HHS Public Access

Author manuscript

Intern J Pattern Recognit Artif Intell. Author manuscript; available in PMC 2016 April 15.

Published in final edited form as:

Intern J Pattern Recognit Artif Intell. 2015 October 1; 29(7): . doi:10.1142/S0218001415500238.

A Pairwise Naïve Bayes Approach to Bayesian Classification

Josephine K. Asafu-Adjei and

Department of Biostatistics, University of North Carolina at Chapel Hill, 3104-E McGavran-Greenberg Hall, CB 7420, Chapel Hill, NC 27599, USA, jasafuad@email.unc.edu

Rebecca A. Betensky

Department of Biostatistics, Harvard School of Public Health, 655 Huntington Avenue, Building 2, 4th Floor, Boston, MA 02115, USA, betensky@hsph.harvard.edu

Abstract

Despite the relatively high accuracy of the naïve Bayes (NB) classifier, there may be several instances where it is not optimal, i.e. does not have the same classification performance as the Bayes classifier utilizing the joint distribution of the examined attributes. However, the Bayes classifier can be computationally intractable due to its required knowledge of the joint distribution. Therefore, we introduce a "pairwise naïve" Bayes (PNB) classifier that incorporates all pairwise relationships among the examined attributes, but does not require specification of the joint distribution. In this paper, we first describe the necessary and sufficient conditions under which the PNB classifier is optimal. We then discuss sufficient conditions for which the PNB classifier, and not NB, is optimal for normal attributes. Through simulation and actual studies, we evaluate the performance of our proposed classifier relative to the Bayes and NB classifiers, along with the HNB, AODE, LBR and TAN classifiers, using normal density and empirical estimation methods. Our applications show that the PNB classifier using normal density estimation yields the highest accuracy for data sets containing continuous attributes. We conclude that it offers a useful compromise between the Bayes and NB classifiers.

Keywords

Bayesian classification; naïve Bayes classifier; optimal classification; pairwise naïve Bayes classifier; semi-naïve Bayes classifier

1. Introduction

Consider a set of P attributes $\mathbf{x} = (x_1, \ldots, x_P)$ measured for an individual belonging to class c_i ($i = 1, \ldots, m$; $m \geq 2$). Suppose c_i is unknown, so that some rule based on $\mathbf{x}$ is needed to classify the individual. Naturally, it is desired that this rule be as accurate as possible. Under zero-one loss (unit cost of misclassification and zero cost of correct classification), one such rule is the Bayes classification rule, which has the smallest expected loss among all other classification rules.[1,6] The Bayes rule classifies an individual with observation $\mathbf{x}$ into class c_i such that

Correspondence to: Josephine K. Asafu-Adjei.

$$c_i = \text{argmax}_i P(C=c_i|\mathbf{X}=\mathbf{x}) = \text{argmax}_i \left[\frac{P(\mathbf{X}=\mathbf{x}, C=c_i)}{P(\mathbf{X}=\mathbf{x})}\right] = \text{argmax}_i \left[\frac{\pi_i f_i(\mathbf{x})}{f(\mathbf{x})}\right], \quad (1)$$

where C denotes the class variable, π_i denotes the prior probability of $\mathbf{X}$ belonging to class c_i, $f_i(\mathbf{x})$ denotes the probability density/mass function of $\mathbf{X}$ in class c_i (assuming known π_i), and the overall probability density/mass function of $\mathbf{X}$ is $f(\mathbf{x}) = \sum_{i=1}^{m} \pi_i f_i(\mathbf{x})$. In this paper, we assume that the cost of misclassifying an individual into class c_i is the same for all classes.

For $m = 2$ classes, the Bayes rule classifies $\mathbf{x}$ into class c_1 if and only if the Bayes classifier

$$\phi_b(\mathbf{x}) = \frac{P(C=c_1|\mathbf{X}=\mathbf{x})}{P(C=c_2|\mathbf{X}=\mathbf{x})} = \frac{\pi_1 f_1(\mathbf{x})/f(\mathbf{x})}{\pi_2 f_2(\mathbf{x})/f(\mathbf{x})} = \frac{\pi_1 f_1(\mathbf{x})}{\pi_2 f_2(\mathbf{x})} \geq 1 \quad (2)$$

and into class c_2 otherwise, where assignment of $\mathbf{x}$ to class c_1 for $\varphi_b(\mathbf{x}) = 1$ is random.[1] A proof of why a rule based on $\varphi_b(\mathbf{x})$ is a Bayes rule can be found in Ref. 6. If we have available training and test samples that are representative of the population of ($\mathbf{X}$, c_i) values, there are several ways to estimate $\varphi_b(\mathbf{x})$. For continuous $\mathbf{X}$, one can assume a parametric, e.g. normal, distribution, and use the training sample to estimate the necessary parameters. For discrete $\mathbf{X}$, one can estimate $\varphi_b(\mathbf{x})$ using the sample probability of each $\mathbf{x}$ value in the training sample. However, a few issues may arise when computing $\varphi_b(\mathbf{x})$. First, estimating $\varphi_b(\mathbf{x})$ is likely to be problematic for high-dimensional data sets. Also, the sample probabilities used to estimate $\varphi_b(\mathbf{x})$ can be very small for moderate to large P and, thus, noninformative. Another issue arises when a test observation $\mathbf{x}$ does not occur in the training sample so that $f_i(\mathbf{x})$ is not estimable.

To help avoid these issues, the naïve Bayes (NB) classifier is often used, which assumes that all X_p are independent given membership in class c_i. For $m = 2$ classes, this classifier is given by

$$\phi_{\text{nb}}(\mathbf{x}) = \frac{\pi_1}{\pi_2} \prod_{p=1}^{P} \frac{f_1(x_p)}{f_2(x_p)}, \quad (3)$$

where $\mathbf{x}$ is classified into class c_1 if $\varphi_{\text{nb}}(\mathbf{x}) \geq 1$ and into class c_2 otherwise. For $m > 2$ classes, $f_i(\mathbf{x})$ in (1) is replaced with $\prod_{p=1}^{P} f_i(x_p)$. To estimate $\varphi_{\text{nb}}(\mathbf{x})$, one can use marginal density estimation for continuous $\mathbf{X}$ and empirical estimation for discrete $\mathbf{X}$. To avoid zero probability estimates, one can use correction methods, such as Laplace estimation.[31] Computing $\varphi_{\text{nb}}(\mathbf{x})$ and classifying a single observation has a total time complexity of $\mathcal{O}(tP) + \mathcal{O}(mP)$, where t is the number of training observations.[31]

Under zero-one loss, any two classifiers $\varphi_1(\mathbf{x})$ and $\varphi_2(\mathbf{x})$ are *equal*, i.e. $\varphi_1(\mathbf{x}) \doteq \varphi_2(\mathbf{x})$,[36] if

$$\phi_1(\mathbf{x}) \geq 0 \text{ if and only if } \phi_2(\mathbf{x}) \geq 0. \quad (4)$$

Since applying a logarithm to $\varphi_b(\mathbf{x})$ and $\varphi_{nb}(\mathbf{x})$ does not change their classification, $\varphi_{nb}(\mathbf{x})$ is *optimal*, i.e. $\varphi_{nb}(\mathbf{x}) \doteq \varphi_b(\mathbf{x})$,[8,35] if $\log(\varphi_{nb}(\mathbf{x}))$ and $\log(\varphi_b(\mathbf{x}))$ are the same sign.

Although the conditional independence assumption of $\varphi_{nb}(\mathbf{x})$ is usually unrealistic in practice, the NB classifier has been shown to perform surprisingly well.[8,9,13,36] Domingos and Pazzani[8] explain that this is due to the zero-one loss function, which does not penalize incorrect estimation of the posterior probability $P(C = c_i \mid \mathbf{X} = \mathbf{x})$ as long as this probability is highest for the correct class. Specifically, even though $\varphi_{nb}(\mathbf{x})$ can produce very poor estimates of $P(C = c_i \mid \mathbf{X} = \mathbf{x})$, the class with the highest posterior probability remains the same.[4,8,20,36]

While investigating the relatively high accuracy of $\varphi_{nb}(\mathbf{x})$, authors such as Kuncheva[18] and Zhang[36] determine the necessary and sufficient conditions for which $\varphi_{nb}(\mathbf{x})$ is optimal for $m = 2$, despite the strong relationships that may exist among the different attributes. When dealing with two binary attributes (X_1, X_2), where $X_1, X_2 \in \{0, 1\}$, Kuncheva demonstrates the optimality of $\varphi_{nb}(\mathbf{x})$ when $\pi_1 = \pi_2$ and the covariance of X_1 and X_2 is the same for both classes.[18] Zhang[36] goes beyond the binary case in determining the optimality conditions for $\varphi_{nb}(\mathbf{x})$, using the fact that $P(\mathbf{X} = \mathbf{x}, C = c_i)$ can be represented as an augmented naive Bayes (ANB) network. An ANB network is a directed acyclic graph where: (1) one node denotes the class variable C while each of the other nodes denotes an attribute X_p, (2) each directed edge points from one parent node to its descendant, and denotes a link between the two nodes, (3) each attribute node may have more than one parent, but one parent must be the class node C, and (4) given its parents, each attribute is independent of all other attributes.[9,23] Unlike an NB network, where each attribute node only has class node C as its parent, an ANB network represents the dependence that can exist between any attribute and its parents.[36] An example of an NB and ANB network is shown in Figs. 1(a) and 1(b).

The ANB representation of $P(\mathbf{X} = \mathbf{x}, C = c_i)$ is

$$P(\mathbf{X}=\mathbf{x}, C=c_i)=\pi_i \prod_{p=1}^{P} f(x_p|\mathrm{pa}(x_p), C=c_i) = \pi_i \prod_{p=1}^{P} f_i(x_p|\mathrm{pa}(x_p)), \quad (5)$$

where $\mathrm{pa}(x_p)$ denotes the values of the parent attributes of X_p.[9,15,26,36] We may want to represent $P(\mathbf{X} = \mathbf{x}, C = c_i)$ in a way that expresses the relationship between any *pair* (X_j, X_k) $(j, k = 1, \ldots, P, j \neq k)$ and another distinct set of attributes, which we denote by $\mathrm{pa}(X_j, X_k)$. In this case, we can use the following "pseudo" ANB representation of $P(\mathbf{X} = \mathbf{x}, C = c_i)$

$$P(\mathbf{X}=\mathbf{x}, C=c_i) = \begin{cases} \pi_i f_i(x_P) \prod\limits_{\substack{l=1 \\ l \text{ odd}}}^{P-2} f_i(x_l, x_{l+1}|\mathrm{pa}(x_l, x_{l+1})) & \text{for odd } P, \\ \pi_i f_i(x_{P-1}, x_P) \prod\limits_{\substack{l=1 \\ l \text{ odd}}}^{P-3} f_i(x_l, x_{l+1}|\mathrm{pa}(x_l, x_{l+1})) & \text{for even } P, \end{cases} \quad (6)$$

where $\mathrm{pa}(x_l, x_{l+1}) = \{x_{l+2}, \ldots, x_P\}$. The ordering in (6) and the definition of $\mathrm{pa}(x_l, x_{l+1})$ is used for notational convenience in the next section, but we note that this ordering is arbitrary

since the representation in (6) is not order-dependent. We define a "pseudo" ANB network as a set of attributes where (1) (X_j, X_k) is related to class and another set of attributes pa(X_j, X_k) and (2) given membership in class c_i and pa(X_j, X_k), each pair (X_j, X_k) is independent of all other attributes.

In his discussion, Zhang[36] uses (5) to show that $\varphi_b(\mathbf{x})$ is the product of $\varphi_{nb}(\mathbf{x})$ and a quantity that reflects the strength of the relationship between X_p and pa(X_p) in each class. He then uses this relation to develop the optimality conditions for $\varphi_{nb}(\mathbf{x})$. In the process, he proves that whether $\varphi_{nb}(\mathbf{x})$ is optimal depends on how the strength of the relationship between X_p and pa(X_p) compares across the two classes. Zhang[36] also establishes sufficient optimality conditions for $\varphi_{nb}(\mathbf{x})$ for bivariate normal data.

On the other hand, the optimality conditions of $\varphi_{nb}(\mathbf{x})$ may not be satisfied. For instance, Kuncheva[18] demonstrates that $\varphi_{nb}(\mathbf{x})$ is not optimal in the case of binary attributes when π_i is not the same for each class. If estimating $\varphi_b(\mathbf{x})$ is problematic, then an alternate classifier that falls between $\varphi_{nb}(\mathbf{x})$ and $\varphi_b(\mathbf{x})$ needs to be considered. This has been recognized in the literature by several authors in their proposals of such alternatives to account for the relationships among subsets of the P attributes (Chow and Liu,[7] Friedman *et al.*,[9] Grossman and Domingos,[11] Hall,[12] Keogh and Pazzani,[15,16] Kononenko,[17] Langley and Sage,[19] Pazzani,[22] Sahami,[26] Silvescu *et al.*,[27] Singh and Provan,[28] Webb *et al.*,[31] Webb and Pazzani,[32] Xie *et al.*,[33] Zaidi *et al.*,[34] Zhang,[35] Zhang *et al.*,[37,38] and Zheng and Webb[39] and Zheng *et al.*[40]).

Of these techniques aimed at relaxing the conditional independence assumption of $\varphi_{nb}(\mathbf{x})$, we focus on five that have demonstrated considerable improvement in classification accuracy relative to the NB classifier. Zheng and Webb[39] develop a lazy Bayesian rule (LBR) classifier that assigns $\mathbf{x}$ to the class c_i that maximizes the estimate $\hat{P}(C=c_i|\mathbf{X}_{sel})\prod_{p=1}^{P}\hat{P}(X_p=x_p|C=c_i,\mathbf{X}_{sel})$, where $\mathbf{X}_{sel}$ is a subset of attributes that is selected using a heuristic wrapper approach aimed at minimizing classification error.[39] Based on the Bayesian tree construction approach of Chow and Liu,[7] Friedman *et al.*[9] propose a tree augmented naïve (TAN) classifier that assigns $\mathbf{x}$ to the class c_i that maximizes the estimate $\hat{\pi}_i\prod_{p=1}^{P}\hat{P}(X_p=x_p|C=c_i,\mathrm{pa}(x_p))$, where $\hat{\pi_i}$ and $\hat{P}(\cdot)$ are computed using smoothing techniques and the function pa(x_p) is chosen using the conditional mutual information between X_p and X_j ($j \neq p$) given membership in class c_i. Keogh and Pazzani[15,16] also develop a variant of TAN called Super-Parent TAN (SP-TAN) where pa(x_p) is instead chosen using a heuristic wrapper approach aimed at minimizing classification error. With respect to accuracy, SP-TAN has been shown to outperform TAN and to be comparable to LBR.[15,16,30] In the spirit of the Bayesian network classification approach by Sahami,[26] which assumes that each X_p is dependent on class membership and at most s other attributes ($0 \leq s \leq P-1$), Webb *et al.*[31] develop their proposed classifier, based on averaged one-dependence estimation (AODE), which assigns $\mathbf{x}$ to the class c_i that maximizes the estimate

$$\sum_{(p:1\leq p\leq P)\wedge\hat{F}(x_p)\geq n_{\text{cut}}} \hat{P}(X_p=x_p, C=c_i)\prod_{j=1}^{P}\hat{P}(X_j=x_j|X_p=x_p, C=c_i), \quad (7)$$

where $\hat{P}(\cdot)$ are computed using Laplace estimation, $\hat{F}(x_p)$ denotes the number of training observations where attribute X_p is equal to the value x_p, and n_{cut} is set to 30 to chosen to ensure that $\hat{P}(\cdot)$ are based on adequate sample sizes. In addition, Zhang *et al.*[37] propose a hidden naïve Bayes (HNB) network, based on the hierarchical NB model of Zhang[35] that extends the NB network such that each attribute X_p not only has class node C as its parent, but also a hidden parent $X_{h,p}$. Based on this network, their proposed classifier assigns **x** to the class c_i that maximizes the estimate $\hat{\pi}_i\prod_{p=1}^{P}\hat{P}(X_p=x_p|C=c_i, X_{h,p}=x_{h,p})$, where $\hat{P}(X_p=x_p|C=c_i, X_{h,p}=x_{h,p})=\sum_{\substack{j=1\\ j\neq k}}^{P} W_{jk}\cdot\hat{P}(X_k=x_k|C=c_i, X_j=x_j)$ and the weights W_{jk} are computed using the conditional mutual information between X_j and X_k ($j\neq k$). The total time complexities (training time + classification time) involved in applying LBR, TAN, SP-TAN, AODE, and HNB are displayed in Table 1 (see Refs. 9, 15, 16, 31, 37 and 39 for details).

However, one potential issue for each of these approaches is that they all require discrete valued attributes. In particular, in applying their proposed classifiers for continuous **X**, Zheng and Webb, Friedman *et al.*, Keogh and Pazzani, Webb *et al.*, and Zhang *et al.* all discretize the attribute data using the entropy minimization approach by Fayyad and Irani[10] to partition the range of each X_p. In discretizing the data, important information may be lost and may lead to higher classification error for classifiers using sample probabilities relative to those using density estimates.

Therefore, to address these potential issues, we propose an alternate "pairwise naïve" (PNB) classifier that accounts for the relationships between all $\binom{P}{2}$ attribute pairs and is applicable for both discrete and continuous data. Our proposed classifier is given by

$$\phi_{\text{pnb}}(\mathbf{x})=\frac{\pi_1}{\pi_2}\prod_{j=1}^{P-1}\prod_{k=j+1}^{P}\frac{f_1(x_j,x_k)}{f_2(x_j,x_k)} \quad (8)$$

for $m = 2$, which classifies **x** into class c_1 if $\varphi_{\text{pnb}}(\mathbf{x}) \geq 1$ and into class c_2 otherwise. For $m >$ 2 classes, $f_i(\mathbf{x})$ in (1) is replaced with $\prod_{j=1}^{P-1}\prod_{k=j+1}^{P} f_i(x_j,x_k)$. We can then compute (8) using bivariate density or Laplace estimation, depending on **X**. Estimating the pairwise probabilities $f_i(x_j, x_k)$ based on t training observations is of time complexity $\mathcal{O}(tP^2)$, while classifying a single observation using the estimates $\prod_{j=1}^{P-1}\prod_{k=j+1}^{P} f_i(x_j,x_k)(i=1,\ldots,m)$ is of time complexity $\mathcal{O}(mP^2)$. Thus, the total time complexity involved in applying $\varphi_{\text{pnb}}(\mathbf{x})$ is $\mathcal{O}(tP^2) + \mathcal{O}(mP^2)$. The total computational time complexity for our proposed classifier is also displayed in Table 1.

It is reasonable to expect that any classifier which relaxes the conditional independence assumption of NB will yield higher classification accuracy relative to NB, since such a classifier should yield more precise estimates of the probabilities $\pi_i f_i(\mathbf{x})$ needed to compute $\varphi_b(\mathbf{x})$. Therefore, in proposing $\varphi_{pnb}(\mathbf{x})$, we could have extended beyond pairs of attributes and instead accounted for the relationship between attribute triples, quadruples, etc. However, in considering only relationships between attribute pairs, our proposed classifier minimizes the risk of encountering computational issues that may arise from taking into account the relationship between attributes in a particular subset. For instance, in the case of discrete valued attributes where X_p has at least two values, the sample probabilities used to estimate each of the probability mass functions $f_i(x_j, x_k)$ in class c_i are larger than those used to estimate $f_i(x_j, x_k, x_l)$, $f_i(x_j, x_k, x_l, x_o)$, etc. Thus, compared with a classifier that considers the relationship between each attribute triple, quadruple, etc., $\varphi_{pnb}(\mathbf{x})$ is less likely to have to deal with noninformative sample probabilities. Also, for discrete $\mathbf{X}$, $\varphi_{pnb}(\mathbf{x})$ has a smaller chance of running into estimability issues since it is more likely that (x_j, x_k) will arise in the training sample than (x_j, x_k, x_l), (x_j, x_k, x_l, x_o), etc. In the case of normal attributes, it is less likely that the covariance matrix for (X_j, X_k) will run into any singularity issues compared with the covariance matrices for (X_j, X_k, X_l), (X_j, X_k, X_l, X_o), etc. and, thus, less likely that the pairwise density estimation required for $\varphi_{pnb}(\mathbf{x})$ will encounter any computational issues.

In this paper, we successfully develop our PNB classifier as an alternative to the NB, LBR, TAN, AODE, and HNB classifiers when the Bayes classifier is not computationally feasible and show, through extensive simulation studies and applications to actual data sets, that the increase in accuracy of our PNB classifier using normal density estimation is statistically significant in data sets containing at least some continuous attributes. In addition, we show through these studies and applications that our PNB classifier using normal density estimation is statistically significantly more accurate than even the Bayes classifier for data sets with all normal attributes.

In Sec. 2, we provide a necessary and sufficient condition for which $\varphi_{pnb}(\mathbf{x})$ is optimal, along with a necessary and sufficient condition for which $\varphi_{pnb}(\mathbf{x})$ gives the same classification results as $\varphi_{nb}(\mathbf{x})$. We then explore sufficient conditions for the optimality of $\varphi_{pnb}(\mathbf{x})$ and $\varphi_{nb}(\mathbf{x})$ for normal $\mathbf{X}$ in Sec. 3. Next, we evaluate the classification performance of the Bayes, PNB, NB, LBR, TAN, AODE, and HNB classifiers using simulation studies in Sec. 4 and selected data sets from the UCI Machine Learning Repository[2,5] in Sec. 5. For these applications, SP-TAN is not considered due to its comparable accuracy to LBR. We then conclude with a final discussion in Sec. 6.

2. Equivalence of PNB Classifier to Bayes and NB Classifiers

We use the pseudo ANB representation in (6) to develop the conditions for which $\varphi_{pnb}(\mathbf{x})$ produces the same classification results as $\varphi_b(\mathbf{x})$ and $\varphi_{nb}(\mathbf{x})$ for $m = 2$. Before we do so, we need to define a measure of the relationship in class c_i between an attribute pair (X_j, X_k) and its parent set $\text{pa}(X_j, X_k)$ for a pseudo ANB network. We call this measure the *local dependence* of (X_j, X_k) in class c_i, which we define in Sec. 2.1 and use to state the equivalence conditions for $\varphi_{pnb}(\mathbf{x})$ in Sec. 2.2.

2.1. Measure for the local dependence of a pair of attributes

In a pseudo ANB network T, we define the following measure for the local dependence of (X_j, X_k).

Definition 1—For $A = (X_j, X_k)$ $(j, k = 1, \ldots, P, j < k)$ in T, the local dependence derivative of A in classes c_1 and c_2 are

$$dd^1_T(a|\mathrm{pa}(a)) = \frac{f_1(a|\mathrm{pa}(a))}{f_1(a)}, \quad dd^2_T(a|\mathrm{pa}(a)) = \frac{f_2(a|\mathrm{pa}(a))}{f_2(a)}. \quad (9)$$

Intuitively, the ratio $dd^i_T(a|\mathrm{pa}(a)) > 1$ determines how strongly the set of attributes pa(X_j, X_k) affect node $A = (X_j, X_k)$ in each class. If A is not related to any other nodes, then $dd^1_T(a|\mathrm{pa}(a)) = dd^2_T(a|\mathrm{pa}(a)) = 1$. If $dd^i_T(a|\mathrm{pa}(a)) > 1$, then the set pa$(A)$ = pa(X_j, X_k) increases the probability of A in class c_i and helps to support classification into class c_i. However, we want to determine the class for which the set pa(X_j, X_k) has greater influence and, thus, the ratio of the two derivatives in (9) is primarily of interest.

Definition 2—For node $A = (X_j, X_k)$ in T, the local dependence derivative ratio at A is

$$ddr_T(a) = \frac{dd^1_T(a|\mathrm{pa}(a))}{dd^2_T(a|\mathrm{pa}(a))}, \quad (10)$$

which measures the influence of the local dependence of A on the classification results. In addition,

1. If pa(A) is empty, $ddr_T(a) = 1$. This makes intuitive sense since A not being related to any other attribute means there is nothing to support classifying A into one class or another.
2. If $dd^1_T(a|\mathrm{pa}(a)) = dd^2_T(a|\mathrm{pa}(a))$, then $ddr_T(a) = 1$. This implies that the local dependence of A is evenly distributed in both classes. Thus, regardless of how strong an influence the set pa(A) has in each class, it has no effect on classification.
3. If $ddr_T(a) > 1$, then the local dependence of A in class c_1 is stronger than in class c_2 and vice versa if $ddr_T(a) < 1$.

2.2. Conditions for equivalence of PNB classifier to Bayes and NB classifiers

In this section, we explore the conditions under which our proposed PNB classifier $\varphi_{\mathrm{pnb}}(\mathbf{x})$ produces the same classification results as the Bayes and NB classifiers $\varphi_b(\mathbf{x})$ and $\varphi_{\mathrm{nb}}(\mathbf{x})$. In doing so, we utilize

$$D=\left[\prod_{\substack{l=1\\ l\text{ odd}}}^{P-2} ddr_T(a_l)\right]\left[\frac{f_1(x_P)}{f_2(x_P)}\prod_{j=1}^{P-1}\prod_{k=j+1}^{P}\frac{f_2(x_j,x_k)}{f_1(x_j,x_k)}\right]$$

$$(j,k)\notin\{(1,2),(3,4),\ldots,(P-2,P-1)\},$$

$$D'=\left[\prod_{\substack{l=1\\ l\text{ odd}}}^{P-3} ddr_T(a_l)\right]\left[\frac{f_1(x_{P-1},x_P)}{f_2(x_{P-1},x_P)}\prod_{j=1}^{P-1}\prod_{k=j+1}^{P}\frac{f_2(x_j,x_k)}{f_1(x_j,x_k)}\right]$$

$$(j,k)\notin\{(1,2),(3,4),\ldots,(P-3,P-2)\},$$

$$D''=\prod_{p=1}^{P} ddr_T(x_p), \quad (11)$$

for a given $\mathbf{x} = (x_1, \ldots, x_P)$, where $A_l = (X_l, X_{l+1})$.

The following theorem defines the relation between $\varphi_b(\mathbf{x})$ and $\varphi_{\text{pnb}}(\mathbf{x})$.

Theorem 1—*For a "pseudo" ANB T,*

$$\begin{cases}\phi_b(\mathbf{x})=\phi_{\text{pnb}}(\mathbf{x})\cdot D & \text{odd } P,\\ \phi_b(\mathbf{x})=\phi_{\text{pnb}}(\mathbf{x})\cdot D' & \text{even } P.\end{cases} \quad (12)$$

Proof: Based on (6), (9), and (10), we have that for odd *P*,

$$\phi_b(\mathbf{x})=\frac{\pi_1}{\pi_2}\frac{f_1(x_P)}{f_2(x_P)}\prod_{\substack{l=1\\ l\text{ odd}}}^{P-2}\frac{f_1(x_l,x_{l+1}|\text{pa}(x_l,x_{l+1}))}{f_2(x_l,x_{l+1}|\text{pa}(x_l,x_{l+1}))}=\left[\frac{\pi_1}{\pi_2}\prod_{j=1}^{P-1}\prod_{k=j+1}^{P}\frac{f_1(x_j,x_k)}{f_2(x_j,x_k)}\right]\left[\prod_{\substack{l=1\\ l\text{ odd}}}^{P-2}\frac{f_1(x_l,x_{l+1}|\text{pa}(x_l,x_{l+1}))}{f_2(x_l,x_{l+1}|\text{pa}(x_l,x_{l+1}))}\right]\times\left[\frac{f_1(x_P)}{f_2(x_P)}\prod_{j=1}^{P-1}\right.$$

For $(j, k) \notin \{(1, 2), (3, 4), \ldots, (P-2, P-1)\}$,

$$\begin{aligned}\phi_b(\mathbf{x}) &= \phi_{\text{pnb}}(\mathbf{x}) \left[\prod_{\substack{l=1 \\ \text{l odd}}}^{P-2} \frac{f_1(x_l, x_{l+1}|\text{pa}(x_l, x_{l+1}))f_2(x_l, x_{l+1})}{f_2(x_l, x_{l+1}|\text{pa}(x_l, x_{l+1}))f_1(x_l, x_{l+1})} \right] \\ &\quad \times \left[\frac{f_1(x_P)}{f_2(x_P)} \prod_{j=1}^{P-1} \prod_{k=j+1}^{P} \frac{f_2(x_j, x_k)}{f_1(x_j, x_k)} \right] \\ &= \phi_{\text{pnb}}(\mathbf{x}) \left[\prod_{\substack{l=1 \\ \text{l odd}}}^{P-2} \frac{dd_T^1(a_l|\text{pa}(a_l))}{dd_T^2(a_l|\text{pa}(a_l))} \right] \left[\frac{f_1(x_P)}{f_2(x_P)} \prod_{j=1}^{P-1} \prod_{k=j+1}^{P} \frac{f_2(x_j, x_k)}{f_1(x_j, x_k)} \right] \\ &= \phi_{\text{pnb}}(\mathbf{x}) \left[\prod_{\substack{l=1 \\ \text{l odd}}}^{P-2} ddr_T(a_l) \right] \left[\frac{f_1(x_P)}{f_2(x_P)} \prod_{j=1}^{P-1} \prod_{k=j+1}^{P} \frac{f_2(x_j, x_k)}{f_1(x_j, x_k)} \right].\end{aligned}$$

A similar argument holds for even P.

From Theorem 1, we see that two factors differentiate $\varphi_b(\mathbf{x})$ from $\varphi_{\text{pnb}}(\mathbf{x})$:

1. the product of the local dependence derivative ratios for $\{X_1, X_2\}$, $\{X_3, X_4\}$, …,
2. the product of the ratios of the group conditional probabilities for all other pairs $\{X_j, X_k\}$ (and X_P for odd P).

Thus, D and D' show how the local dependence of $\{X_1, X_2\}$, $\{X_3, X_4\}$, … distributes in each class, and how these local dependencies work together with the relationships existing among all other $\{X_j, X_k\}$. For instance, if $D = D' = 1$, then $\varphi_b(\mathbf{x})$ and $\varphi_{\text{pnb}}(\mathbf{x})$ are equivalent. Although it is clear from (12) that $D = 1$ or $D' = 1$ is sufficient for the equivalence of $\varphi_b(\mathbf{x})$ and $\varphi_{\text{pnb}}(\mathbf{x})$, it is not a requirement. A necessary and sufficient condition for the equivalence of $\varphi_b(\mathbf{x})$ and $\varphi_{\text{pnb}}(\mathbf{x})$ is provided in the following corollary, which follows from Theorem 1 and (4).

Corollary 1—*For a given* $\mathbf{x}$, *the classifiers* $\varphi_b(\mathbf{x})$ *and* $\varphi_{\text{pnb}}(\mathbf{x})$ *are equal under zero-one loss, i.e.* $\varphi_b(\mathbf{x}) \doteq \varphi_{\text{pnb}}(\mathbf{x})$ *if and only if*

- (*odd* P) *when* $\varphi_{\text{pnb}}(\mathbf{x}) \geq 1$, $\frac{1}{D} \leq \phi_{\text{pnb}}(\mathbf{x})$ *or when* $\varphi_{\text{pnb}}(\mathbf{x}) < 1$, $\frac{1}{D} > \phi_{\text{pnb}}(\mathbf{x})$,
- (*even* P) *when* $\varphi_{\text{pnb}}(\mathbf{x}) \geq 1$, $\frac{1}{D'} \leq \phi_{\text{pnb}}(\mathbf{x})$ *or when* $\varphi_{\text{pnb}}(\mathbf{x}) < 1$, $\frac{1}{D'} > \phi_{\text{pnb}}(\mathbf{x})$.

If this condition holds for every x in the attribute space, i.e. $\varphi_b \doteq \varphi_{\text{pnb}}$, then the PNB classifier is globally optimal.

Suppose there are instances where we can compute $\varphi_{\text{pnb}}(\mathbf{x})$, D, and D', but not $\varphi_b(\mathbf{x})$, e.g. high-dimensional normal data sets. Corollary 1 describes when $\varphi_{\text{pnb}}(\mathbf{x})$ would give the same classification as $\varphi_b(\mathbf{x})$ if $\varphi_b(\mathbf{x})$ were possible to compute.

From Theorem 1, Corollary 1, and the results in Ref. 36, we also have the following corollary.

Corollary 2—*For D and D'' defined as in* (11),

1. $$\begin{cases} \phi_{\text{pnb}}(\mathbf{x}) = \phi_{\text{nb}}(\mathbf{x}) \cdot D^{-1} \cdot D'' & \text{odd } P, \\ \phi_{\text{pnb}}(\mathbf{x}) = \phi_{\text{nb}}(\mathbf{x}) \cdot D'^{-1} \cdot D'' & \text{even } P. \end{cases} \quad (13)$$

2. *For a given* $\mathbf{x}$, $\varphi_{\text{pnb}}(\mathbf{x}) \doteq \varphi_{\text{nb}}(\mathbf{x})$ *if and only if*
   - (*odd P*) *when* $\varphi_{\text{pnb}}(\mathbf{x}) \geq 1$, $D^{-1}D'' \leq \varphi_{\text{pnb}}(\mathbf{x})$ *or when* $\varphi_{\text{pnb}}(\mathbf{x}) < 1$, $D^{-1}D'' > \varphi_{\text{pnb}}(\mathbf{x})$,
   - (*even P*) *when* $\varphi_{\text{pnb}}(\mathbf{x}) \geq 1$, $D'^{-1}D'' \leq \varphi_{\text{pnb}}(\mathbf{x})$ *or when* $\varphi_{\text{pnb}}(\mathbf{x}) < 1$, $D'^{-1}D'' > \varphi_{\text{pnb}}(\mathbf{x})$.

3. Sufficient Conditions for Optimality of PNB Classifier: Normal Case

We now demonstrate sufficient optimality conditions for $\varphi_{\text{pnb}}(\mathbf{x})$ for $m = 2$ classes, where we focus on the normal case due to its ubiquity in practice. In our discussion, we work with $\log(\phi_b(\mathbf{x})) \equiv \phi_b^*(\mathbf{x})$, $\log(\phi_{\text{pnb}}(\mathbf{x})) \equiv \phi_{\text{pnb}}^*(\mathbf{x})$, and $\log(\phi_{\text{nb}}(\mathbf{x})) \equiv \phi_{\text{nb}}^*(\mathbf{x})$ and assume equal priors.

We assume $\mathbf{X}$ is normally distributed in class c_i, with known mean vector $\boldsymbol{\mu}_i = (\mu_{1,i}, \ldots, \mu_{P,i})$ and covariance matrix $\boldsymbol{\Sigma}$ such that X_p has known variance σ^2 and (X_j, X_k) have known correlation ρ. To ensure that $\boldsymbol{\Sigma}$ is positive definite, we assume $-1/(P-1) < \rho < 1$. We then define

$$\omega_1 = \frac{1}{\sigma^2(1-\rho^2)}, \quad \omega_2 = \frac{1}{\sigma^2(1-\rho)[1+(P-1)\rho]}, \quad a_j = \mu_{j,1} - \mu_{j,2},$$

$$x_{j,c} = x_j - \frac{1}{2}(\mu_{j,1} + \mu_{j,2}), \quad l_j = [1+(P-2)\rho]x_{j,c} - \rho \sum_{\substack{k=1 \\ k \neq j}}^{P} x_{k,c}, \quad (14)$$

$$m_j = (P-2)x_{j,c} - \rho \sum_{\substack{k=1 \\ k \neq j}}^{P} x_{k,c},$$

noting that both ω_1 and ω_2 are positive. Using normal density formulas and matrix properties, we have that $\phi^*_{\text{nb}}(\mathbf{x})$ is *not* optimal if

$$\phi^*_b(\mathbf{x})\phi^*_{\text{nb}}(\mathbf{x}) = \frac{\omega_2}{\sigma^2}\left[\sum_{j=1}^{P} a_j^2 l_j x_{j,c} + \sum_{j=1}^{P-1}\sum_{k=j+1}^{P} a_j a_k (x_{j,c} l_k + x_{k,c} l_j)\right], \quad (15)$$

is negative. On the other hand, $\phi^*_{\text{pnb}}(\mathbf{x})$ is optimal if

$$\phi^*_b(\mathbf{x})\phi^*_{\text{pnb}}(\mathbf{x}) = \omega_1\left[\sigma^2\phi^*_b(\mathbf{x})\phi^*_{\text{nb}}(\mathbf{x}) + \omega_2\left(\sum_{j=1}^{P} a_j^2 l_j m_j + \sum_{j=1}^{P-1}\sum_{k=j+1}^{P} a_j a_k (l_j m_k + l_k m_j)\right)\right] \quad (16)$$

is non-negative. For instance, $\phi^*_{\text{pnb}}(\mathbf{x})$ is optimal, while $\phi^*_{\text{nb}}(\mathbf{x})$ is not, when $\mathbf{x}$ = (2.047, 1.896, 0.972), $\boldsymbol{\mu}_1$ = (1, 2, 3) and $\boldsymbol{\mu}_2$ = (3, 2, 3), σ^2 = 1, and ρ = −0.45. Therefore, (15) and (16) provide explicit conditions for when the PNB classifier is optimal and NB is not.

Suppose we consider the case of equal mean differences ($a_j \equiv a$), where Zhang[36] states that $\phi^*_{\text{nb}}(\mathbf{x})$ is optimal for $P = 2$ only under certain conditions. We have that $\phi^*_{\text{pnb}}(\mathbf{x})$ and $\phi^*_{\text{nb}}(\mathbf{x})$ are always optimal if $a_j \equiv a$, where (15) and (16) simplify to

$$\phi^*_b(\mathbf{x})\phi^*_{\text{nb}}(\mathbf{x}) = \omega_2(1-\rho)\left(\frac{a}{\sigma}\sum_{j=1}^{P} x_{j,c}\right)^2,$$

$$\phi^*_b(\mathbf{x})\phi^*_{\text{pnb}}(\mathbf{x}) = (P-1)\omega_1\omega_2\left[a(1-\rho)\sum_{j=1}^{P} x_{j,c}\right]^2,$$

both of which are always non-negative.

4. Simulation Studies

We now investigate the classification performance of our proposed PNB classifier relative to the Bayes, NB, LBR, TAN, AODE, and HNB classifiers for discrete and continuous data, where we focus on classification into m = 2 classes. All analyses are run using R software version 3.1.2,[24] with the **RWeka** package used to construct the LBR, TAN, AODE, and HNB classifiers.

4.1. Binary data

We first consider the case of correlated binary attributes. Using the algorithm of Kang and Jung,[14] we simulate S = 50 data sets such that in each, N observations $\mathbf{y}$ = (y_1, y_2, y_3) have prior probability 0.5 of belonging to class c_i (i = 1, 2), where

$$f_i(y_1, y_2, y_3) = \frac{\exp(\sum_{j=1}^{3} \delta_{ij} y_j + \sum_{j<k} \gamma_{i,jk} y_j y_k + \alpha_i y_1 y_2 y_3)}{\sum_{\text{all values of } (y_1,y_2,y_3)} \exp(\sum_{j=1}^{3} \delta_{ij} y_j + \sum_{i<j} \gamma_{i,jk} y_j y_k + \alpha_i y_1 y_2 y_3)}. \quad (17)$$

In our simulations, we consider the following cases:

1. Case 1: $\boldsymbol{\delta}_1 = (\delta_{11}, \delta_{12}, \delta_{13}) = (0.1, 0.2, 0.3)$, $\boldsymbol{\delta}_2 = (\delta_{21}, \delta_{22}, \delta_{23}) = (0.1, 0.2, 0.3)$; $\boldsymbol{\gamma}_1 = (\gamma_{1,12}, \gamma_{1,13}, \gamma_{1,23}) = (0, 0, 0)$, $\boldsymbol{\gamma}_2 = (\gamma_{2,12}, \gamma_{2,13}, \gamma_{2,23}) = (0.1, 0.2, 0.3)$; $\alpha_1 = 2$, $\alpha_2 = 1$.
2. Case 2: $\boldsymbol{\delta}_1 = (\delta_{11}, \delta_{12}, \delta_{13}) = (0.1, 0.2, 0.3)$, $\boldsymbol{\delta}_2 = (\delta_{21}, \delta_{22}, \delta_{23}) = (0.1, 0.2, 0.3)$; $\boldsymbol{\gamma}_1 = (\gamma_{1,12}, \gamma_{1,13}, \gamma_{1,23}) = (0.55, 0.66, 0.77)$, $\boldsymbol{\gamma}_2 = (\gamma_{2,12}, \gamma_{2,13}, \gamma_{2,23}) = (0.06, 0.05, 0.08)$; $\alpha_1 = -5$, $\alpha_2 = -1$.

We consider $N = 100, 200$ observations for each case when applying each classifier. Using 10-fold cross-validation (CV), we compute the error rates for each classifier, all of which are estimated empirically. Also, to test whether one classifier has a significantly higher or lower CV error rate than another for a particular data set, we use the Nadeau and Bengio's[21] corrected resampled t-test, which they show yields statistical tests with correct test sizes and high power. In Table 2, we present, for each (N, case) and classifier, the mean CV error rate and CPU time in seconds (training time + classification time) averaged across simulations, along with the numbers of w wins (data sets where the classifier has the lowest error), d draws (data sets where the classifier does not have significantly higher error than the winner at the 10% level), and l losses (data sets where the classifier has significantly higher error than the winner at the 10% level). Although our proposed PNB classifier does not appear to dominate if one examines its mean CV error rate and CPU time, we do see that it wins most often in three out of the four (N,case) scenarios among all classifiers other than the Bayes classifier. In addition, the PNB classifier draws with the winner most often for three out of the four scenarios and is never significantly worse than the winner.

4.2. Continuous data: Normal case

To assess the classification performance of our proposed PNB classifier relative to the other classifiers discussed based on data that one could encounter in practical applications, we conduct a simulation study using biomarker data from a cardiovascular study conducted by the High Risk Plaque Initiative [BG Medicine Inc. (Waltham, MA) and other partners].[3] In modeling this simulation study, we consider the 591 continuous biomarkers measured on $N = 136$ subjects belonging to either one of two diagnostic groups, namely, individuals who underwent a near-term cardiovascular event and those who did not. For each of $S = 25$ data sets, we simulate $N = 136$ observations $\mathbf{x} = (x_1, \ldots, x_P)$ such that each has prior probability $\pi_i \equiv 0.5$ ($i = 1, 2$) of belonging to class c_i. We assume $\mathbf{X}$ is normally distributed with mean $\boldsymbol{\mu}_i$ and covariance $\boldsymbol{\Sigma}_i$ in class c_i, where $\boldsymbol{\mu}_i$ and $\boldsymbol{\Sigma}_i$ are set to equal the sample mean vector and covariance of the P biomarkers in the ith diagnostic group. To examine the results compare for differing number of attributes, we simulate the observations $(x_1, \ldots, x_P)$ based on the sample mean vectors and covariance matrix of $P = 25, 50, 100, 300$ randomly selected biomarkers, and also on the sample mean vectors and covariance matrix of all $P = 591$ biomarkers.

In estimating the LBR, TAN, AODE, and HNB classifiers, we discretize all attributes using Fayyad and Irani's discretization method, since all four classifiers used this entropy minimization-based approach in their applications. To estimate the Bayes, PNB, and NB classifiers, we first use normal density estimation assuming either $\Sigma_1 \neq \Sigma_2$ or $\Sigma_1 = \Sigma_2$, under which we use either the sample pooled covariance $\hat{\Sigma}_{\text{pool}}$ or the sample class covariances $\hat{\Sigma}_i$. Next, we empirically estimate the PNB and NB classifiers. Since Fayyad and Irani's entropy minimization-based discretization approach has been noted to fail in certain data sets,[25] we instead discretize each X_p by rounding its values to two decimal places and categorizing the rounded values, which we then use to empirically estimate the PNB and NB classifiers. We round to two decimal places to reduce computation time and the chance of obtaining non-informative sample probabilities. In computing the error rates for each classifier, we use 2-fold CV for each data set.

In Table 3, we present, for each P and each classifier, the mean CV error rate and mean CPU time in seconds averaged across simulations, along with the numbers of w wins, d draws, and l losses using the statistical testing approach described in Sec. 4.1. We see that what our proposed PNB classifier may lack in computational speed, it compensates for in terms of accuracy when we use normal density estimation. Specifically, for each P, the lowest error rates and highest number of wins are obtained when we correctly assume $\Sigma_1 \neq \Sigma_2$ and use the estimates $\hat{\Sigma}_i$. In addition, with the exception of $P = 25$, we see that in computing the normal-based PNB classifier using $\hat{\Sigma}_i$, it consistently wins across all data sets for each P. Considering that this result holds for $P = 300, 591$, where $N = 136 < P$, the potential benefit of this normal-based PNB classifier with respect to accuracy in high-dimensional data sets is worth noting.

5. Applications

We then examine the performance of the Bayes, PNB, NB, LBR, TAN, AODE, and HNB classifiers as applied to 22 data sets from the UCI Machine Learning Repository,[2,5] the description of which can be found in Table 4. For each data set, we only consider attributes with complete cases. In addition, we examine the BG Medicine biomarker data described in Sec. 4.2, along with genetic microarray data obtained from a study conducted by Wang *et al.*,[29] which are labeled "BG Medicine" and "MDACC" in Table 4.

Since we are mainly interested in how results are affected by data type, we consider data sets with either all quantitative or categorical attributes and those with a mix of quantitative and categorical attributes. For the quantitative data sets, we consider the empirical PNB, NB, HNB, AODE, LBR, and TAN classifiers, where attributes are discretized using the approach described in Sec. 4.2. We also apply the normal-based Bayes, PNB, and NB classifiers, which are computed as in Sec. 4.2. In cases where $\mathbf{X}$ is categorical or mixed, we consider the empirical PNB, NB, HNB, AODE, LBR, and TAN classifiers. However, for mixed $\mathbf{X}$, we also estimate $f_k(x_j, x_k)$ for the PNB classifier and $f_k(x_j)$ for the NB classifier using normal density estimation for quantitative attribute pairs (X_j, X_k) and single attributes X_j, and Laplace estimation for categorical (X_j, X_k) and X_j. For attribute pairs (X_j, X_k) where X_j is continuous and X_k categorical, X_j is discretized and $f_k(x_j, x_k)$ is estimated using Laplace estimation.

In Tables 5 and 6, we display the CV the error rates for each data set and classifier using normal density and empirical probability estimation, respectively. For each data type, we also display the numbers of w wins, d draws, and l losses, along with mean CPU time averaged across data sets. With the exception of the "balance" data set, the HNB classifier has the lowest error across the data sets with categorical $\mathbf{X}$. In addition, it also has the fastest mean computation time and most wins. For the mixed data sets, the HNB and normal-based PNB classifiers generally dominate in terms of error rates and number of wins. For the continuous data sets, the normal-based Bayes has most wins and the lowest error, followed by our normal-based PNB classifier.

In short, although its computation time is longer compared with the Bayes, NB, HNB, AODE, LBR, and TAN classifiers, the relatively high accuracy level (in terms of error rates and number of wins) for our proposed normal-based PNB classifier in the mixed and continuous data sets examined cannot be ignored. For instance, if computation of the normal-based Bayes classifier for the continuous data sets was not feasible, the normal-based PNB classifier would have had the most wins among the NB, HNB, AODE, LBR, and TAN classifiers. Due to the varying performance of the normal-based PNB classifier for these data sets based on whether the estimates $\hat{\boldsymbol{\Sigma}}_{\text{pool}}$ or $\hat{\boldsymbol{\Sigma}}_i$ are used, we recommend applying this classifier using both types of covariance estimates. We note that in the continuous and mixed data sets, the empirical PNB classifier generally does not outperform its normal-based counterpart in terms of accuracy. However, in recognition of the fact that the normal-based PNB classifier retains more information than the empirical PNB classifier by not discretizing the continuous attributes, the superior performance of the normal-based PNB classifier is not surprising.

6. Conclusion

We propose the PNB classifier as an alternate approach to not only NB, but also to the LBR, TAN, AODE, and HNB classifiers aimed at relaxing the conditional independence assumption integral to NB. However, our classifier also goes beyond the HNB, AODE, LBR, and TAN classifiers in that rather than discretizing continuous attributes and potentially losing information, which can lead to increased error, it instead allows for the use of density estimation. Through comprehensive simulation studies and applications to actual data sets, we illustrate that the use of normal density estimation for the PNB classifier leads to an increase in computational accuracy over the NB, LBR, TAN, AODE, and HNB classifiers that is statistically significant in data sets with continuous attributes. In data sets containing all normal attributes, we also show that the PNB classifier has an increase in accuracy over the Bayes classifier that is statistically significant. We also formulate exact conditions where the Bayes and PNB classifiers have the same classification performance, even when there are strong dependencies that extend beyond a pair of attributes. In particular, we show that these conditions are based solely on how the relationships between each pair of attributes and all other attributes work together to support or cancel one another to determine classification.

Despite the amount of computation time involved, our experimental results show that the high level of computational accuracy displayed by our proposed normal-based PNB

classifier in the examined data sets with all or mostly continuous attributes is highly beneficial. Specifically, for these types of data sets, our experimental results demonstrate several instances in which the normal-based PNB classifier is statistically significantly more accurate than not only NB, but also the more established HNB, AODE, LBR, and TAN classifiers that have been shown to yield higher accuracy relative to NB. In our simulation study, we have also shown that the increase in accuracy of our normal-based PNB classifier over the normal-based Bayes classifier is statistically significant in certain instances, including those dealing with high-dimensional data where computational issues for the normal-based Bayes classifier are likely to arise. Considering the simplicity of the normal-based PNB classifier and the breadth of the attribute relationships it accounts for, along with the ubiquity of data sets with continuous attributes and the assumption of normality when analyzing such attributes, the normal-based PNB classifier is a promising algorithm that can potentially yield important results in many practical applications.

Acknowledgments

The authors thank BG Medicine, Inc. for providing the cardiovascular biomarker data, Drs. Xingbin Wang (Department Human Genetics, University of Pittsburgh Graduate School of Public Health, Pittsburgh, PA) and Etienne Sibille (Department of Psychiatry, University of Pittsburgh, Pittsburgh, PA) for providing the genetic microarray data, and Dr. William H. Wolberg (University of Wisconsin Hospital, Madison, WI) for providing the "bcw" data set. The authors also thank the Editor, Associate Editor, and reviewers of *International Journal of Pattern Recognition and Artificial Intelligence* and *Machine Learning* for their helpful suggestions, which improved our presentation. This work was supported by the National Institutes of Health [T32NS048005 to J.K.A. and R.A.B.; F32NS081904 to J.K.A.]. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health.

References

1. Anderson, TW. An Introduction to Multivariate Statistical Analysis. 2nd. New York, NY: John Wiley & Sons; 1984.
2. Bache, K.; Lichman, M. UCI Machine Learning Repository. Irvine, CA: University of California, School of Information and Computer Science; 2013. Available at http://archive.ics.uci.edu/ml
3. Balasubramanian R, Houseman EA, Coull BA, Lev MH, Schwamm LH, Betensky RA. Variable importance in matched case-control studies in settings of high dimensional data. J. R. Stat. Soc. C Appl. Stat. 2014
4. Bennett, PN. Technical report. Carnegie Mellon University; 2000. Assessing the calibration of naïve Bayes' posterior estimates.
5. Bennett KP, Mangasarian OL. Robust linear programming discrimination of two linearly inseparable sets. Optim. Methods Softw. 1992; 1:23–34.
6. Breiman, L.; Friedman, JH.; Olshen, RA.; Stone, CJ. Classification and Regression Trees. Belmont, CA: Wadsworth Int. Group; 1984.
7. Chow CK, Liu CN. Approximating discrete probability distributions with dependence trees. IEEE Trans. Info. Theory. 1968; 14:462–467.
8. Domingos P, Pazzani M. On the optimality of the simple Bayesian classifier under zero-one loss. Mach. Learn. 1997; 29:103–130.
9. Friedman N, Geiger D, Goldszmidt M. Bayesian network classifiers. Mach. Learn. 1997; 29:131–163.
10. Fayyad, UM.; Irani, KB. Proc. IJCAI-13. Morgan Kaufmann; 1993. Multi-interval discretization of continuous-valued attributes for classification learning; p. 1022-1027.
11. Grossman, D.; Domingos, P. Proc. 21st Int. Conf. Machine Learning. ACM Press; 2004. Learning Bayesian network classifiers by maximizing conditional likelihood; p. 361-368.

12. Hall MA. A decision tree-based attribute weighting filter for naive Bayes. Knowl.-Based Syst. 2007; 20:120–126.
13. Hastie, T.; Tibshirani, R.; Friedman, J. The Elements of Statistical Learning: Data Mining, Inference, and Prediction. 2nd. New York, NY: Springer Verlag; 2009.
14. Kang SH, Jung SH. Generating correlated binary variables with complete specification of the joint distribution. Biometr. J. 2001; 43(3):263–269.
15. Keogh E, Pazzani MJ. Learning augmented Bayesian classifiers: A comparison of distribution-based and classification-based approaches. Proc. Int. Workshop Art. Intell. and Stats. 1999:225–230.
16. Keogh E, Pazzani MJ. Learning the structure of augmented Bayesian classifiers. Int. J. Artif. Intell. Tools. 2002; 11(4):587–601.
17. Kononenko, I. Proc. Sixth Euro. Working Session on Learning. Porto, Portugal: 1996. Semi-naïve Bayesian classifier; p. 206-219.
18. Kuncheva LI. On the optimality of naïve Bayes with dependent binary features. Pattern Recogn. Lett. 2006; 27:830–837.
19. Langley, P.; Sage, S. Proc. 10th Conf. Uncertainty Art. Intell. Morgan Kaufmann; 1994. Induction of selective Bayesian classifiers; p. 399-406.
20. Monti, S.; Cooper, FG. Proc. 15th Conf. Uncertainty Art. Intell. Morgan Kaufmann; 1999. A Bayesian network classifier that combines a finite mixture model and a naïve Bayes model; p. 447-456.
21. Nadeau C, Bengio Y. Inference for the generalization error. Mach. Learn. 2003; 52:239–281.
22. Pazzani, MJ. Learning from Data: Artificial Intelligence and Statistics V. Springer-Verlag; 1996. Searching for dependencies in Bayesian classifiers; p. 239-248.
23. Pearl, J. Probabilistic Reasoning in Intelligent Systems. San Francisco, CA: Morgan Kaufmann; 1988.
24. R Core Team. R Foundation for Statistical Computing. Vienna, Austria: 2013. R: A language and environment for statistical computing. http://www.R-project.org/
25. Ranuva, ST.; Zaidi, N. AnDE: An Extended Bayesian Learning Technique developed by Dr. Geoff Webb, R package version 1.0, R Foundation for Statistical Computing. Vienna, Austria: 2013. http://CRAN.R-project.org/package=AnDE
26. Sahami, M. Learning limited dependence Bayesian classifiers; Proc. Second Int. Conf. Knowledge Disc. Data Mining; 1996. p. 335-338.
27. Silvescu, A.; Andorf, C.; Dobbs, D.; Honavar, V. Technical report. Iowa State University; 2004. Inter-element dependency models for sequence classification.
28. Singh, M.; Provan, GM. Efficient learning of selective Bayesian network classifiers; Proc. 13th Int. Conf. Machine Learning; 1996. p. 453-461.
29. Wang X, Lin Y, Song C, Sibille E, Tseng GC. Detecting disease-associated genes with confounding variable adjustment and the impact on genomic meta-analysis: With application to major depressive disorder. BMC Bioinformatics. 2012; 13(52):1–15. [PubMed: 22214541]
30. Wang, Z.; Webb, GI. Comparison of lazy Bayesian rule and tree-augmented Bayesian learning; Proc. IEEE Int. Conf. Data Mining; 2002. p. 775-778.
31. Webb GI, Boughton JR, Wang Z. Not so naïve Bayes: Aggregating one-dependence estimators. Mach. Learn. 2005; 58:5–24.
32. Webb, GI.; Pazzani, MJ. Adjusted probability naïve Bayesian induction; Proc. 11th Australian Joint Conf. Artif. Intell; 1998. p. 285-295.
33. Xie, Z.; Hsu, W.; Liu, Z.; Lee, ML. A selective neighborhood based naïve Bayes for lazy learning. In: Chen, MS.; Yu, PS.; Liu, B., editors. Advances in Knowledge Discovery and Data Mining, Proceeding PAKDD. Berlin: Springer; 2002. p. 104-114.
34. Zaidi NA, Cerquides J, Carman MJ, Webb GI. Alleviating naive Bayes attribute independence assumption by attribute weighting. J. Mach. Learn. Res. 2013; 14:1947–1988.
35. Zhang NL. Hierarchical latent class models for cluster analysis. J. Mach. Learn. Res. 2004; 5:697–723.

36. Zhang H. Exploring conditions for the optimality of naïve Bayes. Int. J. Pattern Recogn. Artif. Intell. 2005; 19(2):183–198.
37. Zhang, H.; Jiang, L.; Su, J. Proc. Twentieth National Conf. Artif. Intell. AAAI Press; 2005. Hidden naïve Bayes; p. 919-924.
38. Zhang NL, Nielsen TD, Jensen FV. Latent variable discovery in classification models. Artif. Intell. Med. 2003; 30(3):283–299. [PubMed: 15081076]
39. Zheng Z, Webb GI. Lazy learning of Bayesian rules. Mach. Learn. 2000; 41:53–84.
40. Zheng, Z.; Webb, GI.; Ting, KM. Proc. 16th Int. Conf. Machine Learning. Morgan Kaufmann; 1999. Lazy Bayesian rules: A lazy semi-naïve Bayesian learning technique competitive to boosting decision trees; p. 493-502.

Biographies

Josephine K. Asafu-Adjei received her B.S. degree in Mathematics and Economics, an M.A. degree in Applied Statistics, and a Ph.D. in Statistics from the University of Pittsburgh, USA, in 2000, 2007, and 2011, respectively. Currently, she is serving as a Research Assistant Professor in the Department of Biostatistics and the School of Nursing at the University of North Carolina at Chapel Hill.

Rebecca A. Betensky received her A.B. degree in Mathematics from Harvard University, USA, in 1988, and her Ph.D. in Statistics from Stanford University, USA, in 1992. Currently, she is serving as a Professor of Biostatistics at the Harvard T.H. Chan School of Public Health, USA.

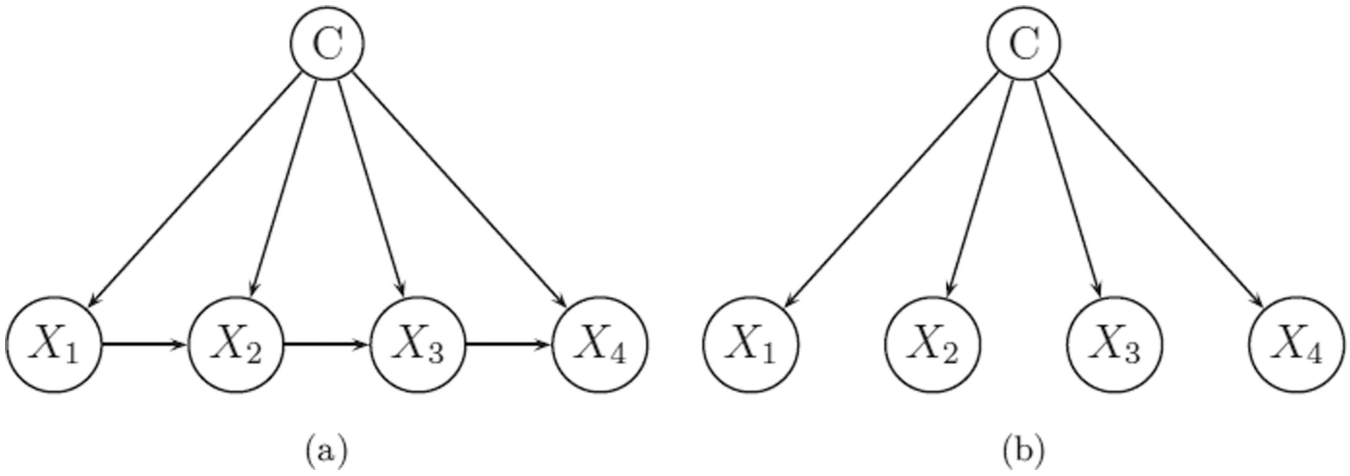

Fig. 1.
Examples of Bayesian networks. (a) ANB and (b) NB.

Table 1

Computational time complexity.

Classifier	Total Time Complexity
NB	$\mathcal{O}(tP) + \mathcal{O}(mP)$
LBR	$\mathcal{O}(tP) + \mathcal{O}(mtP^3)$
TAN	$\mathcal{O}(P^2(t + mv^2 + \log(P))) + \mathcal{O}(mP)$
SP-TAN	$\mathcal{O}(mtP^3) + \mathcal{O}(mP)$
AODE	$\mathcal{O}(tP^2) + \mathcal{O}(mP^2)$
HNB	$\mathcal{O}(P^2(t + mv^2)) + \mathcal{O}(mP^2)$
PNB	$\mathcal{O}(tP^2) + \mathcal{O}(mP^2)$

Note: m is the number of classes, P is the number of attributes, v is the average number of values for each attribute, and t is the number of training examples

Table 2

Summary of error rates, accuracy comparison results, and computation times.

N	Case		PNB	Bayes	HNB	AODE	LBR	TAN	NB
100	1	**ER**[a]	0.47	0.48	0.48	0.48	0.49	0.49	0.48
		w/d/l[b]	17/33/0	9/39/2	5/43/2	8/40/2	4/33/13	9/28/13	11/26/13
		Time[c]	0.08	0.03	0.03	0.04	0.04	0.06	0.06
100	2	**ER**[a]	0.49	0.45	0.48	0.48	0.49	0.49	0.49
		w/d/l[b]	8/42/0	23/25/2	6/35/9	7/33/10	1/30/19	8/26/16	4/31/15
		Time[c]	0.08	0.03	0.03	0.04	0.04	0.06	0.06
200	1	**ER**[a]	0.45	0.45	0.45	0.46	0.47	0.48	0.48
		w/d/l[b]	11/39/0	11/37/2	15/27/8	7/30/13	2/31/17	8/28/14	6/30/14
		Time[c]	0.16	0.06	0.03	0.04	0.05	0.06	0.12
200	2	**ER**[a]	0.48	0.45	0.48	0.48	0.48	0.50	0.50
		w/d/l[b]	11/39/0	23/24/3	6/29/15	6/27/17	2/28/20	3/22/25	2/22/26
		Time[c]	0.16	0.06	0.03	0.04	0.05	0.06	0.12

[a]Mean CV error rate.

[b]Number of wins (**w**), draws (**d**), and losses (**l**) across data sets for each (N, case).

[c]Mean CPU time in seconds (training + classification time).

Intern J Pattern Recognit Artif Intell. Author manuscript; available in PMC 2016 April 15.

Table 3

Summary of error rates, accuracy comparison results, and computation times.

		Normal ($\hat{\Sigma}_{pool}$)[a]			Normal ($\hat{\Sigma}_i$)[b]			Empirical					
P		**Bayes**	**PNB**	**NB**	**Bayes**	**PNB**	**NB**	**PNB**[c]	**NB**[c]	**HNB**	**AODE**	**LBR**	**TAN**
25	**ER**[d]	0.37	0.36	0.36	0.18	0.14	0.17	0.47	0.54	0.46	0.75	0.76	0.76
	w/d/l[e]	0/14/11	0/25/0	1/5/19	8/14/3	17/8/0	2/13/10	0/3/22	0/1/24	0/7/18	0/9/16	0/9/16	0/9/16
	Time[f]	<0.005	0.19	<0.005	<0.005	0.19	<0.005	0.76	0.05	0.08	0.09	0.11	0.10
50	**ER**[d]	0.35	0.34	0.34	0.50	0.10	0.12	0.49	0.53	0.45	0.83	0.84	0.84
	w/d/l[e]	0/7/18	0/25/0	0/7/18	0/3/22	25/0/0	3/12/10	0/0/25	0/0/25	0/8/17	0/5/20	0/6/19	0/6/19
	Time[f]	<0.005	0.75	0.01	<0.005	0.74	<0.005	1.50	0.12	0.15	0.15	0.24	0.17
100	**ER**[d]	0.51	0.33	0.34	0.50	0.03	0.04	0.49	0.54	0.29	0.90	0.91	0.91
	w/d/l[e]	0/1/24	0/25/0	0/2/23	0/2/23	25/0/0	5/14/6	0/1/24	0/0/25	0/15/10	0/3/22	0/3/22	0/2/23
	Time[f]	0.01	3.76	0.01	0.01	3.71	0.01	5.26	0.28	0.34	0.34	0.40	0.35
300	**ER**[d]	0.50	0.35	0.36	0.50	0.02	0.03	0.49	0.55	0.29	0.99	0.99	0.99
	w/d/l[e]	0/4/21	0/25/0	0/4/21	1/4/20	25/0/0	10/14/1	0/0/25	0/0/25	0/13/12	0/1/24	0/1/24	0/1/24
	Time[f]	0.01	28.20	0.02	0.01	28.17	0.02	35.17	0.65	1.04	0.94	0.83	0.83
591	**ER**[d]	0.50	0.33	0.33	0.49	0.02	0.02	0.50	0.54	0.25	1	1	1
	w/d/l[e]	0/4/21	0/25/0	1/4/20	0/4/21	25/0/0	9/13/3	0/0/25	0/1/24	0/20/5	0/0/25	0/0/25	0/0/25
	Time[f]	0.05	105.95	0.04	0.04	106.63	0.04	133.23	1.30	3.34	2.19	1.65	1.63

[a]Normal density estimation using pooled covariance $\hat{\Sigma}_{pool}$.

[b]Normal density estimation using sample covariances $\hat{\Sigma}_i$.

[c]Computed empirically.

[d]Mean CV error rate.

[e]Number of wins (**w**), draws (**d**), and losses (**l**) across data sets for each *P* and each classifier.

[f]Mean CPU time in seconds (training + classification time).

Table 4

Data sets.

Name	*N*	*p*[a]	*m*	Name	*N*	*p*[a]	*m*
adult	48,842	14	2	lung-cancer	32	55	3
balance-scale	625	4	3	MDACC	50	2000	2
bcw	699	9	2	mfeat-mor	2000	6	10
BG Medicine	136	591	2	new-thyroid	215	5	3
glass	214	9	7	pendigits	10,992	16	10
heart	270	13	2	post-operative	88	8	2
hepatitis	155	19	2	satellite	6435	36	6
house-votes	435	16	2	segment	2310	18	7
hypothyroid	3163	25	2	sonar	208	60	2
ionosphere	351	34	2	ttt	958	9	2
iris	150	4	3	vehicle	846	18	4
letter	20,000	16	26	wine	178	13	3

[a]Number of attributes with complete cases.

Table 5

Summary of error rates, accuracy comparison results, and computation times for classifiers using normal density estimation.

			Normal ($\hat{\Sigma}_{pool}$)[a]			Normal ($\hat{\Sigma}_i$)[b]		
Type[c]	**Data**	***V*-Fold CV**	**Bayes**	**PNB**	**NB**	**Bayes**	**PNB**	**NB**
Q	bcw	2	0.04	0.04	0.04	0.05	0.04	0.04
	BG Med	2	0.55	0.37	0.36	0.43	0.32	0.34
	ionosphere	3	0.12	0.16	0.17	0.64	0.64	0.25
	iris	3	0.02	0.04	0.04	0.03	0.03	0.05
	letter	2	0.30	0.38	0.40	0.12	0.31	0.36
	MDACC	2	0.54	0.54	0.54	0.34	0.58	0.54
	new-thyroid	5	0.09	0.07	0.09	0.04	0.04	0.04
	pendigits	2	0.12	0.18	0.19	0.12	0.21	0.35
	satellite	5	0.16	0.19	0.21	0.15	0.17	0.20
	segment	3	0.09	0.11	0.13	0.25	0.30	0.42
	sonar	4	0.24	0.28	0.30	0.32	0.33	0.33
	vehicle	3	0.22	0.43	0.57	0.15	0.41	0.54
	wine	2	0.03	0.06	0.06	0.03	0.04	0.04
	w/d/l[d]		4/9/0	0/13/0	0/7/6	6/4/3	2/11/0	1/5/7
	Time[e]		0.08	0.16	158.63	0.14	160.74	0.08
C	balance	5	NA[f]	NA[f]	NA[f]	NA[f]	NA[f]	NA[f]
	house-votes	5	NA[f]	NA[f]	NA[f]	NA[f]	NA[f]	NA[f]
	lung	3	NA[f]	NA[f]	NA[f]	NA[f]	NA[f]	NA[f]
	ttt	2	NA[f]	NA[f]	NA[f]	NA[f]	NA[f]	NA[f]
	w/d/l[d]		NA[f]	NA[f]	NA[f]	NA[f]	NA[f]	NA[f]
	Time[e]		NA[f]	NA[f]	NA[f]	NA[f]	NA[f]	NA[f]
M	adult	2	NA[f]	0.23	0.21	NA[f]	0.22	0.21
	glass	2	NA[f]	0.53	0.54	NA[f]	0.61	0.57
	heart	3	NA[f]	0.24	0.24	NA[f]	0.24	0.23

Type[c]	Data	*V*-Fold CV	Normal ($\hat{\Sigma}_{pool}$)[a]			Normal ($\hat{\Sigma}_j$)[b]		
			Bayes	PNB	NB	Bayes	PNB	NB
	hepatitis	3	NA[f]	0.19	0.16	NA[f]	0.17	0.17
	hypothyroid	2	NA[f]	0.03	0.03	NA[f]	0.03	0.03
	mfeat-mor	4	NA[f]	0.52	0.59	NA[f]	0.52	0.64
	post-operative	2	NA[f]	0.28	0.28	NA[f]	0.28	0.28
	w/d/l[d]		NA[f]	2/4/0[g]	0/5/1[g]	NA[f]	0/6/0[g]	1/5/1
	Time[e]		NA[f]	0.55	0.08	NA[f]	0.08	1211.97

[a]Normal density estimation using pooled covariance $\hat{\Sigma}$pool.

[b]Normal density estimation using sample covariances $\hat{\Sigma}_j$.

[c]Data type: Q–quantitative; C–categorical; M–mix of quantitative and categorical attributes.

[d]Number of wins (**w**), draws (**d**), and losses (**l**) across data sets for each classifier and each data type.

[e]Mean CPU time in seconds (training + classification time).

[f]Not applicable for data type.

[g]One data set had identical error rates across folds for these classifiers, so that this data set was not counted in computing **w, d**, and **l**.

Table 6

Summary of error rates, accuracy comparison results, and computation times for classifiers using empirical probability estimation.

Type[a]	Data	*V*-fold CV	PNB	NB	HNB	AODE	LBR	TAN
Q	bcw	2	0.05	0.35	0.16	0.32	0.32	0.32
	BG Med	2	0.58	0.53	0.53	0.81	0.81	0.81
	ionosphere	3	0.14	0.36	0.42	0.82	0.91	0.91
	iris	3	0.44	0.69	0.06	0.05	0.05	0.05
	letter	2	0.88	0.96	0.18	0.41	0.57	0.49
	MDACC	2	0.54	0.60	0.66	1.00	1.00	1.00
	new-thyroid	5	0.82	0.30	0.07	0.04	0.06	0.05
	pendigits	2	0.56	0.90	0.17	0.47	0.53	0.53
	satellite	5	0.24	0.77	0.64	0.65	0.72	0.75
	segment	3	0.49	0.87	0.57	0.70	0.87	0.68
	sonar	4	0.44	0.47	0.53	0.81	0.81	0.81
	vehicle	3	0.52	0.77	0.65	0.70	0.78	0.64
	wine	2	0.57	0.60	0.10	0.79	0.78	0.77
	w/d/l[b]		0/2/11	0/0/13	0/6/7	0/2/11	0/1/12	0/2/11
	Time[c]		773.80	0.18	2.98	1.55	1062.94	1.20
C	balance	5	0.32	0.59	0.15	0.13	0.09	0.09
	house-votes	5	0.13	0.39	0.12	0.13	0.14	0.13
	lung	3	0.67	0.81	0.41	0.81	0.81	0.78
	ttt	2	0.36	0.35	0.25	0.26	0.23	0.30
	w/d/l[b]		0/4/0	0/4/0	2/2/0	0/3/1	1/1/1	1/2/1
	Time[c]		2.56	4.51	0.14	0.13	0.87	0.16
M	adult	2	0.33	0.25	0.20	0.20	0.19	0.21
	glass	2	0.72	0.68	0.50	0.66	0.65	0.65
	heart	3	0.28	0.44	0.16	0.44	0.44	0.44
	hepatitis	3	0.51	0.19	0.21	0.20	0.14	0.13
	hypothyroid	2	0.05	0.05	0.04	0.36	0.37	0.36
	mfeat-mor	4	0.54	0.92	0.88	0.87	1	1
	post-operative	2	0.57	0.28	0.79	0.83	0.83	0.85

Type[a]	Data	V-fold CV	PNB	NB	HNB	AODE	LBR	TAN
	w/d/l[b]		0/6/1	0/4/2[d]	2/2/3	0/5/2	1/4/2	1/4/2
	Time[c]		0.54	0.06	0.57	0.45	1576.90	0.39

[a]Data type: Q – quantitative; C – categorical; M – mix of quantitative and categorical attributes.

[b]Number of wins (**w**), draws (**d**), and losses (**l**) across data sets for each classifier and each data type.

[c]Mean CPU time in seconds (training + classification time).

[d]One data set had identical error rates across folds for these classifiers, so that this data set was not counted in computing **w**, **d**, and **l**.