Knowledge-Based Systems 192 (2020) 105361

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

A novel selective naïve Bayes algorithm[☆]

Shenglei Chen[a,*], Geoffrey I. Webb[b], Linyuan Liu[a], Xin Ma[c]

[a] Department of E-Commerce, Nanjing Audit University, Nanjing, China
[b] Faculty of Information Technology, Monash University, VIC 3800, Australia
[c] School of Statistics and Mathematics, Nanjing Audit University, Nanjing, China

ARTICLE INFO

Article history:
Received 3 July 2019
Received in revised form 22 October 2019
Accepted 7 December 2019
Available online 16 December 2019

Keywords:
Naïve Bayes
Attribute selection
Leave-one-out cross validation
Model selection

ABSTRACT

Naïve Bayes is one of the most popular data mining algorithms. Its efficiency comes from the assumption of attribute independence, although this might be violated in many real-world data sets. Many efforts have been done to mitigate the assumption, among which attribute selection is an important approach. However, conventional efforts to perform attribute selection in naïve Bayes suffer from heavy computational overhead. This paper proposes an efficient selective naïve Bayes algorithm, which adopts only some of the attributes to construct selective naïve Bayes models. These models are built in such a way that each one is a trivial extension of another. The most predictive selective naïve Bayes model can be selected by the measures of incremental leave-one-out cross validation. As a result, attributes can be selected by efficient model selection. Empirical results demonstrate that the selective naïve Bayes shows superior classification accuracy, yet at the same time maintains the simplicity and efficiency.

1. Introduction

Naïve Bayes (NB) is one of the most well-known data mining algorithms for classification [1]. It infers the probability that one new example belongs to some class based on the assumption that all attributes are independent of each other given the class [2]. This assumption is motivated by the need to estimate the multi-variate probabilities from the training data. In practice, most combinations of attribute values are either not present in the training data or not present in sufficient numbers. Consequently, direct estimation of each relevant multi-variate probability will not be reliable. Naïve Bayes circumvents this predicament by its conditional independence assumption. In spite of this strict independence assumption, naïve Bayes is a really competent classifier in many real-world applications [3].

Although naïve Bayes has already exhibited remarkable classification accuracy, the conditional independence assumption is rarely true in reality. As a result, it is natural to improve naïve Bayes by relaxing the conditional independence assumption. Approaches to doing so includes attribute weighting, attribute selection, structure extension, and so on.

Previous researches [4,5] show that selecting only some of the attributes might result in better classification accuracy. The conventional greedy search method [4] adds (or removes) an attribute to (or from) an earlier model after evaluating all possible additions or removals of one attribute in one pass through the training data. Since only one attribute is added or removed after one pass, multiple passes through the training data are required to obtain the final model, which involves heavy computational overhead.

This paper proposes an efficient attribute selecting algorithm, called Selective Naïve Bayes (SNB). It adopts only some of the attributes to construct the selective naïve Bayes models. The attribute selection is done through model selection because different model in SNB implies different number of attributes that will be used when classifying an example. The proposed approach evaluates all the possible additions of one attribute, two attributes, $\cdots$, and a attributes, respectively, from empty attribute set in only one pass through the training data, where a is the number of attributes. Consequently, the evaluation of all the a models needs only one pass, instead of multiple passes through the training data. It is efficient because multiple selective naïve Bayes models that have been built in such a way that each is only a trivial extension to another can be evaluated using incremental leave-one-out cross validation in a single extra pass through the training data relative to standard naïve Bayes. Thus, it involves two passes through the training data.

The remainder of this paper is organized as follows. Section 2 reviews the related work with regard to this paper. Section 3 describes the principle of naïve Bayes as the preliminary. Section 4

[☆] No author associated with this paper has disclosed any potential or pertinent conflicts which may be perceived to have impending conflict with this work. For full disclosure statements refer to https://doi.org/10.1016/j.knosys.2019.105361.

[*] Corresponding author.
E-mail address: shenglei.chen@nau.edu.cn (S. Chen).

https://doi.org/10.1016/j.knosys.2019.105361
0950-7051/© 2019 Elsevier B.V. All rights reserved.

proposes the selective naïve Bayes. Section 5 describes in detail the experimental setup and the results. The last section draws the conclusions and outlines the main directions for our future work.

2. Related work

In order to mitigate the effect resulted by the attribute independence assumption, numerous approaches have been proposed in recent years. The existing work can be broadly divided into three categories: attribute weighting, attribute selection, and structure extension.

2.1. Attribute weighting

In regular naïve Bayes, each attribute has been treated equally. However, in real-world applications, different attribute plays different role in discrimination of the class. Some of them are more important than others. Thus a natural way to improve naïve Bayes is to assign different weights to different attributes. This motivates the research of attribute weighting in naïve Bayes.

Zhang and Sheng [6] proposed a gain ratio-based feature weighting method (GRFW). They argued that a feature with higher gain ratio deserves a larger weight and therefore set the weight of each feature to the gain ratio of the feature relative to the average gain ratio across all features.

Hall [7] proposed a decision tree-based feature weighting (DTFW) approach for standard naïve Bayes. The approach weights each feature according to the degree to which it depends on other features' values and assigns lower weights to those features that have many dependencies. To estimate the degree to which a feature depends on others, an unpruned decision tree is built from the training data and the minimal depth d at which the feature is tested in the built tree is saved, and then the weight for this feature is set to $1/\sqrt{d}$. If a feature does not appear in the built tree, the weight is set to zero. In order to stabilize the estimated weights, multiple decision trees are built by using bagging and the weights are averaged across the ensemble. These two feature weighting approaches, GRFW and DTFW, have been adapted for naïve Bayes text classification [3].

Lee et al. [8] proposed a new method for calculating the weights of features in naïve Bayesian learning using weighted average of the Kullback–Leibler measure across the feature values. Then they proposed a new paradigm of assigning weights in classification learning, called value weighting method. Instead of weighting each feature, they assign different weight to each feature value. They presented two approaches to solve the value weighted problem. One is the gradient approach [9]. In order to learn the optimal weights of feature values, an optimization problem is constructed by defining the objective function as the sum of squared error. The batch gradient descent method is employed to solve the optimization problem. The other is the filter approach [10], in which Kullback–Leibler measure is employed for calculating value weights.

Zaidi et al. [11] proposed a weighted naïve Bayes algorithm, Weighting attributes to Alleviate Naïve Bayes' Independence Assumption (WANBIA), that introduces weights in naïve Bayes and learns these weights in a discriminative fashion that is minimizing either the negative conditional log likelihood or the mean squared error objective functions. They employ gradient descent searches to optimize feature weights.

Jiang et al. [12] argued that highly predictive features for NB should be highly correlated with the class (maximum mutual relevance), yet uncorrelated with other features (minimum mutual redundancy). Based on this premise, they proposed a correlation-based feature weighting (CFW) filter for NB, in which the weight for a feature is proportional to the difference between the feature-class correlation (mutual relevance) and the average feature-feature intercorrelation (average mutual redundancy).

2.2. Attribute selection

Attribute selection techniques can broadly fall into the wrapper methods or the filter methods [13]. Wrapper methods search the space of feature subsets, using the validation accuracy of a particular classifier as the measure of utility for a candidate subset. In contrast, filter methods separate the classification and feature selection components, and define a heuristic scoring criterion to act as a proxy measure of the classification accuracy.

Langley and Sage [4] proposed to embed the naïve Bayesian induction scheme within an algorithm that carried out a greedy search through the space of features, called Selective Bayes Classifiers (SBC). The search starts with the empty attribute set and successively add attributes, called forward attribute addition (FSA). They employed incremental leave-one-out technique for estimating accuracy from the training set. Since the search strategy in SBC [4] is so greedy that it often falls into a local optimization, Jiang et al. [14] proposed an improved naïve Bayes algorithm by carrying a random search through the whole space of attributes, called Randomly Selected Naïve Bayes (RSNB). They designed three different versions of RSNB to meet the need of classification, ranking, and class probability estimation. Bermejo et al. [5] proposed to combine the naïve Bayes classifier with incremental wrapper feature subset selection (FSS) algorithm so as to speed up the wrapper FSS process. The resulting method is an embedded version of the classifier into the incremental selection algorithm. All these methods are typical wrapper methods.

A famous filter method is Correlation-based Feature Selection (CFS) proposed by Hall [15]. CFS applies a correlation measure to evaluate the goodness of feature subsets based on the hypothesis that a good feature subset is one that contains features highly correlated with the class, yet uncorrelated with each other. The correlation measure is, in fact, Pearson's correlation, where all variables have been standardized. CFS uses a best first strategy to search within 2^a possible subsets, where a is the features number, as it gives slightly better results in some cases than hill climbing. Another well known filter method is the fast correlation-based filter (FCBF) proposed by Yu and Liu [13]. FCBF can identify relevant features as well as redundancy among relevant features without pairwise correlation analysis.

2.3. Structure extension

The most well known work that can alleviate the independence assumption of NB is the structure extension, known as semi-naïve NBCs [16]. Friedman et al.'s Tree Augmented Naïve Bayes (TAN) [17] approximates the interactions between attributes by a tree structure imposed on the NB structure. That is to say, it requires that the class variable has no parents and each attribute has as parents the class variable and at most one other attribute. They developed an algorithm to learn TAN classifiers in quadratic time, which extends a well-known result by Chow and Liu [18]. TAN is a one pass algorithm, because the probability distributions required for selecting the network structure and parameterizing the conditional probability tables can be obtained in one pass learning through the training examples.

Another significant improvement to NB is AnDE [19,20], which relaxes the attribute independence assumption and averages over all possible n-dependence estimators (nDE), with the aim of reducing the inductive bias in the classifier. It requires no structure searching and only one single pass learning through the training examples. Since AnDE allows every attribute to depend on n common super parent attributes, which is more consistent with the characteristics of real data sets, it has lower bias than Naïve Bayes and TAN. Yu et al. [21] proposed a new approach called attribute value weighted average of one-dependence estimators (AVWAODE), which assigns discriminative weights to different one-dependence estimators by computing the correlation between the different root attribute value and the class. They use two different attribute value weighting measures: the Kullback–Leibler measure and the information gain measure.

k-Dependence Bayesian classifier (KDB) [22] is another famous improvement to NB. It relaxes NB's independence assumption by allowing each attribute to have a maximum of k attributes as parents. In this sense, NB is a 0-dependence Bayesian classifier and TAN is a 1-dependence Bayesian classifier. By increasing the value of k, KDB can generalize to higher degrees of attribute dependence than TAN. KDB constructs classifiers at arbitrary values of k, while retaining much of the computational efficiency of NB. It is a two pass algorithm. The first pass collects the statistics required for selecting a network structure in which each attribute has at most k parents. The second pass computes the conditional probability tables inferred by the structure of k-dependence Bayesian network.

Langseth and Nielsen [23] proposed a relatively new set of models, termed hierarchical naïve Bayes models, which extend the modeling flexibility of naïve Bayes models by introducing latent variables to relax some of the independence statements in these models. They proposed a simple algorithm for learning hierarchical naïve Bayes models in the context of classification. They focus on learning hierarchical naïve Bayes models with the sole aim of obtaining an accurate classifier. The learning algorithm is based on a greedy search over the space of hierarchical naïve Bayes models.

In hidden NB [24,25], a hidden parent is created for each attribute which combines the influences from all other attributes. They presented an approach to creating hidden parents using the average of weighted one-dependence estimators. Hidden NB inherits the structural simplicity of naïve Bayes and can be easily learned without structure searching. They proposed an algorithm for learning hidden NB based on conditional mutual information. They constructed the weight by mutual information, and then averaged the one-dependence estimators as the hidden parent.

Jiang et al. [26] proposed random one-dependence estimators (RODE) by augmenting the structure of naïve Bayes. In RODE, each attribute has at most one parent from other attributes and this parent is randomly selected from $log_2 a$ (where a is the number of attributes) attributes with the maximal conditional mutual information.

Harzevili et al. [27] proposed Mixture of Latent Multinomial Naïve Bayes (MLMNB) classifier to relax the independence assumption of naïve Bayes. MLMNB incorporates a latent variable in a predefined Bayesian network structure to model the dependencies among attributes, yet avoids burden complexities of structure learning approaches. The latent variable is beside the class variable as the parent of all other attributes. Expectation maximization (EM) algorithm is modified for the parameter estimation.

A classifier is typically measured by its classification accuracy on testing instances. In many real-world applications, however, accurate class probability estimation of instances is more desirable than simple classification. Qiu et al. [28] proposed an improved Conditional Log Likelihood (CLL)-based SuperParent (SP) algorithm, in which CLL is used to find the augmenting arcs. The new approach significantly outperforms the classification-based SP approach [29] and the original distribution-based approach [17] in terms of CLL, yet at the same time maintains the high classification accuracy that characterizes the classification-based SP approach.

The above researches focus on classification problems with nominal attributes, on which NB can be implemented efficiently. For tasks with continuous attributes, two strategies can be conducted. One is the prior discretization [30] or inbetween discretization [31], and the other is the density estimation. The density estimation strategy intends to find an underlying distribution for continuous attributes. The distribution can be assumed to be a single Gaussian distribution [32], or a non-Gaussian distribution [33–35]. Nevertheless, these methods are all based on such an assumption that all attributes are conditionally independent with each other, which does not always hold in many real-world applications. In order to remove or relax the restriction of independence among attributes and obtain a better estimation of the true p.d.f., Wang et al. [36] proposed a non-naïve Bayesian classifier where a model of joint p.d.f is estimated by using Parzen windows based on the multivariate kernel function.

Table 1
Legend of symbols.

Symbols	Definition
$\mathcal{D}$	Set of data examples
$\mathcal{D}_{train}$	Set of training examples
$\mathcal{D}_{test}$	Set of test examples
X, X_i	Discrete random variable representing the attribute
x, x_i	Value of attribute variable X or X_i
$\mathbf{x} = \langle x_1, \ldots, x_a \rangle$	Vector representing an example
Y	Discrete random variable representing the class
y	Value of class variable Y
t	Number of training examples
c	Number of classes
a	Number of attributes
s	Number of selected attributes out of a attributes
v	Average number of values per attribute

3. Naïve Bayes

In this section, we present the principle of naïve Bayes [1]. The classification task that NB solves is assumed as follows. Given a training data sample $\mathcal{D}_{train}$ 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 X_i and $y \in \{1, \ldots, c\}$ is the value of class variable Y.

Many different symbols will be used in this paper. In order to make the reading of paper more clear, all the symbols that will be defined in this paper are summarized in Table 1.

3.1. Predicting new examples

From the definition of conditional probability [37], we have

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

As $\mathrm{P}(\mathbf{x}) = \sum_{y=1}^{k} \mathrm{P}(y, \mathbf{x})$, 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.

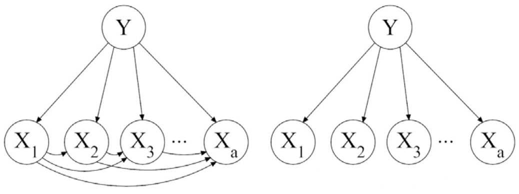
If example $\mathbf{x}$ does not appear frequently enough in the training data, we cannot directly derive accurate estimates of $\mathrm{P}(y, \mathbf{x})$ and must extrapolate these estimates from observations of lower-dimensional probabilities in the data [20]. As a result, we apply the definition of conditional probabilities again and get

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

The first term $\mathrm{P}(y)$ on the right-hand side can be sufficiently accurately estimated from the sample frequencies, if the number of classes, k, is not a huge number. In order to compute the second term $\mathrm{P}(\mathbf{x} \mid y)$ based on low-dimensional probabilities, we factorize it by the chain rule,

$$\mathrm{P}(\mathbf{x} \mid y) = \prod_{i=1}^{a} \mathrm{P}(x_i \mid x_1, x_2, \ldots, x_{i-1}, y). \tag{1}$$

Eq. (1) is corresponding to the Bayesian network [22] shown in Fig. 1(a). This is optimal in theory. But the conditional probability



(a) Bayes (b) Naïve Bayes

Fig. 1. Bayes and naïve Bayesian networks.

$\mathrm{P}(x_i \mid x_1, x_2, \ldots, x_{i-1}, y)$ cannot be sufficiently accurately estimated for data sets with large number of attributes, as the large number of feature dependence arcs results in high complexity. As a result, NB assumes the attributes are independent of each other given the class and simplifies the computation of $\mathrm{P}(\mathbf{x} \mid y)$,

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

Eq. (2) is corresponding to the network structure in Fig. 1(b).

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

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

Thus, NB classifies a new 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 m-estimation [38]. The time complexity of classifying one example is $\mathcal{O}(ca)$.

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 [39].

3.2. Parameters learning

As $\mathrm{P}(y)$ can be inferred easily from $\mathrm{P}(x_i \mid y)$, we will focus on how to learn the probability $\mathrm{P}(x_i \mid y)$ next. In order to estimate $\mathrm{P}(x_i \mid y)$, we should first count the number of training examples with occurrence of the jth state of X_i given its parent y, where $j \in \{1, \ldots, |X_i|\}$. That means we should keep a frequency for each tuple $\langle i, j, y \rangle$. In practice, we define a three dimensional table to save the frequency of examples with occurrence of tuple $\langle i, j, y \rangle$. For example, a table of frequencies for a data set with two attributes and a class variable is represented as in Fig. 2, where $|X_1| = 2$, $|X_2| = 3$, and the number of classes $c = 2$.

The process to populate each element of the table is as follows. For each training example, we can get the value of class variable, suppose it is the yth value. Then for each attribute X_i, we can get the attribute value, suppose it is the jth value. As we find one example with occurrence of tuple $\langle i, j, y \rangle$, we increase the element in the table with index (i, j, y) by 1. This process is summarized in Algorithm 1.

After all the instances in the training set have been scanned, each entry in the table represents the joint frequency of attribute-value and class. Next we can estimate the probability $\mathrm{P}(x_i \mid y)$ using m-estimate [38], where m is set to 1.0 in practice,

Class Att-Value		class 1	class 2
X_1	Value 1		
	Value 2		
X_2	Value 1		
	Value 2		
	Value 3		

Fig. 2. Table of frequencies for a data set with two attributes and a class variable.

Algorithm 1 Training process of NB

```
Count : table of observed counts of combination of 1 attribute value and the class label
for instance inst ∈ $\mathcal{D}_{train}$ do
    get the value of class variable in inst, suppose it is the yth value
    for $X_i$, $i \in \{1, 2, \ldots, a\}$ do
        get the value of attribute $X_i$ in inst, suppose it is the jth value
        increase the element in Count with index $(i, j, y)$ by 1
    end for
end for
```

$$\hat{\mathrm{P}}(x_i \mid y) = \frac{Count(i, j, y) + m * \frac{1}{|X_i|}}{\sum_{j=1}^{|X_i|} Count(i, j, y) + m}.$$

The space complexity of the frequency table is $\mathcal{O}(cav)$, where v is the average number of values per attribute. The time complexity of compiling it is $\mathcal{O}(at)$, as we need to update each entry for every attribute-value for every instance.

4. Selective naïve Bayes

In this section, we first present the motivation behind the selective naïve Bayes algorithm. Then we describe how to build the model space and select the best model. Finally, we provide the selective naïve Bayes algorithm.

4.1. Motivation

We could observe from Eq. (3) that the joint probability $\mathrm{P}_{\mathrm{NB}}(y, \mathbf{x})$ is estimated by the product of prior probability $\mathrm{P}(y)$ and conditional probabilities $\mathrm{P}(x_i \mid y)$. And the computation process actually contains multiple approximations to $\mathrm{P}_{\mathrm{NB}}(y, \mathbf{x})$. These observations imply that it is possible to nest a space of alternative selective models such that each one is a trivial extension to another. Importantly, multiple models that build upon one another in this way can be efficiently evaluated in a single set of computations.

In the light of these observations, we create a space of models that are nested together, and then select the model with the lowest prediction error of leave-one-out cross validation in a single extra pass through the training data.

4.2. Model space

From Eq. (3), we could see that at most a nested selective models will be created when calculating $\mathrm{P}_{\mathrm{NB}}(y, \mathbf{x})$. To be more

Selective NB models	Attributes considered
$\mathrm{P}_{\mathrm{NB}}(y,\mathbf{x})_1$	$\{x_3\}$
$\mathrm{P}_{\mathrm{NB}}(y,\mathbf{x})_2$	$\{x_3, x_1\}$
$\mathrm{P}_{\mathrm{NB}}(y,\mathbf{x})_3$	$\{x_3, x_1, x_2\}$

Fig. 3. Model space example with 3 attributes.

specific, suppose we multiply $\mathrm{P}(y)$ by only the first s conditional probabilities $\mathrm{P}(x_i \mid y)$, $s \in \{1, 2, \dots, a\}$, the selective NB model would be,

$$\mathrm{P}_{\mathrm{NB}}(y, \mathbf{x})_s = \mathrm{P}(y) \prod_{i=1}^{s} \mathrm{P}(x_i \mid y), s \in \{1, 2, \dots, a\}. \tag{4}$$

All these selective NB models form a model space. Because each model is only a minor extension to previous model, all these models can be applied to a test instance in a single nested computation. Consequently all models can be efficiently evaluated.

It is worthwhile to note that the construction of the model space assumes that there is an ordering on the attributes. The reason is that models containing attributes that are later in the ordering will be built upon models containing earlier attributes, as is demonstrated in Eq. (4). As the attributes which are more correlated with the class are preferable, the attributes should be ranked in the descending order of correlation with the class.

Mutual information measures how informative one attribute is about the class [40], as such it is a suitable metric to rank the attributes. Mutual information between an attribute X and the class Y is defined as:

$$\begin{aligned} I(X, Y) &= H(X) - H(X \mid Y) \\ &= \sum_{y \in Y} \sum_{x \in X} \mathrm{P}(x, y) log_2 \frac{\mathrm{P}(x, y)}{\mathrm{P}(x)\mathrm{P}(y)}, \end{aligned} \tag{5}$$

where $H(X) = -\sum_{x \in X} \mathrm{P}(x) log \mathrm{P}(x)$ is the entropy of X, and $H(X \mid Y) = -\sum_{y \in Y} \mathrm{P}(y) \sum_{x \in X} \mathrm{P}(x \mid y) log \mathrm{P}(x \mid y)$ is the conditional entropy.

Here we give an example to show how the model space is constructed. Suppose we have an example with 3 attributes, $\mathbf{x} = \langle x_1, x_2, x_3 \rangle$. After we rank the attributes based on mutual information, the order is x_3, x_1, x_2. So there are 3 selective NB models in the model space, as is shown in Fig. 3. Each model corresponds to a subset of attributes. For example, $\mathrm{P}_{\mathrm{NB}}(y, \mathbf{x})_2$ considers the first 2 attributes in the ordered attribute. We could see that each model is only a minor extension to the previous one, for example, $\mathrm{P}_{\mathrm{NB}}(y, \mathbf{x})_3$ could be easily obtained by multiplying $\mathrm{P}(x_2 \mid y)$ to $\mathrm{P}_{\mathrm{NB}}(y, \mathbf{x})_2$. As a result, all these models can be applied to a test instance in a single nested computation and thus be evaluated very efficiently.

4.3. Model selection

To evaluate the discriminative ability of alternative models and avoid over fitting on training data, we use leave-one-out cross validation error as the evaluation criterion [4]. Rather than building new models for every fold, we exploit incremental cross validation [41], in which the contribution of the training example being left out in each fold is simply subtracted from the frequency table, thus producing a model without that training example. This method not only obtains a low-bias estimate of the generalization error, but also allows the models to be evaluated in one pass through the training data.

In addition, the fact that the models are nested together such that each one is a trivial extension to another, as is shown in Eq. (4), provides us a way to efficiently evaluate these models. That is to say, for the training example being left out in each fold, these models can be simultaneously evaluated inside the process of construction of them. The process of leave-one-out cross validation has been demonstrated in Algorithm 2 (line 4–9).

There are several loss functions to measure model performance for leave-one-out cross validation. Zero–one loss and root mean squared error (RMSE) are among the most common and effective ones. Zero–one loss simply assigns a loss of '0' to correct classification, and '1' to incorrect classification, treating all misclassifications as equally undesirable. RMSE, however, accumulates for each example the squared error, where the error is the difference between 1.0 and the probability estimated by the algorithm for the true class for the example, and then computes the squared root of the mean. This could be computed as,

$$RMSE = \sqrt{\frac{1}{t} \sum_{\mathbf{x} \in \mathcal{D}_{train}} (1 - P(y \mid \mathbf{x}))^2}\ , \tag{6}$$

where y is the true class for the example $\mathbf{x}$. As RMSE gives a finer grained measure of the calibration of the probability estimates compared to zero–one loss, with the error depending not just on which class is predicted, but also on the probabilities estimated for the true class, we use RMSE to evaluate the candidate models in this research.

Consequently, selecting the best model can be described as the following optimization problem,

$$s^* = \operatorname*{argmin}_{s \in \{1,2,\dots,a\}} \sqrt{\frac{1}{t} \sum_{\mathbf{x} \in \mathcal{D}_{train}} \left(1 - \mathrm{P}_{NB}^{LOO}(y \mid \mathbf{x})_s\right)^2} \tag{7}$$

where $\mathrm{P}_{NB}^{LOO}(y \mid \mathbf{x})_s$ can be computed by first estimating $\mathrm{P}_{NB}^{LOO}(y, \mathbf{x})_s$ from training set $(\mathcal{D}_{train} - \{\langle y, \mathbf{x} \rangle\})$ as in Eq. (4), and then normalizing across all possible Y. The best model actually corresponds to an optimal subset of attributes, which is formed by the first s^* attributes in the ordered attributes.

4.4. Selective naïve Bayes algorithm

The training algorithm of selective naïve Bayes is summarized in Algorithm 2. It involves two passes learning through the training data $\mathcal{D}_{train}$. One pass is to form the frequency table from which the probability estimates $\hat{\mathrm{P}}(x_i \mid y)$ and the mutual information between the attributes and class are derived (line 1). The other pass is the leave-one-out cross validation process in which the squared errors for all selective models are accumulated(line 4–9).

Algorithm 2 Training algorithm of selective naïve Bayes

```
1: Form the table of joint attribute-value and class frequencies
   as in Algorithm 1
2: Compute the mutual information according to Eq. (5)
3: Rank the attributes in the descending order of mutual
   information with the class
4: for instance inst ∈ $\mathcal{D}_{train}$ do
5:     Remove inst from the frequency table
6:     Predict inst using all models in Eq. (4)
7:     Accumulate the squared error for each model
8:     Add inst back to the frequency table
9: end for
10: Compute the root mean squared error for each model as in
    Eq. (6)
11: Select the model with the lowest RMSE as in Eq. (7)
```

The space complexity of SNB in each fold is still $\mathcal{O}(cav)$, where v is the average number of values per attribute, as it preserves the same frequency table as NB. The training process of each fold

Table 2
Space and time complexity of SNB and NB in each fold.

Complexity		Algorithms	
		SNB	NB
Space		$\mathcal{O}(cav)$	$\mathcal{O}(cav)$
Time	Training	$\mathcal{O}((c+1)at)$	$\mathcal{O}(at)$
	Test	$\mathcal{O}(cs^*)$	$\mathcal{O}(ca)$

comprises two passes through training data. One is the compilation of the frequency table, the complexity of which is still $\mathcal{O}(at)$. The other is the leave-one-out cross validation, in which it needs to evaluate the probability for each attribute-class tuple for each training example. The complexity of this validation is $\mathcal{O}(cat)$. So the time complexity of the training process is $\mathcal{O}((c+1)at)$. The test process evaluates the conditional probabilities of the first s^* attributes for each class for the testing example. So the time complexity of classifying one example is $\mathcal{O}(cs^*)$ (see Table 2).

5. Experiments and results

Experiments were performed on a quad-processor Core (TM) i7-4790 @ 3.6 GHz Linux computer with 8 GB RAM. The experimental system is implemented in C++.

We run the experiments on 65 data sets from the UCI repository [42] as listed in Table 3. Note that the data sets have been listed in the order of increasing number of instances. 5-bin equal frequency discretization is conducted to discretize the numeric attributes as in [43]. The base probabilities are estimated using m-estimation ($m = 1$) [38]. Missing values have been considered as a distinct value. The experiments have been done with the 10-fold cross validation method.

In the experiments, we will compare SNB to NB and such two weighting improvements of NB as GRFW [6] and WANBIA [11] which adopts negative conditional log likelihood as the objective function. At the same time, we also compare with two higher order Bayesian network classifiers, TAN [17] and AODE [19].

5.1. Comparison of zero–one loss and RMSE

First of all, we compare SNB with alternatives in terms of two measures of classification performance. One is the zero–one loss (ZOL), and the other is RMSE as we adopt RMSE as model selection criterion. The RMSE is defined as follows,

$$RMSE = \sqrt{\frac{1}{|\mathcal{D}_{test}|}\sum_{\mathbf{x}\in\mathcal{D}_{test}}(1 - P(Y = y \mid \mathbf{x}))^2}\ . \tag{8}$$

Tables A.8 and A.9 in the Appendix present the results of ZOL ± standard deviation and RMSE ± standard deviation, respectively, of all 6 algorithms on 65 data sets.

5.1.1. Win/draw/loss analysis

In order to give the results a more intuitive explanation, we present summaries of win/draw/loss records of alternative algorithms in Tables 4 and 5, which indicates the number of data sets on which SNB algorithm has lower, equal or higher measure relative to the alternative. 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.

Table 3
Data sets.

No.	Name	Inst	Att	Class
1	contact-lenses	24	4	3
2	lung-cancer	32	56	3
3	labor-negotiations	57	16	2
4	post-operative	90	8	3
5	zoo	101	16	7
6	promoters	106	57	2
7	echocardiogram	131	6	2
8	lymphography	148	18	4
9	iris	150	4	3
10	teaching-ae	151	5	3
11	hepatitis	155	19	2
12	wine	178	13	3
13	autos	205	25	7
14	sonar	208	60	2
15	glass-id	214	9	3
16	new-thyroid	215	5	3
17	audio	226	69	24
18	hungarian	294	13	2
19	heart-disease-c	303	13	2
20	haberman	306	3	2
21	primary-tumor	339	17	22
22	ionosphere	351	34	2
23	dermatology	366	34	6
24	horse-colic	368	21	2
25	house-votes-84	435	16	2
26	cylinder-bands	540	39	2
27	chess	551	39	2
28	syncon	600	60	6
29	balance-scale	625	4	3
30	soybean	683	35	19
31	credit-a	690	15	2
32	breast-cancer-w	699	9	2
33	pima-ind-diabetes	768	8	2
34	vehicle	846	18	4
35	anneal	898	38	6
36	tic-tac-toe	958	9	2
37	vowel	990	13	11
38	german	1 000	20	2
39	led	1 000	7	10
40	contraceptive-mc	1 473	9	3
41	yeast	1 484	8	10
42	volcanoes	1 520	3	4
43	car	1 728	6	4
44	segment	2 310	19	7
45	hypothyroid	3 163	25	2
46	splice-c4.5	3 177	60	3
47	kr-vs-kp	3 196	36	2
48	abalone	4 177	8	3
49	spambase	4 601	57	2
50	phoneme	5 438	7	50
51	wall-following	5 456	24	4
52	page-blocks	5 473	10	5
53	optdigits	5 620	64	10
54	satellite	6 435	36	6
55	musk2	6 598	166	2
56	mushrooms	8 124	22	2
57	thyroid	9 169	29	20
58	pendigits	10 992	16	10
59	sign	12 546	8	3
60	nursery	12 960	8	5
61	magic	19 020	10	2
62	letter-recog	20 000	16	26
63	adult	48 842	14	2
64	shuttle	58 000	9	7
65	connect-4	67 557	42	3

We can see that SNB decreases zero–one loss and RMSE more often than not relative to both NB and GRFW. And these differences are statistically significant. SNB achieves lower zero–one loss and RMSE significantly less often than both WANBIA and AODE, while it achieves lower RMSE almost as often as higher than TAN. The reason might be that WANBIA directly optimizes the conditional log-likelihood function and learns the weights in

Table 4
Win/draw/loss records SNB vs. alternatives in terms of ZOL on 65 data sets.

NB		GRFW		WANBIA		TAN		AODE	
W/D/L	*p*	W/D/L	*p*	W/D/L	*p*	W/D/L	*p*	W/D/L	*p*
33/14/18	**0.0489**	45/1/19	**0.0016**	21/1/43	**0.0081**	29/1/35	0.5323	18/4/43	**0.0019**

Table 5
Win/draw/loss records SNB vs. alternatives in terms of RMSE on 65 data sets.

NB		GRFW		WANBIA		TAN		AODE	
W/D/L	*p*	W/D/L	*p*	W/D/L	*p*	W/D/L	*p*	W/D/L	*p*
36/9/20	**0.044**	55/0/10	<**0.0001**	22/0/43	**0.0125**	30/0/35	0.6201	16/1/48	<**0.0001**

Table 6
Average ranks of the algorithms.

Algorithm	ZOL rank	RMSE rank
GRFW	2.5000	2.1385
NB	2.9923	3.1000
NB-loo	3.4077	3.5231
TAN	3.5923	3.6308
WANBIA	4.1923	4.0615
AODE	4.3154	4.5462
F_F statistic	10.2653	15.6653

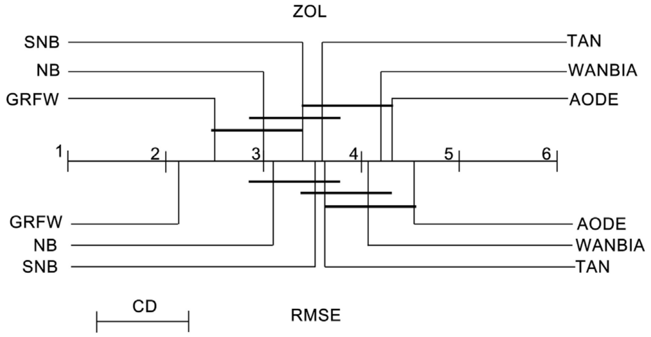


Fig. 4. Comparison of six algorithms against each other with the Nemenyi test.

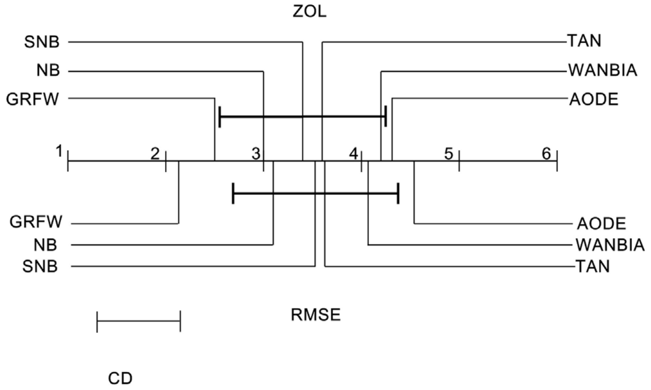


Fig. 5. Comparison of SNB against the others with the Bonferroni–Dunn test.

a discriminative fashion. AODE assumes all attributes depend on one common attribute rather than attribute independence as in NB. However, WANBIA and AODE's superior accuracy comes at considerable cost in increased training time.

The fact that SNB is comparable with TAN indicates that model selection is a powerful technique because TAN allows dependence among attributes while SNB does not.

5.1.2. Friedman test

According to Demšar [44] and García and Herrerato [45], we perform the nonparametric Friedman test followed by the post hoc Nemenyi test to statistically compare these 6 algorithms on 65 data sets.

The average ranks of the algorithms obtained by applying the Friedman test with respect to ZOL and RMSE are shown in Table 6. Note that both ZOL and RMSE correspond to error rate in nature, so the higher the algorithm ranks, the better performance the algorithm has. The null hypothesis of the Friedman test is that all algorithms are equivalent. At the bottom of Table 6, we present the Iman and Davenport's F_F statistic, which is distributed according to the F distribution with $6-1=5$ and $(6-1)\times(65-1)=320$ degrees of freedom, as this is not as conservative as Friedman's χ_F^2 statistic [44]. We could see that the F_F statistics for both ZOL and RMSE are greater than 2.21, which is the critical value of $F(5,320)$ for $\alpha=0.05$. So we reject the null hypothesis. This implies that these 6 algorithms are not equivalent in terms of both ZOL and RMSE.

In order to further investigate which algorithm is significantly different to others, we perform the Nemenyi test, which states that the performance of two classifiers is significantly different if the corresponding average ranks differ by at least the critical difference $CD = q_\alpha\sqrt{\frac{6\times7}{6\times65}}$, where the critical value q_α is 2.85 for $\alpha = 0.05$ [44]. So the value of critical difference CD for $\alpha=0.05$ is 0.9353. Fig. 4 graphically represents the comparison of 6 algorithms against each other with the Nemenyi test. The middle line in the diagram is the axis on which we plot the average ranks of different algorithms. Above the axis are the ranks with respect to ZOL, and below are those with respect to RMSE. When comparing all the algorithms against each other, we connect the groups of algorithms that are not significantly different. The critical difference is showed below the graph. The analysis reveals that with respect to RMSE, SNB performs significantly worse than AODE, significantly better than GRFW, and not significantly different with NB, TAN and WANBIA. With respect to ZOL, SNB does not perform significantly different with all the other 5 algorithms.

As the Nemenyi test is not powerful enough to detect the difference between these algorithms, we further compare the proposed approach with the other algorithms with the Bonferroni–Dunn test. At $p=0.05$, CD is $2.576\times\sqrt{\frac{6\times7}{6\times65}}=0.8454$. We mark the interval of one CD to the left and right of the average rank of SNB as in Fig. 5. As the ranks of GRFW and AODE are outside the marked interval in terms of both RMSE and ZOL, we can conclude that SNB performs significantly better than GRFW, and significantly worse than AODE. At the same time, the performance of SNB is comparable to those of NB, TAN and WANBIA.

These results illustrate the limitations of Nemenyi and Bonferroni–Dunn approaches. While it might be considered a negative that with the reduced statistical power resulting from these multiple testing corrections SNB is not considered significantly often more accurate than NB, it could also be considered a

Table 7
Average running time on 65 data sets of 6 algorithms.

	SNB	NB	GRFW	WANBIA	TAN	AODE
Average running time (s)	0.123	0.015	0.015	1003.185	0.062	0.262
Logarithmic value	−2.095	−4.174	−4.174	6.911	−2.788	−1.341

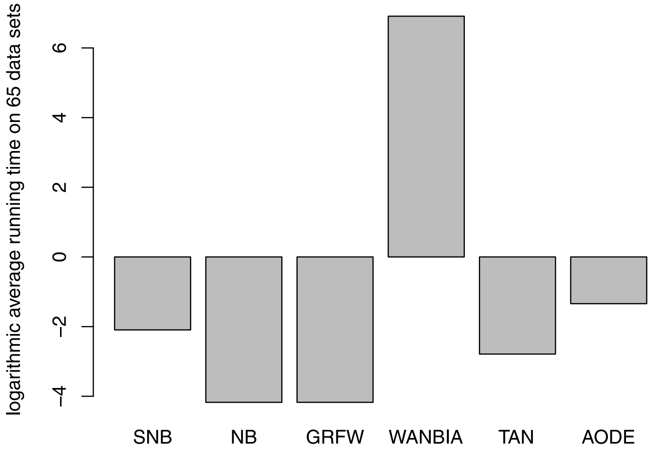


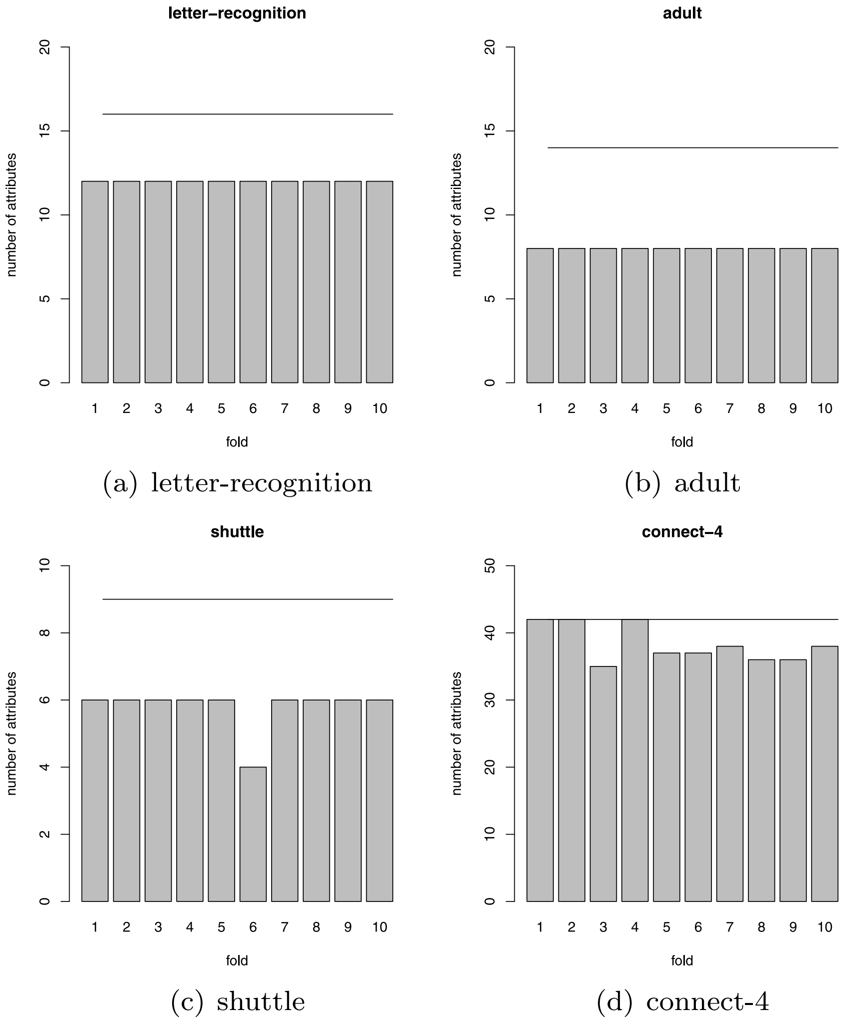
Fig. 6. Comparison of logarithmic average running time on 65 data sets of 6 algorithms.

tremendous positive that it is not considered significantly often less accurate than more complex techniques TAN and WANBIA.

5.2. Comparison of running time

This section will compare the average running time of all 6 algorithms on 65 data sets. Because most data sets are small, the separated training time and classification time on those data sets are zeros. So we report only the total time of each running. Table 7 lists the average running time in seconds on 65 data sets of 6 algorithms. Because there is a big difference in the magnitude, we compute the logarithmic value of each average running time. Fig. 6 displays the bar graph of the logarithmic average running time of 6 algorithms.

We can find that NB and GRFW require the least running time. SNB requires more running time than NB and GRFW. Its running time is comparable to those of TAN and AODE. WANBIA requires the most running time as it involves computationally intensive gradient descent search.



(a) letter-recognition

(b) adult

(c) shuttle

(d) connect-4

Fig. 7. Number of attributes selected in each fold on 4 data sets.

Table A.8
Zero–one loss.

Data set	SNB	NB	GRFW	WANBIA	TAN	AODE
contact-lenses	0.3750 ± 0.3425	0.3750 ± 0.3425	0.1250 ± 0.1955	0.3750 ± 0.3504	0.3750 ± 0.3758	0.4167 ± 0.3574
lung-cancer	0.4062 ± 0.3342	0.4375 ± 0.2684	0.3438 ± 0.3016	0.5625 ± 0.3042	0.5938 ± 0.2265	0.4688 ± 0.2885
labor-negotiations	0.0526 ± 0.0618	0.0351 ± 0.0422	0.0877 ± 0.1178	0.1404 ± 0.1546	0.1053 ± 0.1234	0.0526 ± 0.0675
post-operative	0.3000 ± 0.1710	0.3444 ± 0.1966	0.4000 ± 0.2117	0.3111 ± 0.2031	0.3667 ± 0.2075	0.3444 ± 0.1882
zoo	0.0198 ± 0.0384	0.0297 ± 0.0477	0.0198 ± 0.0384	0.0297 ± 0.0477	0.0099 ± 0.0527	0.0198 ± 0.0384
promoters	0.0660 ± 0.0596	0.0755 ± 0.0617	0.0755 ± 0.0719	0.0566 ± 0.0607	0.1321 ± 0.1036	0.1038 ± 0.0648
echocardiogram	0.3588 ± 0.1340	0.2748 ± 0.1347	0.2824 ± 0.0907	0.2977 ± 0.1490	0.3664 ± 0.1549	0.3435 ± 0.1143
lymphography	0.1486 ± 0.0979	0.1486 ± 0.0979	0.1824 ± 0.0967	0.1689 ± 0.1054	0.1757 ± 0.1003	0.1486 ± 0.0991
iris	0.0800 ± 0.0687	0.0733 ± 0.0693	0.0867 ± 0.0935	0.0533 ± 0.0649	0.0667 ± 0.0632	0.0600 ± 0.0655
teaching-ae	0.5099 ± 0.1711	0.5298 ± 0.1579	0.5166 ± 0.1668	0.4503 ± 0.0985	0.4901 ± 0.1245	0.4834 ± 0.1179
hepatitis	0.2065 ± 0.1228	0.1613 ± 0.1151	0.1935 ± 0.1198	0.1806 ± 0.1058	0.1484 ± 0.1280	0.1935 ± 0.1244
wine	0.0225 ± 0.0347	0.0225 ± 0.0347	0.0281 ± 0.0404	0.0449 ± 0.0516	0.0618 ± 0.0643	0.0281 ± 0.0404
autos	0.3317 ± 0.1177	0.3902 ± 0.1648	0.3415 ± 0.1409	0.2195 ± 0.0786	0.2293 ± 0.1374	0.2537 ± 0.1104
sonar	0.2548 ± 0.1122	0.2452 ± 0.0889	0.2452 ± 0.1305	0.1731 ± 0.1069	0.2788 ± 0.0840	0.1394 ± 0.0888
glass-id	0.2570 ± 0.0978	0.2570 ± 0.1019	0.2664 ± 0.0843	0.2804 ± 0.0976	0.2617 ± 0.0944	0.1589 ± 0.0576
new-thyroid	0.0558 ± 0.0590	0.0419 ± 0.0487	0.0512 ± 0.0343	0.1116 ± 0.0704	0.0791 ± 0.0647	0.0512 ± 0.0544
audio	0.2566 ± 0.0864	0.2389 ± 0.0548	0.2212 ± 0.0799	0.1681 ± 0.0599	0.2920 ± 0.0926	0.2301 ± 0.0649
hungarian	0.1871 ± 0.0862	0.1565 ± 0.0698	0.1599 ± 0.0779	0.2041 ± 0.0785	0.1973 ± 0.0606	0.1429 ± 0.0676
heart-disease-c	0.1716 ± 0.0880	0.1683 ± 0.0803	0.1617 ± 0.0681	0.2013 ± 0.1005	0.2112 ± 0.1005	0.1848 ± 0.1067
haberman	0.2582 ± 0.1155	0.2647 ± 0.1285	0.2843 ± 0.1303	0.2712 ± 0.1294	0.2843 ± 0.1023	0.2712 ± 0.1188
primary-tumor	0.5133 ± 0.0897	0.5162 ± 0.0883	0.5221 ± 0.0719	0.5723 ± 0.0826	0.5752 ± 0.0960	0.5162 ± 0.0984
ionosphere	0.1368 ± 0.0793	0.1197 ± 0.0854	0.1140 ± 0.0896	0.1282 ± 0.0731	0.0684 ± 0.0510	0.0826 ± 0.0405
dermatology	0.0246 ± 0.0248	0.0191 ± 0.0242	0.0109 ± 0.0180	0.0464 ± 0.0222	0.0464 ± 0.0390	0.0219 ± 0.0275
horse-colic	0.1630 ± 0.0665	0.2065 ± 0.0928	0.1902 ± 0.0678	0.2527 ± 0.0782	0.2092 ± 0.0629	0.2038 ± 0.0590
house-votes-84	0.0437 ± 0.0353	0.0943 ± 0.0256	0.0690 ± 0.0330	0.0598 ± 0.0406	0.0552 ± 0.0375	0.0529 ± 0.0346
cylinder-bands	0.2037 ± 0.0262	0.2093 ± 0.0326	0.2444 ± 0.0343	0.1593 ± 0.0301	0.3296 ± 0.0719	0.1611 ± 0.0421
chess	0.1143 ± 0.0634	0.1125 ± 0.0551	0.1906 ± 0.0403	0.1053 ± 0.0407	0.0926 ± 0.0492	0.1053 ± 0.0631
syncon	0.0483 ± 0.0398	0.0483 ± 0.0398	0.0733 ± 0.0471	0.0150 ± 0.0097	0.0300 ± 0.0249	0.0200 ± 0.0163
balance-scale	0.0832 ± 0.0207	0.0832 ± 0.0207	0.0896 ± 0.0148	0.0160 ± 0.0155	0.1328 ± 0.0156	0.1120 ± 0.0159
soybean	0.0952 ± 0.0267	0.0893 ± 0.0244	0.1010 ± 0.0383	0.0630 ± 0.0251	0.0469 ± 0.0136	0.0542 ± 0.0184
credit-a	0.1391 ± 0.0344	0.1449 ± 0.0303	0.1449 ± 0.0308	0.1449 ± 0.0323	0.1696 ± 0.0370	0.1261 ± 0.0210
breast-cancer-w	0.0300 ± 0.0204	0.0258 ± 0.0223	0.0272 ± 0.0217	0.0501 ± 0.0261	0.0415 ± 0.0273	0.0386 ± 0.0248
pima-ind-diabetes	0.2500 ± 0.0808	0.2591 ± 0.0707	0.2487 ± 0.0631	0.2435 ± 0.0748	0.2526 ± 0.0509	0.2513 ± 0.0636
vehicle	0.4090 ± 0.0477	0.4090 ± 0.0477	0.4551 ± 0.0540	0.3050 ± 0.0518	0.2837 ± 0.0603	0.3132 ± 0.0563
anneal	0.1013 ± 0.0274	0.0891 ± 0.0261	0.1114 ± 0.0306	0.0301 ± 0.0137	0.0468 ± 0.0182	0.0735 ± 0.0232
tic-tac-toe	0.2683 ± 0.0277	0.3069 ± 0.0427	0.3006 ± 0.0361	0.0177 ± 0.0115	0.2286 ± 0.0395	0.2683 ± 0.0432
vowel	0.3586 ± 0.0564	0.4061 ± 0.0557	0.4030 ± 0.0357	0.2323 ± 0.0423	0.0667 ± 0.0259	0.0808 ± 0.0296
german	0.2490 ± 0.0376	0.2520 ± 0.0325	0.3070 ± 0.0565	0.2380 ± 0.0495	0.2700 ± 0.0515	0.2410 ± 0.0535
led	0.2670 ± 0.0622	0.2670 ± 0.0622	0.2650 ± 0.0650	0.2620 ± 0.0639	0.2660 ± 0.0569	0.2700 ± 0.0604
contraceptive-mc	0.4793 ± 0.0363	0.4949 ± 0.0534	0.5044 ± 0.0468	0.4637 ± 0.0528	0.4739 ± 0.0345	0.4671 ± 0.0455
yeast	0.4245 ± 0.0504	0.4245 ± 0.0504	0.4575 ± 0.0277	0.4252 ± 0.0504	0.4481 ± 0.0360	0.4205 ± 0.0402
volcanoes	0.3362 ± 0.0287	0.3421 ± 0.0278	0.3770 ± 0.0331	0.3441 ± 0.0304	0.3559 ± 0.0250	0.3539 ± 0.0331
car	0.1400 ± 0.0255	0.1400 ± 0.0255	0.1968 ± 0.0256	0.0671 ± 0.0123	0.0567 ± 0.0182	0.0845 ± 0.0193
segment	0.1476 ± 0.0162	0.1476 ± 0.0245	0.1398 ± 0.0191	0.0775 ± 0.0169	0.0615 ± 0.0142	0.0563 ± 0.0091
hypothyroid	0.0452 ± 0.0051	0.0360 ± 0.0112	0.0496 ± 0.0082	0.0313 ± 0.0109	0.0332 ± 0.0126	0.0348 ± 0.0118
splice-c4.5	0.0371 ± 0.0107	0.0444 ± 0.0112	0.0579 ± 0.0104	0.0718 ± 0.0078	0.0466 ± 0.0129	0.0375 ± 0.0087
kr-vs-kp	0.1139 ± 0.0256	0.1214 ± 0.0217	0.1017 ± 0.0178	0.0257 ± 0.0109	0.0776 ± 0.0228	0.0854 ± 0.0187
abalone	0.4728 ± 0.0209	0.4893 ± 0.0249	0.4893 ± 0.0242	0.4613 ± 0.0148	0.4692 ± 0.0285	0.4551 ± 0.0214
spambase	0.0943 ± 0.0104	0.1050 ± 0.0149	0.0922 ± 0.0133	0.0574 ± 0.0115	0.0696 ± 0.0106	0.0635 ± 0.0114
phoneme	0.2356 ± 0.0137	0.2615 ± 0.0129	0.2834 ± 0.0147	0.2850 ± 0.0237	0.2733 ± 0.0177	0.2100 ± 0.0144
wall-following	0.1135 ± 0.0110	0.1743 ± 0.0149	0.1378 ± 0.0136	0.0565 ± 0.0104	0.1147 ± 0.0116	0.1514 ± 0.0101
page-blocks	0.0691 ± 0.0091	0.1376 ± 0.0126	0.1162 ± 0.0114	0.0378 ± 0.0121	0.0541 ± 0.0100	0.0502 ± 0.0066
optdigits	0.0893 ± 0.0123	0.0861 ± 0.0124	0.0879 ± 0.0098	0.0413 ± 0.0097	0.0438 ± 0.0064	0.0283 ± 0.0095
satellite	0.2054 ± 0.0179	0.2022 ± 0.0168	0.2036 ± 0.0178	0.1414 ± 0.0214	0.1310 ± 0.0126	0.1301 ± 0.0131
musk2	0.1458 ± 0.0133	0.2496 ± 0.0101	0.2113 ± 0.0110	0.0308 ± 0.0091	0.0917 ± 0.0086	0.1511 ± 0.0101
mushrooms	0.0059 ± 0.0021	0.0196 ± 0.0036	0.0098 ± 0.0027	0.0000 ± 0.0000	0.0001 ± 0.0004	0.0002 ± 0.0005
thyroid	0.2661 ± 0.0130	0.2754 ± 0.0152	0.2900 ± 0.0111	0.2128 ± 0.0121	0.2294 ± 0.0111	0.2421 ± 0.0136
pendigits	0.1447 ± 0.0112	0.1447 ± 0.0112	0.1563 ± 0.0087	0.0404 ± 0.0080	0.0576 ± 0.0064	0.0254 ± 0.0029
sign	0.3851 ± 0.0114	0.3851 ± 0.0114	0.4084 ± 0.0083	0.3574 ± 0.0069	0.2853 ± 0.0094	0.2960 ± 0.0119
nursery	0.0973 ± 0.0066	0.0973 ± 0.0066	0.3674 ± 0.0182	0.0745 ± 0.0066	0.0654 ± 0.0062	0.0733 ± 0.0059
magic	0.2330 ± 0.0092	0.2478 ± 0.0118	0.2510 ± 0.0095	0.1683 ± 0.0082	0.1613 ± 0.0076	0.1726 ± 0.0084
letter-recog	0.3146 ± 0.0100	0.3226 ± 0.0110	0.3177 ± 0.0112	0.2008 ± 0.0084	0.1941 ± 0.0085	0.1514 ± 0.0089
adult	0.1758 ± 0.0048	0.1809 ± 0.0050	0.2377 ± 0.0044	0.1504 ± 0.0032	0.1641 ± 0.0037	0.1679 ± 0.0032
shuttle	0.0291 ± 0.0030	0.0311 ± 0.0022	0.1442 ± 0.0135	0.0088 ± 0.0012	0.0097 ± 0.0013	0.0101 ± 0.0010
connect-4	0.2779 ± 0.0055	0.2783 ± 0.0059	0.2875 ± 0.0071	0.2427 ± 0.0045	0.2354 ± 0.0050	0.2422 ± 0.0047

5.3. Effect of attribute selection

Different model in selective naïve Bayes implies different number of attributes that will be used when classifying an example. As a result, model selection in selective naïve Bayes actually selects different number of attributes. This section will examine how many attributes have been selected on different data sets to demonstrate the effect of attributes selection.

In the 10-fold cross validation experiment, number of attributes selected in different fold might also be different. Consequently, it is reasonable to report 10 numbers of attributes of all folds on one data set. At the same time, it is cumbersome to report these numbers on all 65 data sets. So we report the numbers of attributes of all folds on only the biggest 4 data sets, letter-recognition, adult, shuttle, and connect-4, as in Fig. 7. In

Table A.9
RMSE.

Data set	SNB	NB	GRFW	WANBIA	TAN	AODE
contact-lenses	0.4594 ± 0.2218	0.5017 ± 0.2450	0.4263 ± 0.2643	0.5682 ± 0.3842	0.6077 ± 0.1831	0.5226 ± 0.2221
lung-cancer	0.5912 ± 0.2866	0.6431 ± 0.2364	0.5821 ± 0.2903	0.7136 ± 0.2947	0.7623 ± 0.1357	0.6614 ± 0.2444
labor-negotiations	0.1856 ± 0.1226	0.1782 ± 0.1187	0.2509 ± 0.2067	0.3537 ± 0.2631	0.2935 ± 0.1975	0.2104 ± 0.1455
post-operative	0.5009 ± 0.1097	0.5103 ± 0.1033	0.5579 ± 0.1328	0.5257 ± 0.1334	0.5340 ± 0.1393	0.5136 ± 0.1059
zoo	0.1573 ± 0.1017	0.1623 ± 0.0953	0.1462 ± 0.0970	0.1744 ± 0.1423	0.1309 ± 0.1131	0.1344 ± 0.0935
promoters	0.2206 ± 0.1184	0.2544 ± 0.1247	0.2624 ± 0.1675	0.2297 ± 0.1452	0.3264 ± 0.1659	0.2795 ± 0.0940
echocardiogram	0.4923 ± 0.0569	0.4564 ± 0.0878	0.4705 ± 0.0897	0.4774 ± 0.0871	0.5276 ± 0.1017	0.4829 ± 0.0808
lymphography	0.3523 ± 0.1472	0.3465 ± 0.1437	0.3897 ± 0.1366	0.4025 ± 0.1963	0.3813 ± 0.1227	0.3274 ± 0.1395
iris	0.2538 ± 0.1337	0.2489 ± 0.1331	0.2661 ± 0.1484	0.2314 ± 0.1668	0.2211 ± 0.1353	0.2224 ± 0.1303
teaching-ae	0.6297 ± 0.0855	0.6364 ± 0.0891	0.6358 ± 0.0996	0.6156 ± 0.0447	0.6189 ± 0.0671	0.6105 ± 0.0684
hepatitis	0.4054 ± 0.1321	0.3746 ± 0.1284	0.4138 ± 0.1292	0.4191 ± 0.1167	0.3434 ± 0.1479	0.3711 ± 0.1079
wine	0.1283 ± 0.0858	0.1283 ± 0.0858	0.1427 ± 0.0919	0.1947 ± 0.1337	0.2026 ± 0.1223	0.1528 ± 0.1007
autos	0.5350 ± 0.0952	0.5830 ± 0.1148	0.5536 ± 0.1253	0.4633 ± 0.0769	0.4725 ± 0.1291	0.4760 ± 0.1102
sonar	0.4600 ± 0.1118	0.4628 ± 0.0874	0.4798 ± 0.1229	0.4124 ± 0.1680	0.4856 ± 0.0890	0.3349 ± 0.1109
glass-id	0.4485 ± 0.0821	0.4465 ± 0.0809	0.4572 ± 0.0667	0.4963 ± 0.0988	0.4360 ± 0.0585	0.3654 ± 0.0546
new-thyroid	0.2193 ± 0.1029	0.2078 ± 0.0994	0.2089 ± 0.0663	0.3305 ± 0.0979	0.2554 ± 0.0991	0.2221 ± 0.0850
audio	0.4760 ± 0.0745	0.4665 ± 0.0595	0.4371 ± 0.0605	0.4091 ± 0.0696	0.5212 ± 0.0855	0.4639 ± 0.0606
hungarian	0.3798 ± 0.0699	0.3702 ± 0.0857	0.3923 ± 0.1034	0.3763 ± 0.0808	0.3895 ± 0.0711	0.3506 ± 0.0845
heart-disease-c	0.3699 ± 0.0846	0.3664 ± 0.0878	0.3820 ± 0.0837	0.3831 ± 0.0927	0.4177 ± 0.0861	0.3605 ± 0.0844
haberman	0.4254 ± 0.0809	0.4220 ± 0.0907	0.4471 ± 0.0924	0.4252 ± 0.0941	0.4433 ± 0.0759	0.4402 ± 0.0820
primary-tumor	0.6972 ± 0.0604	0.6970 ± 0.0591	0.7027 ± 0.0513	0.7361 ± 0.0546	0.7280 ± 0.0579	0.6972 ± 0.0585
ionosphere	0.3248 ± 0.1179	0.3257 ± 0.1351	0.3307 ± 0.1681	0.3535 ± 0.1232	0.2573 ± 0.1077	0.2841 ± 0.0724
dermatology	0.1388 ± 0.0504	0.1103 ± 0.0608	0.1044 ± 0.0452	0.2059 ± 0.0558	0.1826 ± 0.0695	0.1145 ± 0.0617
horse-colic	0.3627 ± 0.0561	0.4169 ± 0.0733	0.4274 ± 0.0823	0.4671 ± 0.0823	0.4289 ± 0.0672	0.4029 ± 0.0709
house-votes-84	0.2052 ± 0.0705	0.2997 ± 0.0428	0.2483 ± 0.0627	0.2405 ± 0.1029	0.2181 ± 0.0792	0.2016 ± 0.0736
cylinder-bands	0.4047 ± 0.0211	0.4225 ± 0.0325	0.4767 ± 0.0276	0.3783 ± 0.0455	0.4405 ± 0.0420	0.3656 ± 0.0451
chess	0.2979 ± 0.0515	0.2944 ± 0.0509	0.3616 ± 0.0449	0.2732 ± 0.0540	0.2594 ± 0.0470	0.2855 ± 0.0485
syncon	0.2077 ± 0.0729	0.2075 ± 0.0728	0.2573 ± 0.0787	0.1145 ± 0.0420	0.1602 ± 0.0688	0.1287 ± 0.0448
balance-scale	0.3905 ± 0.0202	0.3905 ± 0.0202	0.3952 ± 0.0174	0.1208 ± 0.0735	0.3971 ± 0.0186	0.3999 ± 0.0234
soybean	0.2943 ± 0.0419	0.2945 ± 0.0430	0.3072 ± 0.0504	0.2276 ± 0.0436	0.2014 ± 0.0341	0.2224 ± 0.0402
credit-a	0.3314 ± 0.0458	0.3321 ± 0.0454	0.3779 ± 0.0409	0.3299 ± 0.0408	0.3704 ± 0.0443	0.3164 ± 0.0387
breast-cancer-w	0.1625 ± 0.0894	0.1570 ± 0.0950	0.1593 ± 0.0946	0.2198 ± 0.0505	0.1928 ± 0.0618	0.1778 ± 0.0879
pima-ind-diabetes	0.4039 ± 0.0491	0.4217 ± 0.0459	0.4320 ± 0.0658	0.4030 ± 0.0444	0.4225 ± 0.0442	0.4071 ± 0.0438
vehicle	0.5971 ± 0.0388	0.5971 ± 0.0388	0.6280 ± 0.0396	0.4792 ± 0.0463	0.4638 ± 0.0458	0.4653 ± 0.0343
anneal	0.2472 ± 0.0325	0.2769 ± 0.0400	0.3142 ± 0.0460	0.1565 ± 0.0389	0.1813 ± 0.0366	0.2311 ± 0.0373
tic-tac-toe	0.4216 ± 0.0157	0.4309 ± 0.0218	0.5201 ± 0.0356	0.1297 ± 0.0315	0.4023 ± 0.0269	0.3995 ± 0.0212
vowel	0.5621 ± 0.0330	0.5905 ± 0.0343	0.5847 ± 0.0226	0.4778 ± 0.0429	0.2316 ± 0.0407	0.2593 ± 0.0347
german	0.4172 ± 0.0273	0.4162 ± 0.0281	0.5002 ± 0.0561	0.4140 ± 0.0339	0.4389 ± 0.0476	0.4147 ± 0.0305
led	0.4945 ± 0.0365	0.4945 ± 0.0365	0.4961 ± 0.0387	0.4937 ± 0.0403	0.5000 ± 0.0376	0.4970 ± 0.0364
contraceptive-mc	0.5998 ± 0.0150	0.6052 ± 0.0254	0.6126 ± 0.0302	0.5896 ± 0.0171	0.5955 ± 0.0148	0.5938 ± 0.0183
yeast	0.6092 ± 0.0198	0.6092 ± 0.0198	0.6406 ± 0.0123	0.6100 ± 0.0195	0.6204 ± 0.0226	0.6063 ± 0.0195
volcanoes	0.5280 ± 0.0154	0.5298 ± 0.0182	0.5344 ± 0.0208	0.5292 ± 0.0180	0.5313 ± 0.0155	0.5284 ± 0.0184
car	0.3395 ± 0.0142	0.3395 ± 0.0142	0.3760 ± 0.0241	0.2137 ± 0.0260	0.2405 ± 0.0171	0.3065 ± 0.0151
segment	0.3499 ± 0.0244	0.3492 ± 0.0323	0.3483 ± 0.0257	0.2625 ± 0.0302	0.2215 ± 0.0255	0.2069 ± 0.0143
hypothyroid	0.1784 ± 0.0166	0.1820 ± 0.0275	0.2224 ± 0.0185	0.1453 ± 0.0221	0.1528 ± 0.0262	0.1636 ± 0.0277
splice-c4.5	0.1759 ± 0.0182	0.1883 ± 0.0213	0.2328 ± 0.0206	0.2670 ± 0.0155	0.1917 ± 0.0248	0.1720 ± 0.0211
kr-vs-kp	0.2894 ± 0.0136	0.3022 ± 0.0163	0.2771 ± 0.0264	0.1493 ± 0.0212	0.2358 ± 0.0223	0.2658 ± 0.0155
abalone	0.5674 ± 0.0117	0.5896 ± 0.0180	0.5906 ± 0.0181	0.5620 ± 0.0062	0.5635 ± 0.0080	0.5576 ± 0.0077
spambase	0.2814 ± 0.0175	0.3042 ± 0.0192	0.2875 ± 0.0212	0.2103 ± 0.0140	0.2377 ± 0.0187	0.2282 ± 0.0212
phoneme	0.4496 ± 0.0084	0.4792 ± 0.0109	0.4891 ± 0.0147	0.5251 ± 0.0237	0.5048 ± 0.0133	0.4397 ± 0.0123
wall-following	0.2937 ± 0.0162	0.3944 ± 0.0153	0.3541 ± 0.0161	0.2155 ± 0.0176	0.3113 ± 0.0146	0.3677 ± 0.0136
page-blocks	0.2356 ± 0.0142	0.3421 ± 0.0156	0.3042 ± 0.0154	0.1752 ± 0.0223	0.2127 ± 0.0211	0.2024 ± 0.0110
optdigits	0.2800 ± 0.0154	0.2778 ± 0.0163	0.2816 ± 0.0169	0.1987 ± 0.0232	0.1919 ± 0.0125	0.1542 ± 0.0236
satellite	0.4417 ± 0.0199	0.4402 ± 0.0200	0.4420 ± 0.0196	0.3444 ± 0.0248	0.3396 ± 0.0173	0.3307 ± 0.0161
musk2	0.3159 ± 0.0156	0.4959 ± 0.0099	0.4488 ± 0.0126	0.1726 ± 0.0238	0.2946 ± 0.0144	0.3837 ± 0.0115
mushrooms	0.0786 ± 0.0120	0.1229 ± 0.0124	0.0980 ± 0.0139	0.0000 ± 0.0000	0.0083 ± 0.0082	0.0112 ± 0.0098
thyroid	0.4580 ± 0.0098	0.4701 ± 0.0116	0.4820 ± 0.0098	0.4007 ± 0.0083	0.4156 ± 0.0103	0.4334 ± 0.0109
pendigits	0.3590 ± 0.0113	0.3590 ± 0.0113	0.3751 ± 0.0112	0.1868 ± 0.0184	0.2140 ± 0.0130	0.1420 ± 0.0047
sign	0.5485 ± 0.0053	0.5485 ± 0.0053	0.5655 ± 0.0063	0.5273 ± 0.0049	0.4736 ± 0.0058	0.4835 ± 0.0042
nursery	0.2820 ± 0.0052	0.2820 ± 0.0052	0.4501 ± 0.0139	0.2299 ± 0.0096	0.2194 ± 0.0068	0.2510 ± 0.0047
magic	0.3933 ± 0.0065	0.4098 ± 0.0086	0.4616 ± 0.0103	0.3502 ± 0.0076	0.3437 ± 0.0068	0.3505 ± 0.0079
letter-recog	0.5346 ± 0.0084	0.5376 ± 0.0088	0.5348 ± 0.0085	0.4340 ± 0.0078	0.4120 ± 0.0085	0.3755 ± 0.0092
adult	0.3569 ± 0.0045	0.3657 ± 0.0047	0.4299 ± 0.0051	0.3229 ± 0.0028	0.3354 ± 0.0040	0.3476 ± 0.0035
shuttle	0.1618 ± 0.0041	0.1632 ± 0.0035	0.3237 ± 0.0088	0.0837 ± 0.0052	0.0907 ± 0.0046	0.0944 ± 0.0033
connect-4	0.4786 ± 0.0026	0.4787 ± 0.0026	0.4858 ± 0.0039	0.4479 ± 0.0024	0.4435 ± 0.0031	0.4506 ± 0.0018

the bar graph of Fig. 7, bars of different heights show different numbers of attributes selected in each fold on the data set. The above horizontal line represents the number of attributes that will be employed in normal NB on that data set.

We could find that on the data set letter-recognition, SNB selects 12 out of 16 attributes in all folds. On adult, 8 out of 14 attributes have been selected in all folds. On shuttle, SNB selects 6 out of 9 attributes except in the 6th fold. The 6th fold selects only 4 attributes. On connect-4, 42 out of 42 attributes have been selected in three folds. The rounded average number of attributes selected in the other 7 folds is 38. We could see that attribute selection indeed decreases the number of attributes used when classifying an example while increasing the prediction accuracy.

6. Conclusion and future work

This paper presents an efficient selective naïve Bayes algorithm that can perform attribute selection by model selection. Multiple models can be evaluated efficiently in incremental leave-one-out cross validation as these models are built such that each one is a trivial extension to another. Empirical results demonstrate the following characteristics of SNB:

- SNB is superior to NB and GRFW in terms of ZOL and RMSE at the cost of a modest increase in running time.
- SNB is comparable with TAN in terms of accuracy and running time.
- SNB achieves lower ZOL and RMSE significantly less often than both WANBIA and AODE. However, WANBIA and AODE's superior accuracy comes at considerable cost in increased running time.
- SNB decreases the number of attributes when classifying an example while increasing the prediction accuracy.

The superiority of SNB comes from the model selection in leave-one-out cross validation. So in the future, we will try to expand the model space further. Moreover, more efficient model selection techniques are also worthwhile to explore.

Acknowledgments

This research has been supported by the Natural Science Foundation of Jiangsu, China under grant BK20171495, the Humanities and Social Science Fund of Ministry of Education of China under grant 15YJAZH097, Qinglan Project of Jiangsu Province, China, Priority Academic Program Development of Jiangsu Higher Education Institutions, China. We also want to thank the anonymous reviewers for their insightful comments which have helped to greatly improve the revised version of the paper.

Appendix

Detailed results of zero–one loss and RMSE ± standard deviation are presented in Tables A.8 and A.9.

References

[1] X. Wu, V. Kumar, J.R. Quinlan, J. Ghosh, Q. Yang, H. Motoda, G.J. Mclachlan, A. Ng, B. Liu, P.S. Yu, Top 10 algorithms in data mining, Knowl. Inf. Syst. 14 (1) (2008) 1–37.

[2] R.O. Duda, P.E. Hart, Pattern Classification and Scene Analysis, John Wiley and Sons, 1973.

[3] L. Zhang, L. Jiang, C. Li, G. Kong, Two feature weighting approaches for naive bayes text classifiers, Knowl.-Based Syst. 100 (2016) 137–144, http://dx.doi.org/10.1016/j.knosys.2016.02.017.

[4] P. Langley, S. Sage, Induction of selective bayesian classifiers, in: Proceedings of the 10th International Conference on Uncertainty in Artificial Intelligence, Morgan Kaufmann Publishers Inc., MIT, Cambridge, MA, USA, 1994, pp. 399–406.

[5] P. Bermejo, J.A. Gmez, J.M. Puerta, Speeding up incremental wrapper feature subset selection with naive bayes classifier, Knowl.-Based Syst. 55 (2014) 140–147, http://dx.doi.org/10.1016/j.knosys.2013.10.016.

[6] H. Zhang, S. Sheng, Learning weighted naive bayes with accurate ranking, in: Proceedings of the Fourth IEEE International Conference on Data Mining, ICDM '04, IEEE Computer Society, Washington, DC, USA, 2004, pp. 567–570.

[7] M. Hall, A decision tree-based attribute weighting filter for naive bayes, Knowl.-Based Syst. 20 (2) (2007) 120–126, aI 2006.

[8] C.H. Lee, F. Gutierrez, D. Dou, Calculating feature weights in naive bayes with kullback-leibler measure, in: IEEE International Conference on Data Mining, 2012, pp. 1146–1151.

[9] C.H. Lee, A gradient approach for value weighted classification learning in naive bayes, Knowl.-Based Syst. 85 (2015) 71–79, http://dx.doi.org/10.1016/j.knosys.2015.04.020.

[10] C.H. Lee, An information-theoretic filter approach for value weighted classification learning in naive bayes, Data Knowl. Eng. 113 (2018) 116–128, http://dx.doi.org/10.1016/j.datak.2017.11.002.

[11] N.A. Zaidi, J. Cerquides, M.J. Carman, G.I. Webb, Alleviating naive bayes attribute independence assumption by attribute weighting, J. Mach. Learn. Res. 14 (1) (2013) 1947–1988.

[12] L. Jiang, L. Zhang, C. Li, J. Wu, A correlation-based feature weighting filter for naive bayes, IEEE Trans. Knowl. Data Eng. 31 (2) (2019) 201–213, http://dx.doi.org/10.1109/TKDE.2018.2836440.

[13] L. Yu, H. Liu, Feature selection for high-dimensional data: A fast correlation-based filter solution, in: Proceedings of the Twentieth International Conference on International Conference on Machine Learning, ICML'03, AAAI Press, 2003, pp. 856–863.

[14] L. Jiang, Z. Cai, H. Zhang, D. Wang, Not so greedy: Randomly selected naive bayes, Expert Syst. Appl. 39 (12) (2012) 11022–11028, http://dx.doi.org/10.1016/j.eswa.2012.03.022.

[15] M.A. Hall, Correlation-based feature selection for discrete and numeric class machine learning, in: Seventeenth International Conference on Machine Learning, 2000.

[16] M.J. Flores, J.A. Gámez, A.M. Martínez, Domains of competence of the semi-naive bayesian network classifiers, Inform. Sci. 260 (1) (2014) 120–148.

[17] N. Friedman, D. Geiger, M. Goldszmidt, Bayesian network classifiers, Mach. Learn. 29 (2–3) (1997) 131–163.

[18] C. Chow, C. Liu, Approximating discrete probability distributions with dependence trees, Inf. Theory IEEE Trans. 14 (3) (1968) 462–467.

[19] G.I. Webb, J.R. Boughton, Z. Wang, Not so naive bayes: Aggregating one-dependence estimators, Mach. Learn. 58 (1) (2005) 5–24.

[20] G.I. Webb, J.R. Boughton, F. Zheng, K.M. Ting, H. Salem, Learning by extrapolation from marginal to full-multivariate probability distributions: Decreasingly naive bayesian classification, Mach. Learn. 86 (2) (2012) 233–272.

[21] L. Yu, L. Jiang, D. Wang, L. Zhang, Attribute value weighted average of one-dependence estimators, Entropy 19 (9) (2017) 501.

[22] M. Sahami, Learning limited dependence bayesian classifiers, in: Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining, ACM, New York, NY, 1996, pp. 335–338.

[23] H. Langseth, T.D. Nielsen, Classification using hierarchical naive bayes models, Mach. Learn. 63 (2) (2006) 135–159.

[24] L. Jiang, H. Zhang, Z. Cai, A novel bayes model: Hidden naive bayes, IEEE Trans. Knowl. Data Eng. 21 (10) (2009) 1361–1371, http://dx.doi.org/10.1109/TKDE.2008.234.

[25] H. Zhang, L. Jiang, J. Su, Hidden naive bayes, in: Proceedings of the 20th National Conference on Artificial Intelligence - Volume 2, AAAI'05, AAAI Press, 2005, pp. 919–924.

[26] L. Jiang, Random one-dependence estimators, Pattern Recognit. Lett. 32 (3) (2011) 532–539.

[27] N.S. Harzevili, S.H. Alizadeh, Mixture of latent multinomial naive bayes classifier, Appl. Soft Comput. 69 (2018) 516–527, http://dx.doi.org/10.1016/j.asoc.2018.04.020.

[28] C. Qiu, L. Jiang, C. Li, Not always simple classification: Learning superparent for class probability estimation, Expert Syst. Appl. 42 (13) (2015) 5433–5440.

[29] E.J. Keogh, M.J. Pazzani, Learning augmented bayesian classifiers: A comparison of distribution-based and classification-based approaches, in: 7th International Workshop on Artificial Intelligence and Statistics, PMLR, Stockholmsmässan, Stockholm Sweden, 1999, pp. 225–230.

[30] U.M. Fayyad, K.B. Irani, Multi-interval discretization of continuous-valued attributes for classification learning, in: Proceedings of the 13th International Joint Conference on Artificial Intelligence, 1993, pp. 1022–1027.

[31] L.M. Wang, X.L. Li, C.H. Cao, S.M. Yuan, Combining decision tree and naive bayes for classification, Knowl.-Based Syst. 19 (7) (2006) 511–515.

[32] R.R. Bouckaert, Naive bayes classifiers that perform well with continuous variables, in: G.I. Webb, X. Yu (Eds.), AI 2004: Advances in Artificial Intelligence, Springer Berlin Heidelberg, Berlin, Heidelberg, 2005, pp. 1089–1094.

[33] G.H. John, P. Langley, Estimating continuous distributions in bayesian classifiers, in: Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence, UAI'95, Morgan Kaufmann Publishers Inc, San Francisco, CA, USA, 1995, pp. 338–345.

[34] B. Liu, Y. Yang, G.I. Webb, J. Boughton, A comparative study of bandwidth choice in kernel density estimation for naive bayesian classification, in: T. Theeramunkong, B. Kijsirikul, N. Cercone, T.-B. Ho (Eds.), Advances in Knowledge Discovery and Data Mining, Springer Berlin Heidelberg, Berlin, Heidelberg, 2009, pp. 302–313.

[35] D. Soria, J.M. Garibaldi, F. Ambrogi, E.M. Biganzoli, I.O. Ellis, A non-parametric version of the naive bayes classifier, Knowl.-Based Syst. 24 (6) (2011) 775–784, http://dx.doi.org/10.1016/j.knosys.2011.02.014.

[36] X. Wang, Y. He, D.D. Wang, Non-naive bayesian classifiers for classification problems with continuous attributes, IEEE Trans. Cybern. 44 (1) (2014) 21–39, http://dx.doi.org/10.1109/TCYB.2013.2245891.

[37] D. Koller, N. Friedman, Probabilistic Graphical Models - Principles and Techniques, The MIT Press, Cambridge, 2009.

[38] B. Cestnik, Estimating Probabilities: A Crucial Task in Machine Learning, 1990, pp. 147–149.

[39] I. Rish, An empirical study of the naive bayes classifier, in: IJCAI 2001 Workshop on Empirical Methods in Artificial Intelligence, Vol. 3, 2001, pp. 41–46.

[40] D.J. MacKay, Information Theory, Inference and Learning Algorithms, Cambridge university press, 2003.

[41] R. Kohavi, The power of decision tables, in: N. Lavrac, S. Wrobel (Eds.), ECML, Springer, 1995, pp. 174–189.

[42] D. Dua, C. Graff, UCI machine learning repository, 2017, URL http://archive.ics.uci.edu/ml.

[43] S. Chen, A.M. Martínez, G.I. Webb, L. Wang, Selective ande for large data learning: a low-bias memory constrained approach, Knowl. Inf. Syst. 50 (2) (2017) 475–503, http://dx.doi.org/10.1007/s10115-016-0937-9.

[44] J. Demšar, Statistical comparisons of classifiers over multiple data sets, J. Mach. Learn. Res. 7 (2006) 1–30.

[45] S. García, F. Herrera, An extension on statistical comparisons of classifiers over multiple data sets for all pairwise comparisons, J. Mach. Learn. Res. 9 (12) (2008) 2677–2694.