IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received February 10, 2022, accepted February 24, 2022, date of publication February 28, 2022, date of current version March 4, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3155231

Exponential Loss Minimization for Learning Weighted Naive Bayes Classifiers

TAEHEUNG KIM AND JONG-SEOK LEE
Department of Industrial Engineering, Sungkyunkwan University, Suwon 16419, Republic of Korea
Corresponding author: Jong-Seok Lee (jongseok@skku.edu)

This work was supported in part by the National Research Foundation of Korea (NRF) funded by the Korean Government (Ministry of Science and ICT (MSIT)) under Grant 2019R1A4A1024732 and Grant 2020R1A5A1019649, and in part by the Institute for Information and Communications Technology Planning & Evaluation (IITP) funded by the Korean Government (MSIT) under Grant 20210002920012002.

ABSTRACT The naive Bayesian classification method has received significant attention in the field of supervised learning. This method has an unrealistic assumption in that it views all attributes as equally important. Attribute weighting is one of the methods used to alleviate this assumption and consequently improve the performance of the naive Bayes classification. This study, with a focus on nonlinear optimization problems, proposes four attribute weighting methods by minimizing four different loss functions. The proposed loss functions belong to a family of exponential functions that makes the optimization problems more straightforward to solve, provides analytical properties of the trained classifier, and allows for the simple modification of the loss function such that the naive Bayes classifier becomes robust to noisy instances. This research begins with a typical exponential loss which is sensitive to noise and provides a series of its modifications to make naive Bayes classifiers more robust to noisy instances. Based on numerical experiments conducted using 28 datasets from the UCI machine learning repository, we confirmed that the proposed scheme successfully determines optimal attribute weights and improves the classification performance.

INDEX TERMS Attribute weighting, classification, exponential loss, naive bayes, nonlinear optimization.

I. INTRODUCTION

Based on the Bayesian decision theorem, a Bayesian classifier predicts a test instance as a class that has the highest membership probability. This implies that learning a Bayesian classifier involves estimating the prior and posterior distributions from the training data. When training a Bayesian classifier, to estimate the posterior distributions, knowledge of the relationships among attributes is required. In particular, Bayesian network classifiers, also called Bayesian belief networks [1], require background knowledge in the form of a graph structure. Determining the optimal structure of Bayesian networks is a well-known NP-hard problem.

Among Bayesian classifiers, the naive Bayesian classifier has the simplest structure as it assumes that all attributes in the training data are conditionally independent of each other and equally important in determining the classes. These assumptions prove advantageous for naive Bayes in that it is easy to implement and can be trained with a small dataset. Although naive Bayes has strong assumptions on training data, it has been the focus of many researchers and practitioners because it often performs remarkably well in many domains [2], such as rRNA sequence classification [3], anti-spam filtering, identifying data correctness in wireless sensor networks [4]–[6], document classification [7], and sentiment classification in e-commerce product reviews [8].

Owing to its unrealistic assumptions, naive Bayes sometimes exhibits poor performance. There are several approaches for improving the classification accuracy of naive Bayes [9]. One promising approach is the attribute weighting technique, called weighted naive Bayes, which works by assigning different weights to each attribute to alleviate the assumption that all attributes are equally important. Let $w(j)$ be the weight of the jth attribute. If we restrict $w(j)$ to be either one or zero, then the attribute weighting becomes attribute selection, where one indicates inclusion of the attribute, and zero indicates exclusion of the attribute. Therefore, in the approaches of attribute weighting, the value of $w(j)$ is relaxed

The associate editor coordinating the review of this manuscript and approving it for publication was Sergio Consoli.

to a real number (usually a positive number for interpretation purposes). In this study, we mainly discuss the weighting approach, while still considering attribute selection by allowing $w(j)$ to take the zero value if necessary.

For the weighted naive Bayes to be effective, it is important to assign appropriate weights to each attribute. This has been widely studied by many researchers and practitioners with respect to attribute weighting methods, and they are divided into three groups: filter, wrapper, and embedded approaches [10]. Given a training dataset, the simplest method to assign attribute weights is to measure the importance of each attribute using external criteria and apply the measured importance directly to each attribute before learning a naive Bayes classifier. Such methods are filter approaches. In filter algorithms, calculating the attribute importance or selecting significant attributes is performed as a pre-processing step. Because it is separated from learning algorithms for classification or regression, it is relatively faster than wrapper and embedded approaches. One method to calculate the importance of an attribute is using the Kullback-Leibler measure for the amount of information [11]. A decision tree is often used to measure the importance of an attribute [12], in which the weight of the jth attribute is calculated as $1/\sqrt{d_j}$, where d_j is the depth at which the jth attribute first appears in a fully grown tree. The receiver operating characteristic (ROC) [13] can be a good choice for weight values when training data have a class imbalance problem [14]. Motivated by the fact that the area under the ROC curve (AUC) can be interpreted as a class-separating ability of each attribute, the study in [15] used the AUC scores as attribute weights in the weighted naive Bayes framework. The correlation-based feature selection (CFS) [16] is a widely used method that composes a subset of features based on their correlations. The main assumption to employ CFS is that important features are highly correlated with a class label but almost uncorrelated with other features. The research employing CFS for the weighted naive Bayes was proposed in that redundant features are removed with the expectation that the conditional independence assumption of the original naive Bayes can be relaxed [17]. Unlike other weighted naive Bayes research, it imposes the weights when estimating both prior and posterior probabilities. Another correlation-based weighting approach, CFW (correlation-based feature weighting), was proposed in [18]. Similar to CFS, CFW assumes that highly predictive features would be highly correlated with a class label (maximum mutual relevance) but uncorrelated with other attributes (minimum mutual redundancy) for naive Bayes. The mutual information was used to measure each attribute-class correlation and attribute-attribute correlations.

Although the aforementioned filter approaches provided reasonable attribute weights and successful experimental results, there is no guarantee of improvement on the classification accuracy of naive Bayes because attribute weighting is performed independently of training a classifier. To obtain weights that are optimal with respect to classification performance, wrapper approaches have been proposed. They assign attribute weights or select a subset of attributes in a heuristic optimization manner by utilizing a learning algorithm of interest in the training step with the learning objective of maximizing predictive power. In an earlier study, the selective Bayesian classifier (SBC) [19], a stepwise attribute selection technique, was combined with naive Bayes. To maximize classification accuracy, the algorithm repeatedly inserts important attributes or deletes irrelevant ones. The selective naive Bayes (SNB) algorithm [20] was proposed to overcome the heavy computational complexity of SBC. It first calculates mutual information with the class of each attribute and selects the attributes in descending order of the mutual information. In another wrapper approach for finding optimal attribute weights, the differential evolutionary algorithm was applied to determine the optimal weight values [21].

Wrapper approaches usually require a large amount of computational time because they repeatedly train and evaluate a classifier. Another weakness is that they need to prepare a validation set to evaluate a classifier within iterative procedures, which could be a problem with a small amount of data. Embedded approaches are designed to train a classifier and simultaneously assign attribute weights (or select important attributes). A well-known embedded approach is the decision tree [22], [23], which is suitable for a small dataset because the given dataset does not need to be divided into training and validation sets. The main idea of the embedded approaches is to formulate attribute weighting as an optimization problem, in which the objective function is the performance of a classifier and the decision variables are the attribute weights. The weighted naive Bayes based on the gradient-based L-BFGS-M method [24] was proposed by introducing two objective functions: conditional log-likelihood (CLL) and mean square error (MSE) [25], which focus on maximizing the likelihood of data from a probability perspective and minimizing predictive error from a classification perspective. This method was extended by giving different attribute weights to different class labels [26]. A matrix of weights which size is $n_c \times m$ (n_c: the number of class, m: the number of attributes) must be constructed to implement this idea. The superior performance was shown by conducting benchmark tests with real-world datasets. Another objective function, which maximizes the difference between membership probabilities for correctly-classified and misclassified instances, was introduced in [27] for the weighted naive Bayes. This research also proposed different weight vectors for different classes.

In the context of embedded approaches, we propose an exponential loss minimization method to learn weighted naive Bayes classifiers. We attempt to investigate the intended characteristic of each loss function such as robustness to noisy instances and linkage from our research to the related studies. Beginning with a typical exponential loss of the classification margin, this study sequentially introduces four different loss functions, where each function has its own analytical reasoning. Because the loss functions are evaluated

within a linear formulation of weighted naive Bayes, this study only considers binary classification.

The remainder of this paper is organized as follows. Section II reviews the naive Bayesian classification and mathematical formulation of the weighted naive Bayes. In Section III, we propose four different loss functions and their solutions for attribute weights. This section provides their analytical properties and empirical evidence from illustrative experiments. To confirm the performance of the proposed weighted naive Bayes classifiers, numerical experiments are presented in Section IV. Section V concludes the paper and discusses future research directions.

II. BACKGROUND

A. NAIVE BAYES CLASSIFIER

A Bayesian classifier predicts the class labels of unknown instances with a corresponding maximum posterior probability. Let m be the number of attributes and $y \in \{-1, 1\}$ be the class label. The predicted class $\hat{y}_i$ of the ith unseen instance $\boldsymbol{x}_i = [x_{i1}, x_{i2}, \ldots x_{im}]$ is then determined as in (1).

$$\hat{y}_i = \arg\max_y P(y)P(\boldsymbol{x}_i|y) \tag{1}$$

Estimating the posterior probability $P(\boldsymbol{x}_i|y)$ increases the model complexity and requires a large amount of training data. Therefore, naive Bayes simply assumes that all attributes are conditionally independent. With this assumption, we use (2) instead of (1).

$$\hat{y}_i = \arg\max_y P(y)\prod_{j=1}^{m} P(x_{ij}|y) \tag{2}$$

When the jth attribute is numerical, there are several methods to compute $P(x_{ij}|y)$, such as changing the jth attribute to a discrete attribute by applying the histogram method or interpreting it as some other probability distribution. The empirical distribution can be estimated using the kernel density estimation [28]. In this study, we assume that the numerical attribute follows a normal distribution. (3) is used as an estimation of the posterior probability, where $\mu_{(j|y)}$ and $\sigma_{(j|y)}$ are the mean and standard deviations of the jth attribute, respectively, given class label y.

$$P(x_{ij}|y) \cong \frac{1}{\sqrt{2\pi}\sigma_{(j|y)}} \exp\left[-\frac{\{x_{ij} - \mu_{(j|y)}\}^2}{2\sigma^2_{(j|y)}}\right] \tag{3}$$

For a categorical attribute, the posterior probability is estimated according to the proportion of the number of training instances in class y that take a particular value x_{ij}, denoted by $n(x_{ij}|y)$, to the total number of instances in class y, $n(y)$. After applying Laplace's correction, the estimated posterior probability is shown in (4), where n_j is the number of categories (cardinality) of the jth attribute.

$$P(x_{ij}|y) \cong \frac{n(x_{ij}|y) + 1}{n(y) + n_j} \tag{4}$$

B. WEIGHTED NAIVE BAYES CLASSIFIER

Since the assumption of equally important attributes is rarely satisfied in practice, several studies have attempted to relax this assumption [9]. Attribute weighting is one of the most frequently used relaxation methods. The underlying principle of weighted naive Bayesian classification is that some attributes are more (or less) important than others in a classification task. This leads to the modification of (2), which is given in (5).

$$\hat{y}_i = \arg\max_y P(y)\prod_{j=1}^{m} P_{w(j)}(x_{ij}|y), \tag{5}$$

where $P_{w(j)}(x_{ij}|y)$ represents the weighted posterior probability, which is usually defined in an exponential form as in (6).

$$P_{w(j)}(x_{ij}|y) := P(x_{ij}|y)^{w(j)} \tag{6}$$

Note that the weight of the jth attribute, $w(j)$, representing the significance of the attribute, can be any positive number. If it is allowed to take either 0 or 1, (6) is reduced to an attribute selection problem. In the binary class case, which is the main focus of this study, (5) can be described by an "If ... then ..." rule, as in (7).

$$\begin{aligned}\text{If } P(y_i = 1)\prod_{j=i}^{m} P_{w(j)}(x_{ij}|y_i = 1) > \\ P(y_i = -1)\prod_{j=i}^{m} P_{w(j)}(x_{ij}|y_i = -1),\end{aligned} \tag{7}$$

$$\text{then } \boldsymbol{x}_i \text{ is classified as "1" (otherwise, "}-1\text{").}$$

Using (6) and taking the logarithm, the weighted naive Bayes becomes a linear function of the weights, as shown in (8), with the classification rule.

$$\hat{f}(\boldsymbol{x}_i) = P_0 + \boldsymbol{w}\boldsymbol{P}_{\boldsymbol{x}i},$$
$$\boldsymbol{x}_i \text{ is classified as } \begin{cases} 1, & \text{if } \hat{f}(\boldsymbol{x}_i) > 0 \\ -1, & \text{if } \hat{f}(\boldsymbol{x}_i) \le 0, \end{cases} \tag{8}$$

where

$$P_0 = \log\frac{P(y=1)}{P(y=-1)},$$
$$\boldsymbol{P}_x = \left[\log\frac{P(x_{i1}|y_i=1)}{P(x_{i1}|y_i=-1)}, \ldots, \log\frac{P(x_{im}|y_i=1)}{P(x_{im}|y_i=-1)}\right],$$
$$\boldsymbol{w} = [w(1), w(2), \ldots, w(m)].$$

III. PROPOSED METHOD

In this section, we propose an optimization approach for learning weighted naive Bayes classifiers. The optimization problem is formulated in that training a weighted naive Bayes classifier involves finding optimal attribute weights that maximize the overall accuracy of the classifier. To this end, we first introduce a typical exponential loss of classification margin and then sequentially introduce three other loss functions, namely, binomial deviance, modified binomial deviance, and generalized binomial deviance loss functions.

With all the loss functions, the optimization problems can be solved using gradient-based nonlinear optimization methods, such as quasi-Newton methods [29].

A. EXPONENTIAL LOSS MINIMIZATION

To obtain the optimal weights that maximize the classification accuracy, an exponential loss can be used. With the given ith instance $\boldsymbol{x}_i$ and its class label $y_i \in \{-1, 1\}$, the exponential loss $L_{exp}(\boldsymbol{x}_i, y_i)$ is defined as an exponential form of the classification margin, as shown in (9).

$$L_{exp}(\boldsymbol{x}_i, y_i) = \exp\{-y_i f(\boldsymbol{x}_i)\} \tag{9}$$

With a training set of n instances, the total exponential loss is given as a summation of (9). One of the proposed weighting methods, namely exponential naive Bayes (ENB), is defined in (10), which is the solution for minimizing the total exponential loss. Once we find the optimal weights, we can use (8) to classify new instances.

$$\boldsymbol{w}_{ENB} = \arg\min_{\boldsymbol{w}} \sum_{i=1}^{n} \exp\{-y_i(P_0 + \boldsymbol{w}\boldsymbol{P}_{xi})\} \tag{10}$$

The validity of the ENB can be demonstrated as follows. It is straightforward to show that (10) is equivalent to (11).

$$\boldsymbol{w}_{ENB} = \arg\min_{\boldsymbol{w}} [T_{FNR} + T_{FPR}] \tag{11}$$

because

$$T_{FNR} = \sum_{i|y_i=+1} \frac{P(y=-1)}{P(y=+1)} \prod_{j=1}^{m} \left\{ \frac{P(x_{ij}|y_i=-1)}{P(x_{ij}|y_i=+1)} \right\}^{w(j)},$$

$$T_{FPR} = \sum_{i|y_i=-1} \frac{P(y=+1)}{P(y=-1)} \prod_{j=1}^{m} \left\{ \frac{P(x_{ij}|y_i=+1)}{P(x_{ij}|y_i=-1)} \right\}^{w(j)}.$$

It is obvious that both T_{FNR} and T_{FPR} are non-negative and measure incorrect classification of instances. If a positive instance is misclassified as a negative class, T_{FNR} increases. Likewise, if a negative instance is incorrectly predicted, T_{FPR} increases. Hence, the optimal attribute weights of w_{ENB} simultaneously minimize the ''false negative rate'' and ''false positive rate'' of a classifier.

As an ideal case, if infinite training instances are given, it is known that the minimization of exponential loss is equivalent to the minimization of binomial deviance loss [30], which is given in (12).

$$L_{dev}(\boldsymbol{x}_i, y_i) = \log\left[1 + \exp\{-2y_i f(\boldsymbol{x}_i)\}\right] \tag{12}$$

However, with a finite training dataset, the optimization result of binomial deviance loss minimization must be different to that of exponential loss minimization in terms of scale. As $y_i f(\boldsymbol{x}_i) \to -\infty$, the value of (9) increases exponentially, whereas that of (12) increases almost linearly. Therefore, we can expect that L_{dev} is less sensitive than L_{exp} to noisy instances, examples of which are instances that are difficult to classify. In (13), our second proposed model, namely deviance naive Bayes (DNB), is depicted. We can train a weighted naive Bayes classifier by minimizing the total binomial deviance loss.

$$\boldsymbol{w}_{DNB} = \arg\min_{\boldsymbol{w}} \sum_{i=1}^{n} \log\left[1 + \exp\{-2y_i(P_0 + \boldsymbol{w}\boldsymbol{P}_{\boldsymbol{x}i})\}\right] \tag{13}$$

Observing that the loss increase of L_{dev} is slower than that of L_{exp} as the classification margin increases, we additionally propose to remove the constant '2' from the classification margin $y_i(P_0 + \boldsymbol{w}\boldsymbol{P}_{\boldsymbol{x}i})$ in (13) to further reduce the loss increase. We believe that this modification would make the trained weighted naive Bayes classifier less sensitive to noisy instances. We name it the log-likelihood naive Bayes (LNB), which is shown in (14).

$$\boldsymbol{w}_{LNB} = \arg\min_{\boldsymbol{w}} \sum_{i=1}^{n} \log\left[1 + \exp\{-y_i(P_0 + \boldsymbol{w}\boldsymbol{P}_{\boldsymbol{x}i})\}\right] \tag{14}$$

This simple modification explains why the solution to the optimization problem in (14) is reasonable, as shown in the following theorem.

Theorem 1: $\boldsymbol{w}_{LNB}$ is the maximum likelihood estimator of the attribute weights.

Proof: By converting (14) into a maximization problem and removing the logarithm, we obtain (15).

$$\boldsymbol{w}_{LNB} = \arg\max_{\boldsymbol{w}} \prod_{i=1}^{n} \frac{1}{1 + \exp\{-y_i(P_0 + \boldsymbol{w}\boldsymbol{P}_{\boldsymbol{x}i})\}} \tag{15}$$

(15) can be rewritten as

$$\boldsymbol{w}_{LNB} = \arg\max_{\boldsymbol{w}} \prod_{i|y_i=1} \frac{\hat{f}(\boldsymbol{x}_i)}{1 + \exp\hat{f}(\boldsymbol{x}_i)} \prod_{i|y_i=-1} \frac{1}{1 + \exp\hat{f}(\boldsymbol{x}_i)}, \tag{16}$$

resulting in the product of two groups of terms, where each group of terms corresponds to one of the classes.

By expanding (16), we obtain

$$\begin{aligned}
\boldsymbol{w}_{LNB} &= \arg\max_{\boldsymbol{w}} \prod_{i|y_i=1} \frac{P(\boldsymbol{x}_i, y_i=1)}{P(\boldsymbol{x}_i, y_i=1) + P(\boldsymbol{x}_i, y_i=-1)} \\
&\quad \times \prod_{i|y_i=-1} \frac{P(\boldsymbol{x}_i, y_i=-1)}{P(\boldsymbol{x}_i, y_i=1) + P(\boldsymbol{x}_i, y_i=-1)} \\
&= \arg\max_{\boldsymbol{w}} \prod_{i|y_i=1} P(y_i=1|\boldsymbol{x}_i) \prod_{i|y_i=-1} P(y_i=-1|\boldsymbol{x}_i) \\
&= \arg\max_{\boldsymbol{w}} \prod_{i=1}^{n} P(y_i|\boldsymbol{x}_i).
\end{aligned} \tag{17}$$

The last term in (17) contains a product of the posterior probabilities for all of the training instances. Therefore, $\boldsymbol{w}_{LNB}$ is the maximum likelihood estimator of the attribute weights. □

By Theorem 1, we could deduce that the LNB is eventually equivalent to the CLL in [25].

Further, we focus on the multiplier of the classification margin $y_i(P_0 + \boldsymbol{w}\boldsymbol{P}_{\boldsymbol{x}i})$ in (13). Because we change the multiplier of the margin from -2 to -1 for the robustness of

a classifier to noisy instances, we can apply a generalized multiplier, namely $-\alpha$, and let the nonlinear optimization routine find its best value according to a given training set. Thus, it is reasonable to interpret the multiplier as a tuning parameter for the noise level in the data. By introducing $-\alpha$, the generalized DNB (GDNB) is given in (18).

$$\boldsymbol{w}_{GDNB} = \arg\min_{\boldsymbol{w}} \sum_{i=1}^{n} \log\left[1 + \exp\left\{-\alpha y_i (P_0 + \boldsymbol{w}\boldsymbol{P}_{\boldsymbol{x}i})\right\}\right] \tag{18}$$

By defining GDNB, DNB and LNB can be seen as the special cases of GDNB with $\alpha = 2$ and $\alpha = 1$, respectively. α is a tuning parameter that controls the penalty of misclassified instances when evaluating the loss function, which means that if α is high, it is likely to focus more on outliers or noisy instances. Because the noise level would be different for different data, its optimal value should be determined such that it maximizes the classification performance of a trained classifier. To find the optimal attribute weights and optimal tuning parameter α simultaneously, (18) is reformulated as (19).

$$\begin{aligned}\boldsymbol{w}_{GDNB} &= \arg\min_{\alpha,\boldsymbol{w}} \sum_{i=1}^{n} \log\left[1 + \exp\left\{-\alpha y_i (P_0 + \boldsymbol{w}\boldsymbol{P}_{\boldsymbol{x}i})\right\}\right] \\ &= \arg\min_{w_0,\boldsymbol{w}'} \sum_{i=1}^{n} \log\left[1 + \exp\left\{-y_i (w_0 P_0 + \boldsymbol{w}'\boldsymbol{P}_{\boldsymbol{x}i})\right\}\right]\end{aligned} \tag{19}$$

From the right-most side of (19), we observe that finding α is equivalent to assigning weights to not only posterior probabilities, $P(x_{ij}|y)$, but also prior probability, $P(y)$. According to the context in which attribute weights indicate the degree of importance of the attributes, w_0 can be defined as the confidence of prior information. With GDNB, we need to modify the classification rule in (8), as shown below.

$$\hat{f}(\boldsymbol{x}_i) = w_0 P_0 + \boldsymbol{w}\boldsymbol{P}_{\boldsymbol{x}i},$$
$$\boldsymbol{x}_i \text{ is classified as } \begin{cases} 1, & \text{if } \hat{f}(\boldsymbol{x}_i) > 0 \\ -1, & \text{if } \hat{f}(\boldsymbol{x}_i) \leq 0 \end{cases} \tag{20}$$

In addition to the above loss functions, we introduce non-negative constraints, which are shown in (21), for all four sets of attributes. The constraints are intended to provide interpretation ability to the proposed methods because a negative weight cannot be interpreted in terms of importance.

$$w_0 \geq 0, \quad \boldsymbol{w} \geq \boldsymbol{0} \tag{21}$$

By minimizing the proposed loss functions with respect to (10), (13), (14), and (19) with the constraints in (21), the proposed methods, namely, ENB, DNB, LNB, and GDNB, are trained.

The proposed learning algorithm is briefly described in Algorithm 1. Most of the computational time of the proposed method depends on the L-BFGS-B because training an original naive Bayesian classifier is done only once and thus is a relatively small burden. The computational cost of one iteration of the L-BFGS-B update is $O(p^2m)$ in the worst case [24], where p is the control parameter for memory allocation. In this study, we set $p = 5$ as recommended in [24]. The proposed loss functions defined by the summation over training instances should be evaluated in each iteration of the L-BFGS-B update. Therefore, the computational complexity of one iteration in the proposed algorithm becomes $O(p^2mn)$.

Algorithm 1 Attribute Weighting via Loss Minimization

Require: $D = \{\boldsymbol{x}_i, y_i\}_{1 \leq i \leq N}$
  $P_0, \boldsymbol{P}_{\boldsymbol{x}} = \text{NaiveBayes}(D)$ ▷ train an original naive Bayes.
  OptFun = Choose among {ENB, DNB, LNB, GDNB}
  if OptFun = GDNB **then**
    $\boldsymbol{w}$ = Initialize weights vector as $\boldsymbol{1}_{\{1\times(m+1)\}}$
    $\text{LB} = \boldsymbol{0}_{\{1\times(m+1)\}}$
  else
    $\boldsymbol{w}$ = Initialize weights vector as $\boldsymbol{1}_{(1\times m)}$
    $\text{LB} = \boldsymbol{0}_{(1\times m)}$
  end if
  $\boldsymbol{w}^*_{OptFun} = \text{L-BFGS-B}(\boldsymbol{w}, P_0, \boldsymbol{P}_{\boldsymbol{x}}, D, \text{OptFun}, \text{LB})$
  return $\boldsymbol{w}^*_{OptFun}$

B. COMPARISON OF LOSS FUNCTIONS

This subsection compares and interprets the loss functions using an illustrative example. Figure 1 depicts three loss functions, namely, exponential, binomial deviance, and modified binomial deviance loss functions, at varying classification margins yf, which monotonically decrease and have non-negative values. The exponential loss is always greater than the other loss functions and increases exponentially in the negative margin, whereas the others increase less rapidly. This means that when noisy instances, which are difficult to classify, are included in a training set, it is expected that a classifier trained by minimizing the exponential loss would be more affected by those instances. Therefore, an overfitting problem could possibly occur because their losses account for a large part of the total loss that an employed nonlinear optimizer attempts to reduce.

The value of the binomial deviance is higher and lower than that of the modified binomial deviance at the negative and positive margins, respectively. This implies that the modified binomial deviance is less sensitive to misclassified instances than others, while it assigns a larger amount of loss to misclassified instances than to correctly classified instances; therefore, the less-steep losses are expected to be more robust to noisy data, as intended.

To show the difference among the loss functions, we generated an illustrative 2D example consisting of two classes that are easily separable but have few opposite class instances (marked 1 to 5 in Figure 2(a)) in each of the class regions. Hence, we intentionally generated instances that would be incorrectly classified to observe their losses. We then trained four naive Bayes classifiers (NB, ENB, DNB, and LNB) from

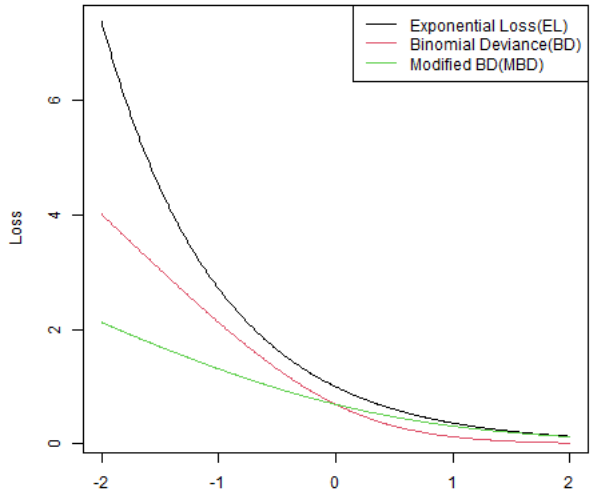
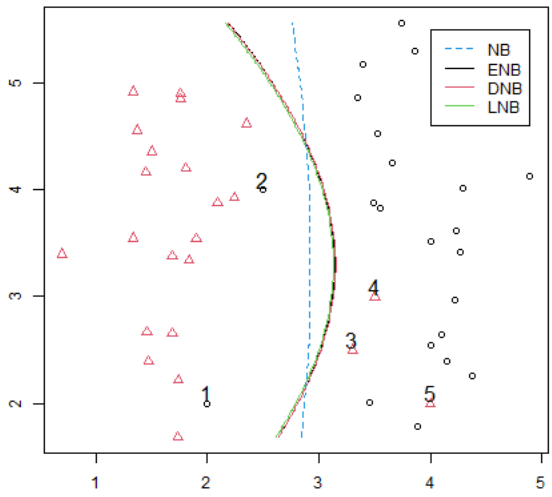
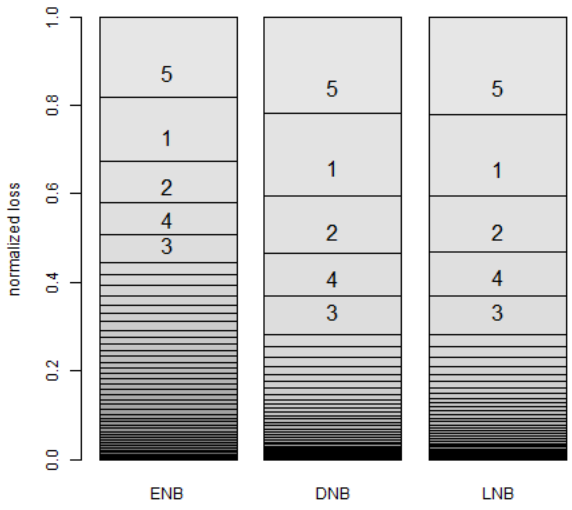


FIGURE 1. Exponential, binomial deviance, and modified binomial deviance loss functions.

the synthetic dataset. We did not include GDNB because its result would be similar to that of DNB or LNB for this small and simple example. Figure 2(a) depicts the decision boundaries for which the standard NB is distinct from the others. The instances marked 1 to 5 were misclassified by all the decision boundaries. Although the classification results are the same, the loss values of the instances varied across methods. More specifically, ENB showed a significantly different distribution of instance losses in comparison with DNB and LNB. Figure 2(b) shows the loss values of each training instance, which were normalized by the total loss such that the sum of the normalized losses became 1, for an effective comparison. The misclassified instances were marked from 1 to 5. Note that the loss values are the results after the optimization procedure, which implies that they are the already-minimized values. As can be seen from the figure, the misclassified instances of LNB and DNB account for more than 70% of the total loss, whereas ENB has less than 60%. This implies that ENB is more focused on instances 1 to 5 than DNB and LNB because, to minimize the total loss, it was beneficial to reduce the losses of the noisy instances more than those of other well-classified instances. This example shows that our reasoning regarding the robustness to noise in sequentially introducing the loss functions is valid. We provide more empirical evidence for the robustness in Section IV-C.



(a) Decision boundaries of standard NB and weighted NB classifiers commonly misclassifying five instances



(b) Loss value distributions of loss functions and loss proportion of misclassified instances

FIGURE 2. Effects of three loss functions: ENB, DNB, and LNB.

IV. NUMERICAL EXPERIMENTS

A. EXPERIMENTAL SETTING

To confirm the performance of the proposed weighted naive Bayes, numerical experiments were conducted based on 28 real datasets from the University of California, Irvine (UCI) machine learning repository [31]. The datasets are listed in Table 1. They were collected to evaluate classifiers in various circumstances in terms of the percentage of minority class instances, total number of instances, and total number of attributes. The datasets are sorted in ascending order of minority ratio; the dataset "Nursery" is the most imbalanced case (2.53%), whereas the dataset "Breast Cancer" has the most balanced class distribution (37.26%). Each dataset contains different numbers of instances ranging from 151 to 28,056 and different numbers of attributes, ranging from 4 to 64. Most datasets consist of either numerical or categorical attributes. Some datasets, such as "Chess," have both. To focus on binary classification, the datasets with more than two classes were converted into binary class cases by

assigning one class as a minority class and integrating the other classes into a majority class.

Based on the real-world datasets, we conducted comparative experiments with eleven naive Bayes methods, namely the standard naive Bayes (NB), selective Bayesian classifier (SBC) [19], Kullback-Leibler NB (KLNB) [11], weighted naive Bayes with tree induction (TreeNB) [12], deep feature weighting NB (DFWNB) [17], correlation based filter weighting NB (CBFWNB) [18], weighted naive Bayes with mean squared error loss (MSE) [25], and our proposed methods (ENB, DNB, LNB, and GDNB).

B. GENERAL PERFORMANCE

The experimental results are summarized in Table 2, which records the accuracy and relative improvement rates for each combination of a classifier and a dataset. The relative improvement rate (RI) in parentheses is calculated as $(Accuracy(\cdot) - Accuracy(NB))/Accuracy(NB) \times 100(\%)$, where $Accuracy(\cdot)$ is the accuracy of each method and $Accuracy(NB)$ is the accuracy of the standard NB, to determine how much improvement can be achieved by the weighted NB methods over the standard NB. The average RIs are shown in the last row of the table. Every experiment involved 10-fold cross-validation; thus, the accuracy reported in the table is the average of 10 accuracy values. The underlined numbers indicate the best (boldface) or worst (italic) accuracy for each dataset.

In three datasets (''Steel Plate Faults,'' ''Image Segmentation,'' and ''Wholesale Customers''), there was a significant improvement in the attribute weighting methods. For example, while NB correctly classified 66.66% of instances in the ''Steel Plate Faults'' dataset, the proposed DNB showed an accuracy of 93.6%, resulting in a 40.43% improvement. The correlation heatmaps of the three datasets are shown in Figures 4, 5, and 6. It is clear that the attributes in those datasets are highly correlated, which implies that some of the attributes are redundant for classification. From this example, we can see that the proposed method can improve the classification performance by attribute weighting for highly correlated data. In fact, compared with NB, one of the proposed methods showed better accuracy for all datasets except for ''Wall-Following Robot Navigation Data (4L).'' All other attribute weighting methods also performed worse than NB for this dataset. Considering that this dataset has the smallest number of attributes, it may be unnecessary to apply attribute weighting to the dataset. For the proposed methods, the attribute selection rate was additionally computed by the number of attributes with non-zero weights over total number of attributes in each dataset. As can be seen from Table 3, the selection rates are similar across the methods but different across the datasets. The dropped attributes by zero weights appeared in the most cases, while there was no attribute selection in four datasets (No. 1, 2, 16, and 24). Especially, we observed that only about 20% of attributes contributed to the classification task for the ''Multiple Attribute (mfeat-zer)'' dataset (No. 11).

From Table 2, we can observe that the attribute weighting methods generally outperformed the standard NB. Next, we compare the weighted naive Bayes by only considering the RI values. As can be seen from the last row of the table, the proposed methods showed the largest improvement compared with the standard NB by 10.65% on average. The average RI for the proposed methods was computed by taking the largest RI value from each row in the last four columns (ENB, DNB, LNB, and GDNB) and averaging them because this study provides four options to perform attribute weighting. The proposed methods were followed by MSE (10.26%), CBFWNB (9.83%), SBC (8.82%), DFWNB (8.29%), KLNB (6.18%), and TreeNB (5.99%) in sequence of performance. Note that KLNB, TreeNB, DFWNB and CBFWNB are filter-based methods, SBC is a wrapper method, and MSE is one of the embedded methods, as are the proposed methods. These performance results are consistent with previous works showing that it is more desirable to find the attribute weights with simultaneous consideration of classification performance, as discussed in the introduction section.

Other observations from the experiments were as follows: In the severely class-imbalanced datasets (''Nursery,'' ''Car Evaluation,'' and ''Letter Recognition''), the filter and wrapper methods (KLNB, TreeNB, and SBC) showed even worse performances than the standard NB. Among the proposed methods, DNB, LNB, and GDNB showed similar performances, and ENB performed worse. This implies that it would be preferable to perform a logarithm in a loss function instead of using a pure exponential loss, and our logical derivation from ENB to GDNB is reasonable. It is notable that MSE recorded the highest accuracy in 8 datasets, which means that minimizing the squared error is still a promising method to train a weighted naive Bayes classifier.

We conducted the Wilcoxon signed-ranks test [32] on the experimental results to convince the generalization performance of the proposed methods, and the test results are summarized in Table 4. Each cell contains the p-value for the hypothesis that the classification performance of a row classifier is different from that of a column classifier. The p-values below the chosen significance level 0.05 are underlined. The boldface means that a row classifier outperformed a column classifier while the italic implies the opposite. Table 4 shows that the GDNB outperformed every benchmark classifier except for CBFWNB, MSE, and LNB.

C. ROBUSTNESS TEST

This subsection evaluates the robustness of the proposed methods. From one point of view, a robust classifier refers to a classifier that is minimally influenced by outliers in the training data. In this experiment, noise data were intentionally added to a training set to observe the degradation of the classification performance. If the degradation is small, we can consider the classifier to be robust.

As in the previous experiments, 10-fold cross-validation was employed. After extracting $p\%$ (noise rate) random instances from the folded training set, they were converted

TABLE 1. Description of datasets.

No	Dataset	% minority	Class description	# instances	# attributes	
					Continuous	Discrete
1	Nursery	2.53	(Very recom, remainder)	12,960	0	8
2	Car Evaluation	3.76	(vgood, remainder)	1,728	0	6
3	Letter Recognition	3.95	(A, remainder)	20,000	16	0
4	Page Blocks Classification	6.01	(2, remainder)	5,473	10	0
5	Wall-Following Robot Navigation Data (4L)	6.01	(slight-left-turn, remainder)	5,456	4	0
6	Wall-Following Robot Navigation Data (24L)	6.01	(slight-left-turn, remainder)	5,456	24	0
7	Chess (King-Rook vs. King)	7.08	(ten, remainder)	28,056	3	3
8	Steel Plate Faults	8.14	(Pastry, remainder)	1,941	27	0
9	Cardiotocography	9.27	(pattern10, remainder)	2,126	21	0
10	Multiple Attribute (mfeat-kar)	10	(0, remainder)	2,000	64	0
11	Multiple Attribute (mfeat-zer)	10	(0, remainder)	2,000	47	0
12	Pen-Based Recognition of Handwritten Digits	10.38	(0, remainder)	3,498	16	0
13	Banknote Authentication	13.99	(1, remainder)	886	4	0
14	Image Segmentation	14.29	(BRICKFACE, remainder)	2,310	19	0
15	Turkey Student Evaluation	14.45	(6, remainder)	5,820	28	0
16	Wall-Following Robot Navigation Data (4R)	15.13	(slight-right-turn, remainder)	5,456	4	0
17	Wall-Following Robot Navigation Data (24R)	15.13	(slight-right-turn, remainder)	5,456	24	0
18	Yeast	16.44	(mit, remainder)	1,484	8	0
19	Wholesale Customers	17.5	(1, remainder)	440	6	0
20	Wine Quality - White	17.97	(7, remainder)	4,898	11	0
21	Contraceptive Method Choice	22.61	(2, remainder)	1,473	9	0
22	Abalone	32.13	(I, remainder)	4,177	8	0
23	Glass Identification	32.71	(1, remainder)	214	9	0
24	Teaching Assistant Evaluation	33.11	(2, remainder)	151	5	0
25	QSAR Biodegradation	33.74	(RB, NRB)	1,055	41	0
26	Tic-Tac-Toe Endgame	34.66	(negative, positive)	958	0	9
27	MAGIC Gamma Telescope	35.16	(h, remainder)	19,020	10	0
28	Breast Cancer	37.26	(0, 1)	569	30	0

TABLE 2. Experimental results (accuracy and relative improvement).

No	NB	SBC	KLNB	TreeNB	DFWNB	CBFWNB	MSE	ENB	DNB	LNB	GDNB
1	98.04 (-)	97.48 (-0.57)	97.48 (-0.57)	97.48 (-0.57)	98.3 (0.27)	*97.45 (-0.6)*	98.04 (0)	**98.94 (0.92)**	98.9 (0.88)	98.9 (0.88)	98.9 (0.88)
2	97.92 (-)	*96.3 (-1.65)*	*96.3 (-1.65)*	*96.3 (-1.65)*	**99.42 (1.53)**	96.51 (-1.44)	97.92 (0)	99.07 (1.17)	99.07 (1.17)	98.84 (0.94)	98.96 (1.06)
3	98.68 (-)	*96.06 (-2.66)*	*96.06 (-2.66)*	98.94 (0.26)	98.4 (-0.28)	98.7 (0.02)	**99.24 (0.57)**	96.15 (-2.56)	98.96 (0.28)	99.09 (0.42)	99.11 (0.44)
4	96.33 (-)	96.97 (0.66)	95.8 (-0.55)	96.45 (0.12)	*94.88 (-1.51)*	95.25 (-1.12)	**97.02 (0.72)**	96.14 (-0.2)	96.6 (0.28)	95.98 (-0.36)	96.16 (-0.18)
5	98.64 (-)	98.63 (-0.01)	96.24 (-2.43)	98.44 (-0.2)	99.63 (1)	**99.82 (1.2)**	98.64 (0)	*71.13 (-27.89)*	95.71 (-2.97)	95.97 (-2.71)	96.15 (-2.52)
6	*91.11 (-)*	93.99 (3.16)	94.01 (3.18)	92.08 (1.06)	96.52 (5.94)	96.7 (6.14)	**96.74 (6.18)**	94.72 (3.96)	96.08 (5.45)	96.22 (5.61)	96.17 (5.55)
7	92.9 (-)	92.92 (0.02)	92.92 (0.02)	92.92 (0.02)	*89.77 (-3.37)*	**92.94 (0.04)**	92.92 (0.02)	92.92 (0.02)	92.76 (-0.15)	92.92 (0.02)	92.92 (0.02)
8	*66.66 (-)*	91.86 (37.8)	91.86 (37.8)	77.95 (16.94)	80.41 (20.63)	84.54 (26.82)	93.51 (40.28)	75.58 (13.38)	**93.61 (40.43)**	93.51 (40.28)	93.51 (40.28)
9	92.01 (-)	90.74 (-1.38)	*89.23 (-3.02)*	90.17 (-2)	90.14 (-2.03)	90.14 (-2.03)	93.84 (1.99)	92.81 (0.87)	93.84 (1.99)	**94.03 (2.2)**	**94.03 (2.2)**
10	98.35 (-)	98.6 (0.25)	*90 (-8.49)*	97.8 (-0.56)	98 (-0.36)	**99 (0.66)**	98.9 (0.56)	97.95 (-0.41)	98.15 (-0.2)	98.35 (0)	98.5 (0.15)
11	*92.35 (-)*	98.7 (6.88)	92.65 (0.32)	**99.1 (7.31)**	95.5 (3.41)	95 (2.87)	**99.1 (7.31)**	94.2 (2)	98.95 (7.15)	99.05 (7.26)	**99.1 (7.31)**
12	96.4 (-)	97.43 (1.07)	*90.82 (-5.79)*	97.03 (0.65)	93.14 (-3.38)	92.86 (-3.67)	**98.34 (2.01)**	94.02 (-2.47)	97.91 (1.57)	98.14 (1.8)	98.03 (1.69)
13	91.75 (-)	94.25 (2.72)	*91.19 (-0.61)*	92.55 (0.87)	92.05 (0.33)	92.05 (0.33)	94.13 (2.59)	93.34 (1.73)	94.25 (2.72)	**94.34 (2.82)**	94.12 (2.58)
14	*82.86 (-)*	98.35 (18.69)	85.71 (3.44)	98.27 (18.6)	93.51 (12.85)	94.81 (14.42)	98.92 (19.38)	98.7 (19.12)	98.74 (19.16)	98.74 (19.16)	98.74 (19.16)
15	*71.54 (-)*	**85.55 (19.58)**	**85.55 (19.58)**	73.48 (2.71)	75.43 (5.44)	76.29 (6.64)	**85.55 (19.58)**	**85.55 (19.58)**	**85.55 (19.58)**	**85.55 (19.58)**	**85.55 (19.58)**
16	*90.67 (-)*	96.41 (6.33)	91.95 (1.41)	94.28 (3.98)	**99.08 (9.28)**	**99.08 (9.28)**	95.58 (5.42)	94.59 (4.32)	95.65 (5.49)	94.68 (4.42)	95.6 (5.44)
17	*65.98 (-)*	84.86 (28.61)	84.86 (28.61)	83.1 (25.95)	96.15 (45.73)	**97.07 (47.12)**	87.74 (32.98)	82.46 (24.98)	87.11 (32.02)	87.41 (32.48)	87.61 (32.78)
18	85.5 (-)	**89.28 (4.42)**	85.78 (0.33)	88.33 (3.31)	87.16 (1.94)	87.16 (1.94)	88.27 (3.24)	*84.49 (-1.18)*	87.66 (2.53)	87.93 (2.84)	87.93 (2.84)
19	*49.65 (-)*	**82.71 (66.59)**	**82.71 (66.59)**	82.48 (66.12)	81.82 (64.79)	81.82 (64.79)	**82.71 (66.59)**	82.48 (66.12)	82.25 (65.66)	82.48 (66.12)	82.48 (66.12)
20	*72.56 (-)*	81.91 (12.89)	82.05 (13.08)	78.38 (8.02)	75.92 (4.63)	79.8 (9.98)	82.17 (13.24)	82.07 (13.11)	82.25 (13.35)	**82.3 (13.42)**	82.25 (13.35)
21	68.88 (-)	77.38 (12.34)	77.38 (12.34)	75 (8.89)	*65.31 (-5.18)*	74.83 (8.64)	77.58 (12.63)	77.38 (12.34)	77.31 (12.24)	77.99 (13.23)	**78.33 (13.72)**
22	78.76 (-)	80.7 (2.46)	79.86 (1.4)	79.22 (0.58)	80.38 (2.06)	79.19 (0.55)	**81.54 (3.53)**	*76.77 (-2.53)*	78.76 (0)	81.2 (3.1)	81.18 (3.07)
23	*63.51 (-)*	72.56 (14.25)	64.82 (2.06)	66.25 (4.31)	76.19 (19.97)	76.19 (19.97)	77.37 (21.82)	76.42 (20.33)	76.9 (21.08)	**77.85 (22.58)**	76.82 (20.96)
24	64.67 (-)	64 (-1.04)	64.67 (0)	*60.67 (-6.19)*	80 (23.7)	**86.67 (34.02)**	64 (-1.04)	65.33 (1.02)	66.67 (3.09)	65.33 (1.02)	65.33 (1.02)
25	*75.04 (-)*	81.37 (8.44)	83.67 (11.5)	78.66 (4.82)	83.02 (10.63)	84.91 (13.15)	86.13 (14.78)	75.81 (1.03)	84.05 (12.01)	**86.8 (15.67)**	86.51 (15.29)
26	70.31 (-)	69.69 (-0.88)	*65.41 (-6.97)*	70.41 (0.14)	70.83 (0.74)	73.96 (5.19)	72.3 (2.83)	75.02 (6.7)	**75.23 (7)**	68.33 (-2.82)	68.86 (-2.06)
27	*72.71 (-)*	76.69 (5.47)	76.34 (4.99)	74.92 (3.04)	**79.97 (9.98)**	77.66 (6.81)	77.25 (6.24)	73.45 (1.02)	75.55 (3.91)	77.28 (6.29)	77.43 (6.49)
28	93.32 (-)	95.78 (2.64)	*92.62 (-0.75)*	94.38 (1.14)	96.49 (3.4)	96.49 (3.4)	96.84 (3.77)	**97.01 (3.95)**	96.13 (3.01)	96.31 (3.2)	96.84 (3.77)
Avg. RI		8.82	6.18	5.99	8.29	9.83	10.26		10.65		

into noisy instances by changing their class labels. This means that we generated class noise. After learning a classifier with the noisy training set, the classification accuracy was measured using the remaining fold, which is a noise-free test set. This process was repeated 10 times per fold to reduce the uncertainty caused by random sampling. We considered the noise rates (p) from 0% to 40% with a 1% increment.

The results are shown in Figure 7. Each graph in the figure represents the change in the average accuracy at varying noise rates. NB was used as a baseline, and the four proposed methods were included in the comparison. As the noise level increases, the predictive accuracy of all classifiers tends to decrease. In general, LNB and GDNB showed the best performance and the least degradation in performance with increasing noise rate, whereas the standard NB performed the worst and showed a rapid decrease in accuracy as more noisy instances were included. Among the proposed methods, ENB appeared to be the most sensitive to noise. In the cases of (b), (c), and (e), although the proposed methods showed similar performances at the zero noise rate, the prediction accuracy of ENB remarkably decreased with a slight increase in noise.

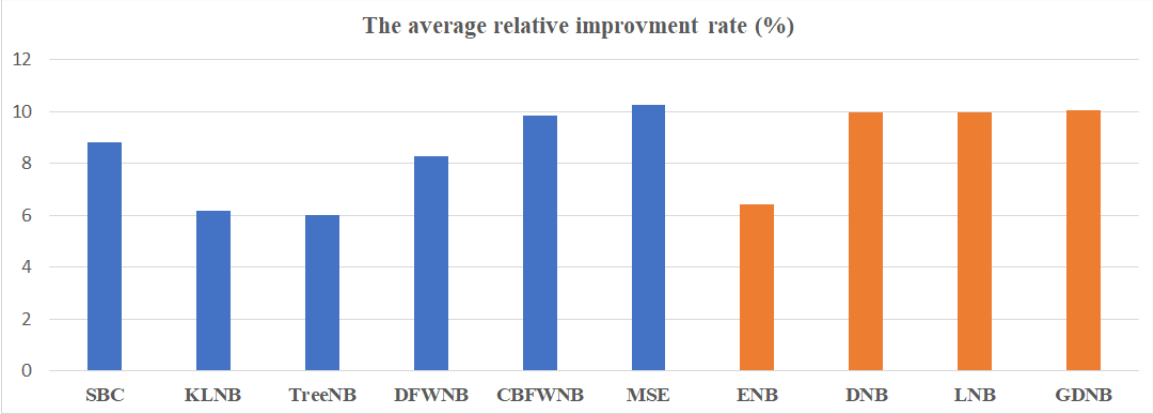


FIGURE 3. Experimental results (the average RI).

TABLE 3. The attribute selection rates (%).

No	ENB	DNB	LNB	GDNB
1	100	100	100	100
2	100	100	100	100
3	43.75	62.5	81.25	81.25
4	60	70	70	70
5	75	75	100	100
6	50	70.83	75	70.83
7	100	83.33	100	100
8	51.85	51.85	51.85	55.56
9	57.14	47.62	52.38	57.14
10	54.69	54.69	54.69	54.69
11	21.28	23.4	23.4	21.28
12	50	43.75	50	43.75
13	75	75	75	75
14	63.16	63.16	63.16	57.89
15	14.29	14.29	14.29	14.29
16	100	100	100	100
17	87.5	75	79.17	79.17
18	87.5	87.5	87.5	87.5
19	83.33	83.33	83.33	83.33
20	81.82	81.82	81.82	81.82
21	77.78	88.89	88.89	88.89
22	62.5	62.5	75	75
23	55.56	55.56	55.56	55.56
24	100	100	100	100
25	63.41	63.41	65.85	56.1
26	55.56	55.56	55.56	55.56
27	80	60	70	70
28	33.33	40	40	43.33

These results support our concept of classifier robustness when introducing DNB, LNB, and GDNB in Section III.

We conducted another simulation study for the robustness to the dimensionality of data. It began with a two-dimensional separable dataset where one attribute (X_{A1}) was generated from $N(0, 2^2)$ and the other attribute (X_{B1}) was generated from $X_{B1}|Y = 1 \sim N(-1.5, 0.5^2)$ and $X_{B1}|Y = -1 \sim N(1.5, 0.5^2)$). It is clear that not X_{A1} but X_{B1} solely contributes to the classification of two classes. In order to increase the dimension of a dataset while preserving the separability, we generated the ith set of attributes from $X_{A,i} \sim N(X_{A,i-1}, 0.1^2)$ and $X_{B,i} \sim N(X_{B,i-1}, 0.1^2)$ and concatenated it to the previously generated sets ($1st$ to $(i-1)th$). Notice that

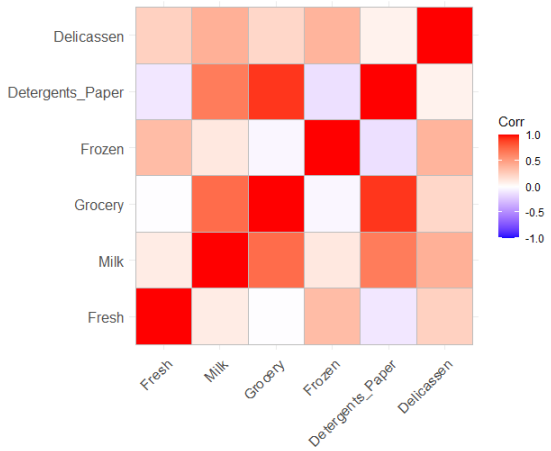


FIGURE 4. Correlation heatmap of "Wholesale customers".

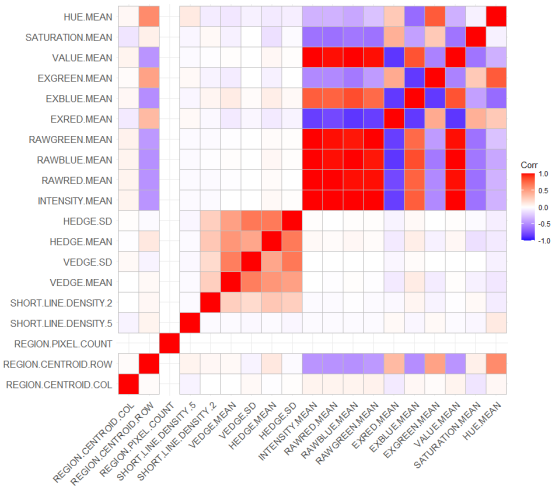


FIGURE 5. Correlation heatmap of "Image segmentation".

the group of 'A' attributes $\{X_{A,i}\}_{i=1}^{d}$ does not contribute to the classification, whereas the group of 'B' attributes $\{X_{B,i}\}_{i=1}^{d}$ evidently does. We examined the dimensions from $d = 2$ to

TABLE 4. Wilcoxon signed-ranks test results (row versus column).

	NB	SBC	KLNB	TreeNB	DFWNB	CBFWNB	MSE	ENB	DNB	LNB	GDNB
ENB	**0.0063**	0.1658	0.2312	0.6115	0.5462	0.3505	*0.0019*	-	*0.0002*	*0.0018*	*0.0008*
DNB	**0.0000**	0.0619	**0.0001**	**0.0003**	0.0795	0.1866	*0.0476*	**0.0002**	-	0.0803	*0.0397*
LNB	**0.0001**	**0.0230**	**0.0000**	**0.0005**	0.0685	0.1271	0.3260	**0.0018**	0.0803	-	0.1773
GDNB	**0.0001**	**0.0107**	**0.0000**	**0.0002**	**0.0451**	0.0877	0.3777	**0.0008**	**0.0397**	0.1773	-

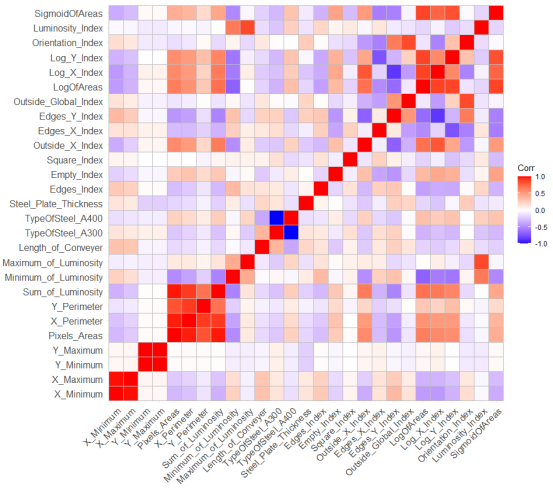


FIGURE 6. Correlation heatmap of "Steel plate faults".

$d = 200$ with the increament of 2. Each training set contained 100 instances and each test set consisted of 1,000 instances, where the class distribution is balanced. After training the classifiers by the proposed ENB, DNB, LNB, and GDNB, we computed the weight ratio by $\sum w(B,i)/(\sum w(A,i) + \sum w(B,i))$ for each classifier. Knowing that $\{X_{A,i}\}_{i=1}^{d}$ is irrelevant but $\{X_{B,i}\}_{i=1}^{d}$ is important to the classification, if the ratio is at least greater than 0.5 or ideally close to 1, we can confirm that the proposed methods are robust to the dimensionality of data. Figure 8 shows the results verifying the robustness. For each of the proposed methods, we reported the test accuracy (black solid line) and the weight ratio (red dashed line). As can be seen from the figure, the test accuracy was very close to 1 and the weight ratio ranged from 0.99 to 1 across all dimensions. It means that the attribute weights of $\{X_{A,i}\}_{i=1}^{d}$ were extremely small and the classification performance was almost perfect while the dimensionality increases.

D. MULTI-CLASS PROBLEMS

We have shown the generalization performance of the proposed method for the binary classification problems. We now attempt to apply the proposed method to several multi-class datasets because many real-world problems require the separation of more than three classes. As mentioned in Section I, our method was designed mainly for binary class cases owing to the underlying principle of the linear formulation for the weighted naive Bayes. Nonetheless, it can be easily generalized to multi-class classifiers using the decomposition strategies [33].

We can decompose a multi-class problem into multiple binary sub-problems. The decomposition strategy has two approaches, the one-against-all (OAA) and the one-against-one (OAO) approaches. The OAA trains k classifiers (k is the number of classes), where one of those discriminates a specific class from the other classes. This approach has the advantage over the OAO that fewer classifiers are trained. However, the class imbalance problem, which causes the accuracy degradation, is inevitable. In the OAO approach, a classifier is trained only for a pair of classes (i and j, $i \neq j$). It therefore decomposes a multi-class problem into ${}_kC_2$ sub-problems. Compared to the OAA, this approach trains more classifiers and the number of classifiers rapidly increases as the number of classes increases. A small size of training set for each classifier is another problem. In this study, we employed the OAO approach because its disadvantages are less likely to appear owing to the simple structure of naive Bayes.

The OAO approach uses the majority voting scheme to classify a test example [34]. The classification scheme with k classes is described below. Let

$$p_{ij}(\boldsymbol{x}) = P_{ij}(y = c_i|\boldsymbol{x}) = \frac{\exp \hat{f}_{ij}(\boldsymbol{x})}{1 + \exp \hat{f}_{ij}(\boldsymbol{x})} \tag{22}$$

where $\hat{f}_{ij}$ is the classifier learned from a dataset containing only the classes c_i and c_j ($1 \leq i, j \leq k$) by the proposed method. $p_{ij}(\boldsymbol{x})$, namely a sub-classifier, returns the probability that a test example $\boldsymbol{x}$ belongs to the class c_i. For a test example $\boldsymbol{x}_l$, the set of sub-classifiers, $\boldsymbol{P}$, is defined by

$$\boldsymbol{P}(\boldsymbol{x}_l) = \begin{bmatrix} 0 & p_{12}(\boldsymbol{x}_l) & \dots & p_{1k}(\boldsymbol{x}_l) \\ p_{21}(\boldsymbol{x}_l) & 0 & \dots & p_{2k}(\boldsymbol{x}_l) \\ \vdots & \vdots & \ddots & \vdots \\ p_{k1}(\boldsymbol{x}_l) & p_{k2}(\boldsymbol{x}_l) & \dots & 0 \end{bmatrix}. \tag{23}$$

Notice that $p_{ij} = 1 - p_{ji}$ and $\boldsymbol{P}$ consists of ${}_kC_2$ sub-classifiers. The diagonal elements of $\boldsymbol{P}$ are all zeros. The classification rule is shown below.

$$\hat{y}_l = c_{\arg\max_i \{p_{i\cdot} | 1 \leq i \leq k\},} \tag{24}$$

where $p_{i\cdot} = \sum_{j=1}^{k} p_{ij}$.

The experimental results are summarized in Table 5 in the same manner of Table 2. The table also shows the number of classes in each dataset. The datasets were chosen from Table 1 with the intention of testing with three classes at least and ten classes at most.

Similar to the binary classification results, all proposed methods outperformed the original naive Bayes. We found

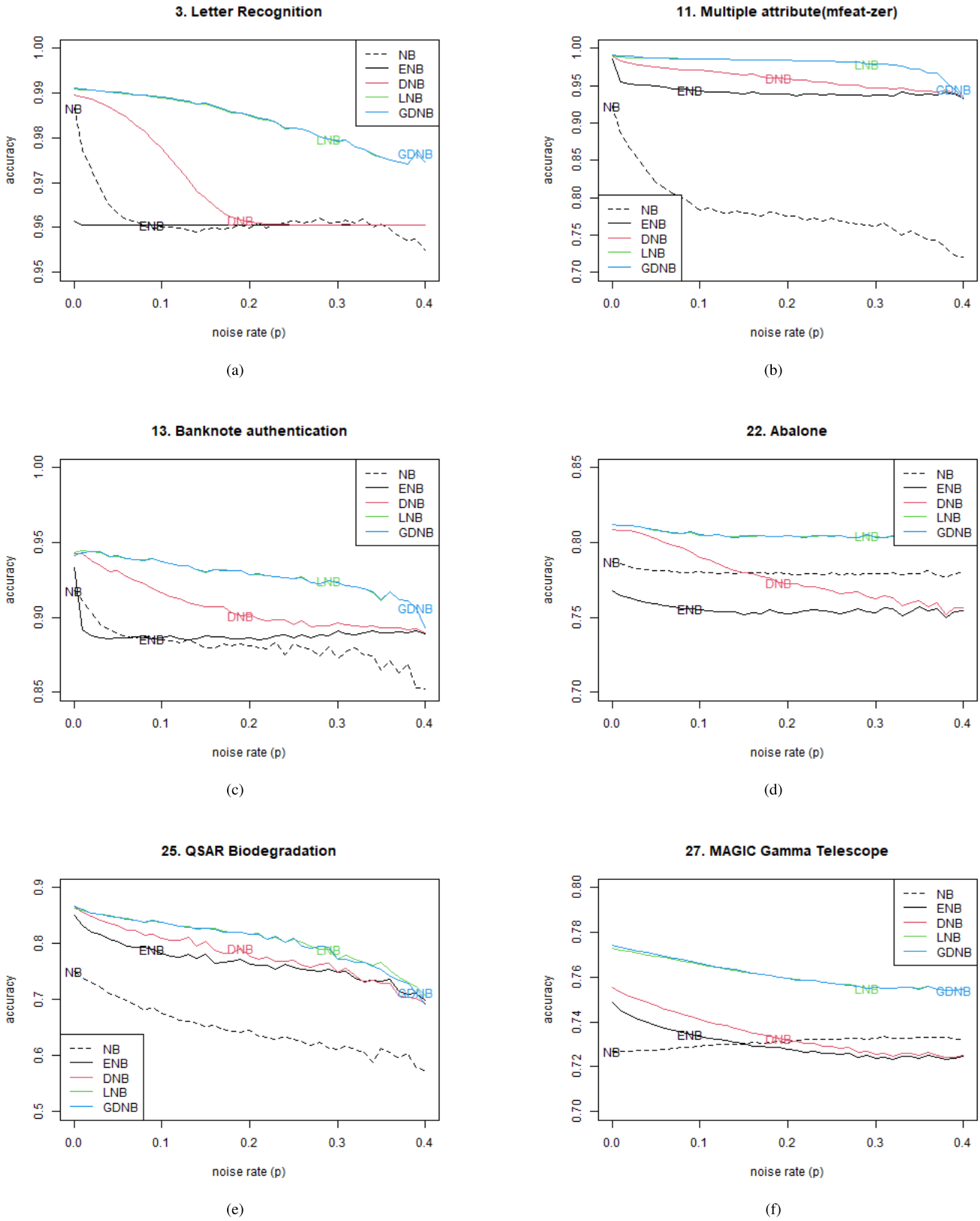


FIGURE 7. Classification performance at varying noise levels.

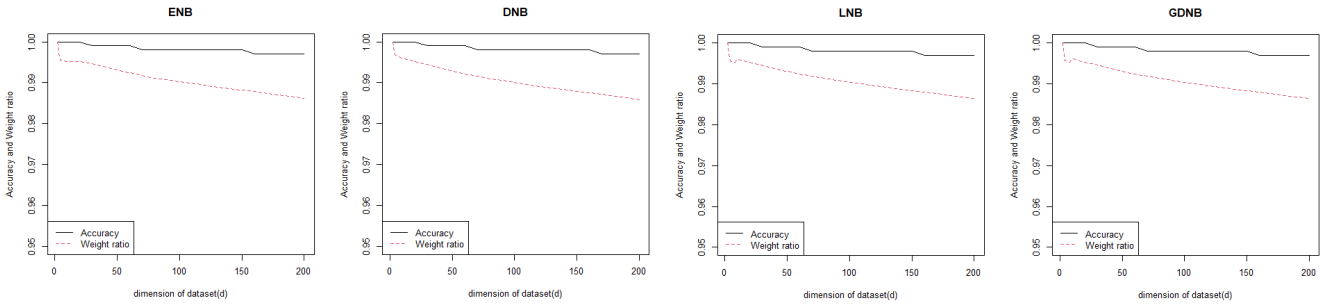


FIGURE 8. Accuracy and weight ratio at increasing dimension.

TABLE 5. Experimental results for multi-class problems (accuracy and relative improvement).

Dataset	#Classes	NB	ENB	DNB	LNB	GDNB
Nursery	4	*90.25 (-)*	92.06 (2)	92.41 (2.39)	92.51 (2.5)	**92.61 (2.62)**
Car Evaluation	4	*86.06 (-)*	88.78 (3.16)	**89.82 (4.37)**	88.6 (2.96)	87.87 (2.1)
Steel Plates Faults	7	*45.74 (-)*	54.76 (19.73)	58.97 (28.92)	54.63 (19.43)	**62.24 (36.07)**
Multiple feature(mfeat-kar)	10	*93.55 (-)*	95.3 (1.87)	**96.1 (2.73)**	94.65 (1.18)	94.65 (1.18)
Image Segmentation	7	*63.81 (-)*	86.19 (35.07)	**88.1 (38.06)**	85.71 (34.33)	85.71 (34.33)
Yeast	10	*47.27 (-)*	49.34 (4.39)	52.39 (10.85)	51.46 (8.86)	**56.19 (18.88)**
Wholesale customers	3	*46.5 (-)*	71.38 (53.53)	**71.61 (54.02)**	71.38 (53.53)	71.38 (53.53)
Avg.RI			17.11	20.19	17.54	21.24

the minimum 17% and the maximum 20% RI improvements by the proposed methods. The RI of DNB was higher than that of LNB, and GDNB was slightly better than DNB. This performance ranking is different from that of the experiments for the binary classification. In the multi-class scenario with the OAO scheme, the GDNB is proven to be a good attribute weighting scheme by finding the optimal attribute weights and the α values simultaneously.

V. CONCLUSION

In this study, new attribute weighting methods for improving naive Bayesian classification have been proposed. The proposed methods consider learning a weighted naive Bayes classifier as a nonlinear optimization problem. Different weights are assigned to attributes by minimizing the proposed loss functions, namely, ENB, DNB, LNB, and GDNB. The validity of each method was confirmed both analytically and empirically. The first method (ENB) is a training-error minimizer. Next, DNB was proposed to extend ENB to an ideal case with an infinite number of training instances. Because DNB is not a maximum likelihood estimator, LNB was proposed by modifying DNB. Finally, GDNB was proposed to automatically find the multiplier of the classification margin according to the noise level of a given training set. The attribute weights determined by the proposed weighted naive Bayes can be seen as a quantitative importance measure of the attributes, that is, the equal attribute importance assumption of the standard NB is relaxed. Because the proposed methods train a classifier and measure its attribute importance simultaneously, one can adjust the complexity of a trained classifier according to the resulting attribute weights. Based on numerical experiments using 28 real-world datasets, we confirmed that the proposed scheme was successful in terms of accuracy and robustness.

This study has a limitation that we considered the binary classification case only due to the linear formulation of the weighted naive Bayes. Although we showed the successful results in Section IV-D by simply generalizing the proposed methods to multi-class problems with the OAO approach, it is still inefficient as ${}_kC_2$ classifiers must be trained. In our ongoing work, we plan to investigate ways to build more elaborate optimization formulations for multi-class weighted naive Bayes.

REFERENCES

[1] P.-N. Tan, M. Steinbach, and V. Kumar, *Introduction to Data Mining*. Reading, MA, USA: Addison-Wesley, 2006.

[2] X. Wu, V. Kumar, J. R. Quinlan, J. Ghosh, Q. Yang, H. Motoda, G. J. McLachlan, A. Ng, B. Liu, and S. Y. Philip, "Top 10 algorithms in data mining," *Knowl. Inf. Syst.*, vol. 14, no. 1, pp. 1–37, Jan. 2008.

[3] Q. Wang, G. M. Garrity, J. M. Tiedje, and J. R. Cole, "Naïve Bayesian classifier for rapid assignment of rRNA sequences into the new bacterial taxonomy," *Appl. Environ. Microbiol.*, vol. 73, no. 16, pp. 5261–5267, 2007.

[4] M. A. Alsheikh, S. Lin, D. Niyato, and H. P. Tan, "Machine learning in wireless sensor networks: Algorithms, strategies, and applications," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 4, pp. 1996–2018, 4th Quart., 2014.

[5] S.-C. Chu, T.-K. Dao, J.-S. Pan, and T.-T. Nguyen, "Identifying correctness data scheme for aggregating data in cluster heads of wireless sensor network based on Naïve Bayes classification," *EURASIP J. Wireless Commun. Netw.*, vol. 2020, no. 1, pp. 1–15, Dec. 2020.

[6] I. Androutsopoulos, J. Koutsias, K. Chandrinos, G. Paliouras, and C. D. Spyropoulos, "An evaluation of Naïve Bayesian anti-spam filtering," 2020, *arXiv:cs/0006013*.

[7] I. Pop, "An approach of the Naïve Bayes classifier for the document classification," *Gen. Math.*, vol. 14, no. 4, pp. 135–138, 2006.

[8] F. Xu, Z. Pan, and R. Xia, "E-commerce product review sentiment classification based on a Naïve Bayes continuous learning framework," *Inf. Process. Manage.*, vol. 57, no. 5, 2020, Art. no. 102221.

[9] H. Zhang, L. Jiang, and L. Yu, "Class-specific attribute value weighting for Naïve Bayes," *Inf. Sci.*, vol. 508, pp. 260–274, Jan. 2020.
[10] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *J. Mach. Learn. Res.*, vol. 3, pp. 1157–1182, May 2003.
[11] C.-H. Lee, F. Gutierrez, and D. Dou, "Calculating feature weights in Naïve Bayes with Kullback–Leibler measure," in *Proc. IEEE 11th Int. Conf. Data Mining*, Dec. 2011, pp. 1146–1151.
[12] M. Hall, "A decision tree-based attribute weighting filter for Naïve Bayes," in *Research and Development in Intelligent Systems XXIII*, M. Bramer, F. Coenen, and A. Tuson, Eds. London, U.K.: Springer, 2007, pp. 59–70.
[13] T. Fawcett, "An introduction to ROC analysis," *Pattern Recognit. Lett.*, vol. 27, no. 8, pp. 861–874, Jun. 2005.
[14] J.-S. Lee and D. Zhu, "When costs are unequal and unknown: A subtree grafting approach for unbalanced data classification*," *Decis. Sci.*, vol. 42, no. 4, pp. 803–829, Nov. 2011.
[15] T. Kim, B. D. Chung, and J.-S. Lee, "Incorporating receiver operating characteristics into Naïve Bayes for unbalanced data classification," *Computing*, vol. 99, no. 3, pp. 203–218, Mar. 2017.
[16] M. A. Hall, "Correlation-based feature selection of discrete and numeric class machine learning," in *Proc. 17th Int. Conf. Mach. Learn.* San Francisco, CA, USA: Morgan Kaufmann, 2000, pp. 359–366.
[17] L. Jiang, C. Li, S. Wang, and L. Zhang, "Deep feature weighting for Naïve Bayes and its application to text classification," *Eng. Appl. Artif. Intell.*, vol. 52, pp. 26–39, Jun. 2016.
[18] L. Jiang, L. Zhang, C. Li, and J. Wu, "A correlation-based feature weighting filter for Naïve Bayes," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 2, pp. 201–213, Feb. 2019.
[19] P. Langley and S. Sage, "Induction of selective Bayesian classifiers," in *Uncertainty Proceedings*, R. L. de Mantaras and D. Poole, Eds. San Francisco, CA, USA: Morgan Kaufmann, 1994, pp. 399–406.
[20] S. Chen, G. I. Webb, L. Liu, and X. Ma, "A novel selective Naïve Bayes algorithm," *Knowl.-Based Syst.*, vol. 192, Mar. 2020, Art. no. 105361.
[21] J. Wu and Z. Cai, "Attribute weighting via differential evolution algorithm for attribute weighted Naïve Bayes (WNB)," *J. Comput. Inf. Syst.*, vol. 7, no. 5, pp. 1672–1679, Mar. 2011.
[22] J. R. Quinlan, *Programs for Machine Learning*. Amsterdam, The Netherlands: Elsevier, 2014, ch. 4.5.
[23] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification Regression Trees*. Evanston, IL, USA: Routledge, 2017.
[24] C. Zhu, R. H. Byrd, P. Lu, and J. Nocedal, "Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization," *ACM Trans. Math. Softw.*, vol. 23, no. 4, pp. 550–560, 1997.
[25] N. A. Zaidi, J. Cerquides, M. J. Carman, and G. I. Webb, "Alleviating Naïve Bayes attribute independence assumption by attribute weighting," *J. Mach. Learn. Res.*, vol. 14, no. 1, pp. 1947–1988, Jul. 2013.
[26] L. Jiang, L. Zhang, L. Yu, and D. Wang, "Class-specific attribute weighted Naïve Bayes," *Pattern Recognit.*, vol. 88, pp. 321–330, Apr. 2019.
[27] S. Taheri, J. Yearwood, M. Mammadov, and S. Seifollahi, "Attribute weighted Naïve Bayes classifier using a local optimization," *Neural Comput. Appl.*, vol. 24, no. 5, pp. 995–1002, Apr. 2014, doi: 10.1007/s00521-012-1329-z.
[28] A. Pérez, P. Larrañaga, and I. Inza, "Bayesian classifiers based on kernel density estimation: Flexible classifiers," *Int. J. Approx. Reasoning*, vol. 50, no. 2, pp. 341–362, Feb. 2009.
[29] J. Nocedal and S. J. Wright, Eds., *Numerical Optimization*. New York, NY, USA: Springer, 1999.
[30] J. Friedman, T. Hastie, and R. Tibshirani, *The Elements of Statistical Learning* (Springer series in Statistics), vol. 1, no. 10. New York, NY, USA: Springer, 2001.
[31] D. Dua and C. Graff. (2017). *UCI Machine Learning Repository*. [Online]. Available: http://archive.ics.uci.edu/ml
[32] J. Demsar, "Statistical comparisons of classifiers over multiple data sets," *J. Mach. Learn. Res.*, vol. 7, pp. 1–30, Dec. 2006.
[33] A. C. Lorena, A. C. P. L. F. de Carvalho, and J. M. P. Gama, "A review on the combination of binary classifiers in multiclass problems," *Artif. Intell. Rev.*, vol. 30, no. 1, p. 19, Aug. 2009.
[34] U.-G. Krebel, "Pairwise classification and support vector machines," in *Advances in Kernel Methods: Support Vector Learning*. Cambridge, MA, USA: MIT Press, 1999, pp. 255–268.

TAEHEUNG KIM received the Ph.D. degree in industrial engineering from Sungkyunkwan University, South Korea. He is currently working as a Research Professor with Sungkyunkwan University. Before rejoining Sungkyunkwan University, he worked with the Production Research Institute, LG Electronics, as a member of the AI Solution Team. His research interest includes the design of learning and optimization algorithms to improve real manufacturing processes.

JONG-SEOK LEE received the Ph.D. degree in industrial engineering from Iowa State University, USA. He worked with the Research and Development Group, SAS Institute, for statistical software development. He is currently an Associate Professor with the Department of Industrial Engineering, Sungkyunkwan University, where he is leading the Info Science Laboratory as a POSCO Fellow Professor. His research papers have been published in the *INFORMS Journal on Computing*, *Information Sciences*, *Decision Sciences*, and other professional journals. His research interests include machine learning, data mining, and their application in the service and manufacturing industries.