AnDE can achieve similar performance as AnDE with the same n.

These observations demonstrate that the attribute selection approach proposed in this paper is a powerful technique to improve classification accuracy while not requiring more memory. It might be explained by the fact that AnDE with higher n has lower bias and big data sets favor these low bias algorithm, although the attribute selection might result in some loss in accuracy.

5.3.2 Comparison in terms of zero-one loss

In this section we assess the performance using zero-one loss. Table 9 in Appendix B presents the zero-one losses for each algorithm on all data sets, which are obtained along with the RMSE results.

Table 6 W/D/L of NB, AODE, ASAODE, SA2DE, SA3DE and A2DE in terms of zero-one loss

		NB size		AODE size								A2DE size	
		NB		AODE		ASAODE		SA2DE$_{\mathrm{dRank}}$		SA3DE$_{\mathrm{dRank\_one}}$		A2DE	
		W/D/L	p	W/D/L	p	W/D/L	p	W/D/L	p	W/D/L	p	W/D/L	p
AODE size	AODE	11/1/3	0.057										
	ASAODE	14/1/0	**<0.001**	11/2/2	**0.022**								
	SA2DEdRank	14/0/1	**<0.001**	13/0/2	**0.007**	11/1/3	0.057						
	SA3DE$_{\mathrm{dRank\_one}}$	12/0/2	**0.013**	11/0/3	0.057	9/0/5	0.424	9/1/4	0.267				
A2DE size	A2DE	11/1/1	**0.006**	12/1/0	**<0.001**	6/1/6	1	7/0/6	1	3/0/10	0.092		
	SA3DE$_{\mathrm{dRank\_two}}$	14/0/0	**<0.001**	14/0/0	**<0.001**	12/0/2	**0.013**	14/0/0	**<0.001**	11/3/0	**<0.001**	13/0/0	**<0.001**

Table 6 presents the win/draw/loss records of involved algorithms in terms of zero-one loss. What is different from Table 5 is that the difference between AODE and NB and the difference between SA3DE$_{\mathrm{dRank\_one}}$ and AODE are not significant. But AODE and SA3DE$_{\mathrm{dRank\_one}}$ still achieve lower zero-one losses more often than their rivals in these comparisons. What is revealed in Table 6 is very similar with that in Table 5.

5.3.3 Comparison in terms of bias and variance

As we expect selective AnDE to exhibit low bias, we run the bias-variance decomposition experiment which utilizes the experimental method proposed by Kohavi and Wolpert [14]. For each data set, 10,000 training examples and 10,000 testing examples are randomly selected. The bias variance decomposition is calculated from the error on the test examples. This process is repeated 10 times to obtain the mean bias and variance.

Table 10 in Appendix C provides the detailed results for each combination of metric, algorithm and data set. Note that we get the bias and variance decomposition for A2DE on only 14 data sets. Table 7 presents the win/draw/loss records of the above 7 algorithms with respect to bias and variance.

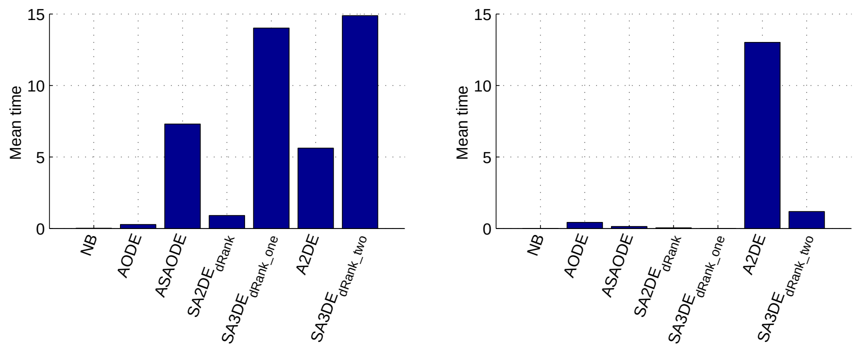
We may observe that NB achieves higher bias significantly more often than all the other algorithms. SA2DE$_{\mathrm{dRank}}$ and SA3DE$_{\mathrm{dRank\_one}}$ obtain lower bias more often than AODE, although the difference between SA3DE$_{\mathrm{dRank\_one}}$ and AODE is not significant. Relative to ASAODE, SA2DE$_{\mathrm{dRank}}$ obtains lower bias more often, but the difference is not significant. SA3DE$_{\mathrm{dRank\_one}}$ obtains lower bias almost as often as ASAODE. SA3DE$_{\mathrm{dRank\_two}}$ obtains lower bias more often than AODE, ASAODE, SA2DE$_{\mathrm{dRank}}$, SA3DE$_{\mathrm{dRank\_one}}$ and A2DE, not significant so only with respect to ASAODE and A2DE. The win/draw/loss records for variance do not indicate a significant difference between any pair of these algorithms except SA2DE$_{\mathrm{dRank}}$ against NB and SA3DE$_{\mathrm{dRank\_one}}$ against NB.

Table 7 W/D/L of NB, AODE, ASAODE, SA2DE, SA3DE and A2DE in terms of bias and variance

			NB size		AODE size								A2DE size	
			NB		AODE		ASAODE		SA2DE$_{\mathrm{dRank}}$		SA3DE$_{\mathrm{dRank\_one}}$		A2DE	
			W/D/L	p	W/D/L	p	W/D/L	p	W/D/L	p	W/D/L	p	W/D/L	p
Bias	AODE size	AODE	14/1/0	**<0.001**										
		ASAODE	14/1/0	**<0.001**	11/2/2	**0.022**								
		SA2DE$_{\mathrm{dRank}}$	14/1/0	**<0.001**	11/2/2	**0.022**	9/2/4	0.267						
		SA3DE$_{\mathrm{dRank\_one}}$	12/1/2	**0.013**	11/1/3	0.057	7/1/7	1	9/1/5	0.424				
	A2DE size	A2DE	13/1/0	**<0.001**	12/2/0	**<0.001**	9/2/3	0.146	8/2/4	0.388	7/2/5	0.774		
		SA3DE$_{\mathrm{dRank\_two}}$	14/1/0	**<0.001**	14/1/0	**<0.001**	10/1/4	0.18	11/2/2	**0.022**	10/5/0	**0.002**	9/2/3	0.146
Variance	AODE size	AODE	6/2/7	1										
		ASAODE	7/1/7	1	6/1/8	0.791								
		SA2DE$_{\mathrm{dRank}}$	2/0/13	**0.007**	5/0/10	0.302	4/0/11	0.118						
		SA3DE$_{\mathrm{dRank\_one}}$	2/0/13	**0.007**	4/0/11	0.118	4/0/11	0.118	4/3/8	0.388				
	A2DE size	A2DE	5/1/8	0.581	6/1/7	1	6/1/7	1	9/1/4	0.267	10/0/4	0.18		
		SA3DE$_{\mathrm{dRank\_two}}$	5/1/9	0.424	4/1/10	0.18	6/1/8	0.791	10/1/4	0.18	9/4/2	0.065	4/1/9	0.267

5.3.4 Comparison in terms of computing time

Table 11 in Appendix D presents the training and classification time obtained by 10-fold cross validation. The mean training and classification time across all data sets for each algorithm is shown in Fig. 1.



(a) Training time (b) Classification time

Fig. 1 Computation time comparison of different algorithms (hours).

We can see that NB needs less training and classification time than all the other algorithms. The reason is that the independence assumption in NB simplifies the model computation. SA2DE$_{\mathrm{dRank}}$ and SA3DE$_{\mathrm{dRank\_one}}$ enjoy consistent advantages over AODE and ASAODE at classification time, while suffering the training time disadvantages over only AODE. It is mainly because SA2DE$_{\mathrm{dRank}}$, SA3DE$_{\mathrm{dRank\_one}}$ and ASAODE require one more pass learning through the data than AODE and the former two compile more complicated tables of instance frequencies at training time, while require less attributes at classification time. It is also true for SA3DE$_{\mathrm{dRank\_two}}$ to A2DE.

6 Conclusion

In order to deal with large data learning, we present memory constrained selective AnDE in this paper which can achieve a satisfying balance between the memory requirement and prediction accuracy. Experimental results show that our novel heuristics provide highly accurate out-of-core learning for large datasets.

In all, selective AnDE enjoys the following characteristics:

1) two passes learning through the training data, which is acceptable to large data learning given the accuracy improvement;

2) comparable accuracy to regular AnDE without attribute selection;

3) the same memory requirement as An'DE, where $n' = n - 1$;

4) considerably lower bias and prediction error relative to An'DE, where $n' = n - 1$;

5) more training time, but less testing time than An'DE, where $n' = n - 1$.

During the process of doing this work, some new ideas have come into our mind. On one hand, it is worthwhile to explore the technique of selecting both parents and children. On the other hand, we can also combine the fast attribute selection technique based on leave one out cross validation proposed in [4] to further improve the classification accuracy.

Acknowledgement

This research has been supported by the Australian Research Council under grant DP140100087, Asian Office of Aerospace Research and Development, Air Force Office of Scientific Research under contract FA23861214030, National Natural Science Foundation of China under grant 61202135, 61272209, Natural Science Foundation of Jiangsu, China under grant BK20130735, Natural Science Foundation of Jiangsu Higher Education Institutions of China under grant 14KJB520019, 13KJB520011, 13KJB520013, the open project program of Key Laboratory of Symbolic Computation and Knowledge Engineering of Ministry of Education, Jilin University, Priority Academic Program Development of Jiangsu Higher Education Institutions.

This research has also been supported in part by the Monash e-Research Center and eSolutions-Research Support Services through the use of the Monash Campus HPC Cluster and the LIEF grant. This research was also undertaken on the NCI National Facility in Canberra, Australia, which is supported by the Australian Commonwealth Government.

References

1. Bache, K., Lichman, M.: UCI machine learning repository (2013). URL `http://archive.ics.uci.edu/ml`
2. Brain, D., Webb, G.I.: The need for low bias algorithms in classification learning from large data sets. In: Principles of Data Mining and Knowledge Discovery, pp. 62–73. Springer (2002)
3. Cestnik, B.: Estimating probabilities: a crucial task in machine learning. In: ECAI, vol. 90, pp. 147–149 (1990)
4. Chen, S., Martinez, A.M., Webb, G.I.: Highly scalable attribute selection for averaged one-dependence estimators. In: Proceedings of the 18th Pacific-Asia Conference on Knowledge Discovery and Data Mining, pp. 86–97. Springer (2014)
5. Domingos, P., Pazzani, M.: Beyond independence: Conditions for the optimality of the simple Bayesian classifier. In: Proc. 13th Intl. Conf. Machine Learning, pp. 105–112 (1996)
6. Duda, R.O., Hart, P.E.: Pattern Classification and Scene Analysis, 1 edn. John Wiley & Sons Inc (1973)
7. Fayyad, U.M., Irani, K.B.: Multi-interval discretization of continuous-valued attributes for classification learning. In: IJCAI, pp. 1022–1027 (1993)
8. Flores, M., Gámez, J., Martínez, A., Puerta, J.: Handling numeric attributes when comparing bayesian network classifiers: does the discretization method matter? Applied Intelligence **34**(3), 372–385 (2011)
9. Friedman, N., Geiger, D., Goldszmidt, M.: Bayesian network classifiers. Machine learning **29**(2-3), 131–163 (1997)
10. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The weka data mining software: an update. SIGKDD Explor. Newsl. **11**(1), 10–18 (2009). DOI 10.1145/1656274.1656278
11. I, S., G, H., M, S., A., G.A.: Evaluation des krebsregisters nrw - schwerpunkt record linkage - abschlussbericht. Tech. rep., Institut für medizinische Biometrie, Epidemiologie und Informatik, Universitätsmedizin Mainz (2009)

12. Jiang, L., Zhang, H.: Weightily averaged one-dependence estimators. In: PRICAI 2006: trends in artificial intelligence, pp. 970–974. Springer (2006)
13. Kaluža, B., Mirchevska, V., Dovgan, E., Luštrek, M., Gams, M.: An agent-based approach to care in independent living. In: Proceedings of the First international joint conference on Ambient intelligence, AmI'10, pp. 177–186. Springer-Verlag, Berlin, Heidelberg (2010)
14. Kohavi, R., Wolpert, D.H.: Bias plus variance decomposition for zero-one loss functions. In: Proceedings of the Thirteenth International Conference on Machine Learning, pp. 275–283. Morgan Kaufman Publishers, Inc. (1996)
15. MacKay, D.J.: Information theory, inference and learning algorithms. Cambridge university press (2003)
16. Petitjean, F., Inglada, J., Gançarski, P.: Satellite Image Time Series Analysis under Time Warping. IEEE Transactions on Geoscience and Remote Sensing **50**(8), 3081–3095 (2012)
17. Reiss, A., Stricker, D.: Creating and benchmarking a new dataset for physical activity monitoring. In: Proceedings of the 5th International Conference on PErvasive Technologies Related to Assistive Environments, PETRA '12, pp. 40:1–40:8. ACM, New York, NY, USA (2012)
18. Rish, I.: An empirical study of the naive bayes classifier. In: IJCAI 2001 workshop on empirical methods in artificial intelligence, vol. 3, pp. 41–46 (2001)
19. Sahami, M.: Learning limited dependence Bayesian classifiers. In: Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, pp. 335–338 (1996)
20. Sonnenburg, S., Franc, V.: COFFIN : A computational framework for linear SVMs. In: Proc. ICML 2010 (2010)
21. Tsang, I.W., Kwok, J.T., Cheung, P.M.: Core vector machines: Fast SVM training on very large data sets. J. Mach. Learn. Res. **6**, 363–392 (2005)
22. Webb, G.I., Boughton, J.R., Wang, Z.: Not so naive bayes: aggregating one-dependence estimators. Machine Learning **58**(1), 5–24 (2005)
23. Webb, G.I., Boughton, J.R., Zheng, F., Ting, K.M., Salem, H.: Learning by extrapolation from marginal to full-multivariate probability distributions: decreasingly naive Bayesian classification. Machine learning **86**(2), 233–272 (2012)
24. Yang, Y., Korb, K., Ting, K.M., Webb, G.I.: Ensemble selection for superparent-one-dependence estimators. In: AI 2005: Advances in Artificial Intelligence, pp. 102–112. Springer (2005)
25. Yang, Y., Webb, G.I., Cerquides, J., Korb, K.B., Boughton, J., Ting, K.M.: To select or to weigh: a comparative study of linear combination schemes for superparent-one-dependence estimators. Knowledge and Data Engineering, IEEE Transactions on **19**(12), 1652–1665 (2007)
26. Yu, L., Liu, H.: Feature selection for high-dimensional data: A fast correlation-based filter solution. In: Proceedings of the Twentieth International Conference on Machine Learning, vol. 3, pp. 856–863 (2003)
27. Zaidi, N.A., Webb, G.I.: Fast and effective single pass Bayesian learning. In: Advances in Knowledge Discovery and Data Mining, pp. 149–160. Springer (2013)
28. Zheng, F., Webb, G.I.: Finding the right family: parent and child selection for averaged one-dependence estimators. In: Machine Learning: ECML 2007, pp. 490–501. Springer (2007)
29. Zheng, F., Webb, G.I., Suraweera, P., Zhu, L.: Subsumption resolution: an efficient and effective technique for semi-naive Bayesian learning. Machine learning **87**(1), 93–125 (2012)
30. Zheng, Z., Webb, G.I.: Lazy learning of Bayesian rules. Machine Learning **41**(1), 53–84 (2000)

Table 8 RMSE of involved algorithms on 15 large data sets

Algo	localization	census-income	USPS-Extended	MITFace-SetA	MITFace-SetB	MSDYear-Prediction	cover-type	MITFace-SetC	poker-hand	uscensus-1990	PAMAP2	kddcup	linkage	satellite	splice
$SA2DE_{MI}$	0.5939± 0.0018	0.2506± 0.0023	0.1625± 0.0021	0.0494± 0.0027	**0.0971± 0.0024**	0.9427± 0.0003	0.4616± 0.0017	0.1506± 0.0018	0.4095± 0.0011	**0.1912± 0.0009**	0.3678± 0.0006	0.1195± 0.0009	0.0096± 0.0009	0.5951± 0.0003	0.1438± 0.0010
$SA2DE_{CMI}$	0.5939± 0.0018	0.2506± 0.0023	0.1632± 0.0024	0.0495± 0.0027	0.0982± 0.0023	0.9427± 0.0003	0.4616± 0.0017	0.1496± 0.0031	**0.4091± 0.0009**	**0.1912± 0.0009**	0.3159± 0.0006	0.1195± 0.0009	0.0096± 0.0009	0.5816± 0.0003	0.1438± 0.0010
$SA2DE_{dRank}$	0.5939± 0.0018	0.2506± 0.0023	0.1502± 0.0035	**0.0449± 0.0025**	0.1002± 0.0026	**0.9412± 0.0003**	0.4512± 0.0018	0.1165± 0.0017	**0.4091± 0.0012**	0.2429± 0.0005	0.3148± 0.0006	**0.0528± 0.0008**	**0.0083± 0.0004**	**0.5631± 0.0004**	0.1438± 0.0010
$SA2DE_{Random}$	**0.5873± 0.0073**	0.3448± 0.0588	**0.1359± 0.0068**	0.0546± 0.0065	0.1026± 0.0059	0.9445± 0.0019	0.4751± 0.0072	0.1243± 0.0054	0.5076± 0.0139	0.2228± 0.0307	0.3817± 0.0209	0.1136± 0.0337	0.0107± 0.0017	0.5666± 0.0076	**0.1011± 0.0127**
$SA2DE_{dRank\_w}$	0.5933± 0.0019	0.2506± 0.0022	0.1504± 0.0035	**0.0449± 0.0025**	0.1002± 0.0027	**0.9412± 0.0003**	**0.4503± 0.0018**	**0.1161± 0.0016**	**0.4091± 0.0012**	0.2424± 0.0006	0.3147± 0.0006	**0.0528± 0.0008**	0.0086± 0.0004	**0.5631± 0.0004**	0.1437± 0.0010
$SA2DE_{dRank\_sub}$	0.5939± 0.0018	0.2273± 0.0021	0.1398± 0.0091	**0.0449± 0.0025**	0.1002± 0.0026	**0.9412± 0.0003**	0.4520± 0.0017	0.1165± 0.0017	**0.4091± 0.0012**	0.2274± 0.0006	0.3133± 0.0006	0.0811± 0.0007	**0.0083± 0.0004**	0.5656± 0.0004	0.1438± 0.0010
$SA2DE_{dRank\_w\_sub}$	0.5933± 0.0019	**0.2272± 0.0021**	0.1399± 0.0091	**0.0449± 0.0025**	0.1002± 0.0027	**0.9412± 0.0003**	0.4524± 0.0017	**0.1161± 0.0016**	**0.4091± 0.0012**	0.2270± 0.0006	**0.3132± 0.0006**	0.0690± 0.0007	0.0086± 0.0005	0.5656± 0.0004	0.1437± 0.0010
NB	0.7106± 0.0007	0.4660± 0.0018	0.2256± 0.0026	0.0982± 0.0029	0.1394± 0.0014	0.9600± 0.0002	0.4953± 0.0012	0.2367± 0.0021	0.5801± 0.0006	0.2911± 0.0006	0.4647± 0.0006	0.1849± 0.0008	0.0125± 0.0006	0.6540± 0.0002	0.0971± 0.0002
AODE	0.6520± 0.0010	0.2932± 0.0020	0.1538± 0.0028	0.1001± 0.0036	0.1682± 0.0025	0.9459± 0.0002	0.4587± 0.0013	0.1564± 0.0014	0.5392± 0.0006	0.2154± 0.0010	0.3881± 0.0008	0.0979± 0.0007	0.0120± 0.0005	0.5783± 0.0003	0.1034± 0.0003
ASAODE	0.6444± 0.0011	**0.2057± 0.0013**	0.1508± 0.0027	0.0509± 0.0020	0.1005± 0.0019	0.9455± 0.0002	0.4582± 0.0014	0.1419± 0.0017	0.5004± 0.0010	**0.1538± 0.0008**	0.3819± 0.0006	0.0611± 0.0007	0.0110± 0.0005	0.5783± 0.0003	**0.0532± 0.0002**
$SA2DE_{dRank}$	0.5939± 0.0018	0.2506± 0.0023	0.1502± 0.0035	**0.0449± 0.0025**	**0.1002± 0.0026**	**0.9412± 0.0003**	0.4512± 0.0018	0.1165± 0.0017	0.4091± 0.0012	0.2429± 0.0005	0.3148± 0.0006	0.0528± 0.0008	0.0083± 0.0004	**0.5631± 0.0004**	0.1438± 0.0010
$SA2DE_{dRank\_acrt}$	0.5939± 0.0018	0.2333± 0.0023	0.1453± 0.0051	**0.0449± 0.0025**	**0.1002± 0.0026**	**0.9412± 0.0003**	0.4469± 0.0017	0.1165± 0.0017	0.4107± 0.0008	0.2429± 0.0005	0.3148± 0.0006	**0.0507± 0.0007**	0.0083± 0.0004	0.5631± 0.0004	0.1438± 0.0010
$SA3DE_{dRank\_one}$	**0.5485± 0.0020**	0.2434± 0.0023	**0.1344± 0.0028**	0.1467± 0.0032	0.1436± 0.0024	0.9384± 0.0003	**0.4357± 0.0015**	**0.1062± 0.0016**	**0.2953± 0.0012**	0.2647± 0.0010	**0.3048± 0.0007**	0.0512± 0.0007	**0.0063± 0.0006**	0.5733± 0.0003	oot[1]
A2DE	0.5865± 0.0018	**0.2403± 0.0026**	0.1485± 0.0028	0.0737± 0.0027	0.1479± 0.0020	0.9368± 0.0003	0.4373± 0.0014	0.1489± 0.0017	0.4956± 0.0007	0.1753± 0.0010	0.3307± 0.0004	0.0833± 0.0007	0.0112± 0.0005	oot[1]	oot[1]
$SA3DE_{dRank\_two}$	**0.5407± 0.0020**	0.2434± 0.0023	**0.1075± 0.0038**	**0.0283± 0.0018**	**0.0704± 0.0022**	**0.9261± 0.0006**	**0.4170± 0.0016**	**0.0832± 0.0012**	**0.2953± 0.0012**	**0.1747± 0.0007**	**0.2365± 0.0006**	**0.0512± 0.0007**	**0.0060± 0.0006**	**0.4981± 0.0005**	oot[1]

[1] Out of time when the wall time is set to 120 hours for each fold.
- Bold face marks the lowest RMSE in each category. The categories are separated by double lines.

A Table of RMSE

Table 9 Zero-one loss of involved algorithms on 15 large data sets

Algo	locali-zation	census-income	USPS-Extended	MITFace-SetA	MITFace-SetB	MSDYear-Prediction	cover-type	MITFace-SetC	poker-hand	uscensus-1990	PAMAP2	kddcup	linkage	satellite	splice
NB	0.5449± 0.0026	0.2410± 0.0017	0.0532± 0.0012	0.0100± 0.0006	0.0199± 0.0004	0.9514± 0.0005	0.3321± 0.0024	0.0582± 0.0010	0.4988± 0.0018	0.0896± 0.0003	0.2365± 0.0007	0.0361± 0.0005	0.0002± 0.0000	0.4425± 0.0002	0.0121± 0.0001
AODE	0.4333± 0.0027	0.1106± 0.0015	0.0244± 0.0008	0.0104± 0.0007	0.0294± 0.0008	0.9281± 0.0013	0.2859± 0.0016	0.0254± 0.0005	0.4812± 0.0028	0.0532± 0.0004	0.1654± 0.0007	0.0154± 0.0002	0.0002± 0.0000	0.3537± 0.0004	0.0134± 0.0001
ASAODE	0.4556± 0.0033	**0.0555± 0.0009**	0.0235± 0.0008	0.0030± 0.0002	0.0108± 0.0004	0.9286± 0.0009	0.2852± 0.0018	0.0211± 0.0005	0.3302± 0.0022	**0.0274± 0.0003**	0.1611± 0.0005	0.0040± 0.0001	0.0002± 0.0000	0.3537± 0.0004	**0.0029± 0.0000**
SA2DE$_{\text{dRank}}$	0.3680± 0.0046	0.0800± 0.0013	0.0235± 0.0010	**0.0021± 0.0002**	**0.0105± 0.0005**	0.9248± 0.0011	0.2731± 0.0026	0.0143± 0.0004	0.1966± 0.0018	0.0664± 0.0003	0.1082± 0.0004	0.0031± 0.0001	**0.0001± 0.0000**	**0.3418± 0.0005**	0.0242± 0.0003
SA2DE$_{\text{dRank\_acrt}}$	0.3680± 0.0046	0.0722± 0.0017	0.0219± 0.0015	**0.0021± 0.0002**	**0.0105± 0.0005**	0.9248± 0.0011	0.2663± 0.0024	0.0143± 0.0004	0.1194± 0.0013	0.0664± 0.0003	0.1082± 0.0004	**0.0029± 0.0001**	**0.0001± 0.0000**	**0.3418± 0.0005**	0.0242± 0.0003
SA3DE$_{\text{dRank\_one}}$	**0.3338± 0.0022**	0.0731± 0.0014	**0.0188± 0.0007**	0.0235± 0.0010	0.0223± 0.0008	**0.9201± 0.0008**	**0.2555± 0.0017**	**0.0120± 0.0003**	**0.0856± 0.0010**	0.0817± 0.0009	**0.1003± 0.0005**	0.0030± 0.0001	**0.0001± 0.0000**	0.3538± 0.0003	oot[1]
A2DE	0.3598± 0.0047	0.0777± 0.0014	0.0227± 0.0008	0.0057± 0.0004	0.0227± 0.0006	0.9095± 0.0012	0.2609± 0.0019	0.0232± 0.0005	0.1185± 0.0019	0.0366± 0.0005	0.1207± 0.0003	0.0108± 0.0002	0.0002± 0.0000	oot[1]	oot[1]
SA3DE$_{\text{dRank\_two}}$	**0.3210± 0.0030**	**0.0731± 0.0014**	**0.0120± 0.0008**	**0.0009± 0.0001**	**0.0052± 0.0003**	**0.8958± 0.0013**	**0.2337± 0.0023**	**0.0074± 0.0002**	**0.0856± 0.0010**	**0.0357± 0.0003**	**0.0639± 0.0003**	**0.0030± 0.0001**	**0.0000± 0.0000**	**0.2716± 0.0005**	oot[1]

[1] Out of time when the wall time is set to 120 hours for each fold.
- Bold face marks the lowest zero-one loss in each category. The categories are separated by double lines.

B Table of zero-one loss

Table 10 Bias and Variance decomposition of involved algorithms on 15 large data sets

	Algo	locali-zation	census-income	USPS-Extended	MITFace-SetA	MITFace-SetB	MSDYear-Prediction	cover-type	MITFace-SetC	poker-hand	uscensus-1990	PAMAP2	kddcup	linkage	satellite	splice
Bias	NB	0.4882	0.2130	0.0476	0.0042	0.0173	0.5468	0.2985	0.0548	0.3691	0.0723	0.2235	0.0377	0.0004	0.3934	0.0029
	AODE	0.3692	0.0668	0.0138	0.0032	0.0128	0.5271	0.2511	0.0224	0.2268	0.0348	0.1397	0.0088	0.0003	0.2600	0.0029
	ASAODE	0.3881	0.0533	0.0137	0.0014	0.0067	0.5243	0.2483	0.0171	0.1695	0.0249	0.1245	0.0044	0.0003	0.2603	0.0029
	SA2DE$_{dRank}$	0.2962	0.0754	0.0080	0.0017	0.0034	0.5238	0.2232	0.0093	0.1645	0.0363	0.0857	0.0049	0.0003	0.2225	0.0029
	SA3DE$_{dRank\_one}$	0.2575	0.0470	0.0065	0.0440	0.0496	0.5310	0.2042	0.0221	0.2177	0.0313	0.0796	0.0046	0.0002	0.2149	0.0029
	A2DE	0.2890	0.0472	oot[1]	0.0007	0.0058	0.5220	0.2301	0.0133	0.1718	0.0306	0.0849	0.0046	0.0003	0.1959	0.0029
	SA3DE$_{dRank\_two}$	0.2511	0.0470	0.0052	0.0017	0.0021	0.5240	0.1874	0.0044	0.2177	0.0280	0.0582	0.0046	0.0002	0.1894	0.0029
Variance	NB	0.0568	0.0185	0.0023	0.0014	0.0017	0.4086	0.0351	0.0019	0.1397	0.0083	0.0373	0.0041	0.0001	0.0508	0.0005
	AODE	0.0748	0.0318	0.0011	0.0008	0.0027	0.4122	0.0385	0.0019	0.1348	0.0081	0.0458	0.0028	0.0001	0.0812	0.0002
	ASAODE	0.0706	0.0074	0.0013	0.0010	0.0031	0.4144	0.0396	0.0027	0.1295	0.0032	0.0567	0.0018	0.0001	0.0811	0.0003
	SA2DE$_{dRank}$	0.0964	0.0563	0.0037	0.0016	0.0026	0.4087	0.0516	0.0033	0.2104	0.0221	0.0412	0.0021	0.0002	0.1049	0.0000
	SA3DE$_{dRank\_one}$	0.1453	0.0360	0.0039	0.0311	0.0250	0.4087	0.0621	0.0114	0.2494	0.0206	0.0385	0.0010	0.0002	0.1276	0.0000
	A2DE	0.0964	0.0253	oot[1]	0.0004	0.0015	0.4103	0.0435	0.0021	0.1536	0.0077	0.0502	0.0032	0.0001	0.0973	0.0001
	SA3DE$_{dRank\_two}$	0.1408	0.0360	0.0015	0.0012	0.0018	0.4047	0.0622	0.0024	0.2494	0.0130	0.0495	0.0010	0.0001	0.1044	0.0000

[1] Out of time when the wall time is set to 120 hours.

C Table of bias and variance

Table 11 Computing Time of involved algorithms on 15 large data sets

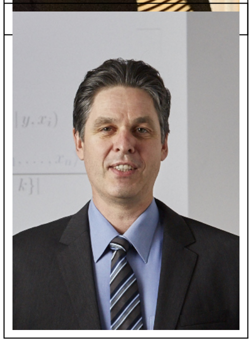
	Algo	locali-zation	census-income	USPS-Extended	MITFace-SetA	MITFace-SetB	MSDYear-Prediction	cover-type	MITFace-SetC	poker-hand	uscensus-1990	PAMAP2	kddcup	linkage	satellite	splice
Training time	NB	0	0	0.004	0.004	0.003	0.001	0.000	0.006	0.000	0.003	0.004	0.004	0.001	0.030	0.244
	AODE	0	0.001	0.488	0.290	0.228	0.016	0.001	0.367	0	0.009	0.013	0.012	0.002	0.928	1.904
	ASAODE	0	0.018	7.824	5.943	5.574	3.469	0.113	6.929	0.009	0.529	2.171	2.736	0.027	35.626	38.653
	$SA2DE_{dRank}$	0	0.001	2.974	0.918	0.265	0.139	0.004	0.162	0.001	0.024	0.029	0.021	0.003	3.975	5.191
	$SA3DE_{dRank\_one}$	0	0.205	64.293	17.577	20.069	1.288	0.328	30.478	0.003	0.747	6.959	0.353	0.007	53.832	oot[1]
	A2DE	0	0.021	27.304	11.186	10.544	0.636	0.043	20.469	0.001	1.917	0.708	0.225	0.003	oot[1]	oot[1]
	$SA3DE_{dRank\_two}$	0	0.021	62.651	20.479	20.495	1.361	0.041	36.427	0.001	1.257	0.799	0.334	0.004	64.493	oot[1]
Testing time	NB	0	0	0.002	0.001	0.001	0.002	0.000	0.003	0	0.001	0.004	0.005	0.001	0.025	0.085
	AODE	0	0.001	0.280	0.151	0.134	0.212	0.007	0.230	0.001	0.031	0.122	0.158	0.002	2.498	2.577
	ASAODE	0	0	0.164	0.001	0.001	0.037	0.004	0.033	0	0.001	0.034	0.008	0.001	1.808	0.034
	$SA2DE_{dRank}$	0	0	0.037	0.019	0.017	0.041	0.001	0.018	0	0.006	0.017	0.007	0.001	0.405	0.228
	$SA3DE_{dRank\_one}$	0	0	0.003	0.001	0.001	0.013	0.001	0.003	0	0	0.004	0.007	0.001	0.056	oot[1]
	A2DE	0	0.021	57.866	13.658	14.162	36.629	0.271	24.365	0.005	1.721	8.186	12.301	0.008	oot[1]	oot[1]
	$SA3DE_{dRank\_two}$	0	0	1.798	0.821	0.873	0.951	0.004	1.522	0	0.018	0.208	0.004	0.001	10.393	oot[1]

[1] Out of time when the wall time is set to 120 hours for each fold.

D Table of computing time

Shenglei Chen received the Ph.D. degree in computer science from Nanjing University of Science and Technology in 2007. He is currently an associate professor in the Department of E-Commerce, Nanjing Audit University, China. He also serves as an adjunct research fellow in the Faculty of Information Technology, Monash University, Australia. His research interests include data mining and machine learning. He has published innovative works in journals and conference proceedings such as Journal of Machine Learning Research, Knowledge-Based Systems and PAKDD.

Ana M. Martínez received her M.S. and Ph.D. degrees in Computer Science from the University of Castilla-La Mancha (Spain) in 2007 and 2012 respectively. She was afterwards working as a research fellow at Monash University (Australia), in a project to classify large amounts of data with semi-naive Bayesian classifiers. As for July 2014, she is a postdoc in Aalborg University (Denmark), as part of the AMIDST (Analysis of MassIve Data STreams) European project (http://amidst.eu).



Geoffrey I. Webb is Director of the Monash University Center for Data Science. He was editor in chief of Data Mining and Knowledge Discovery from 2005 to 2014. He has been Program Committee Chair of both ACM SIGKDD and IEEE ICDM, as well as General Chair of ICDM. He is a Technical Advisor to BigML Inc, who incorporate his best of class association discovery software, Magnum Opus, into their cloud based Machine Learning service. He developed many of the key mechanisms of support-confidence association discovery in the 1980s. His OPUS search algorithm remains the state-of-the-art in rule search. He pioneered multiple research areas as diverse as black-box user modelling, interactive data analytics and statistically-sound pattern discovery. He has developed many useful machine learning algorithms that are widely deployed. He received the 2013 IEEE Outstanding Service Award, a 2014 Australian Research Council Discovery Outstanding Researcher Award and is an IEEE Fellow.

Limin Wang received the PhD degree in computer science from JiLin University in 2005. He is currently a professor in the college of computer science and technology in JiLin University, China. His research interests include probabilistic logic inference and Bayesian network. He has published innovative papers in journals such as Knowledge-Based Systems, Expert System with Applications and Progress in Natural Science.