MULTIPLE CORRELATION ENCODER-BASED NAIVE BAYES

Anonymous authors
Paper under double-blind review

ABSTRACT

Naive Bayes (NB) continues to be one of the top 10 data mining algorithms. However, due to its assumption of attribute conditional independence, NB encounters significant challenges in addressing attribute-class correlations, attribute-attribute correlations, instance-class correlations, instance-instance correlations, and so on. In the last few decades, a large number of improved algorithms have been proposed, but none of them simultaneously addresses all these correlations. To bridge this gap, this paper proposes a novel algorithm called multiple correlation encoder-based naive Bayes (MCENB). In MCENB, we first design a multiple correlation encoder to generate new attributes, where multiple correlations are simultaneously captured and optimized. Specifically, the newly generated attributes are highly correlated with the class, yet uncorrelated with each other. Instances consisting of new attribute values are highly correlated with those in the same class. Subsequently, we augment original attributes by concatenating them with new attributes. Finally, we weight each augmented attribute to alleviate the attribute redundancy and then build NB on the weighted attributes. The experiments across numerous datasets show that MCENB significantly outperforms its benchmark competitors.

1 INTRODUCTION

Bayesian network classifiers (BNCs) (Pearl, 1989; Friedman et al., 1997; Tang et al., 2016; Zhang et al., 2020) have received much attention in the supervised classification due to their obvious advantages, such as the explicit interpretability and the powerful expression ability. In a supervised classification task containing *m* attribute variables, an instance $\boldsymbol{x}$ can be represented as an attribute value vector $< a_1, a_2, ..., a_j, ..., a_m >$, where a_j is the value of $\boldsymbol{x}$ on the *j*-th attribute A_j. Let *C* represent the class variable and *c* represent the value that *C* takes, BNCs use Eq. (1) to classify $\boldsymbol{x}$:

$$\hat{c}(\boldsymbol{x}) = \underset{c \in C}{arg\,max}\, P(c)P(a_1, a_2, ..., a_j, ..., a_m|c), \tag{1}$$

where $\hat{c}(\boldsymbol{x})$ is the class label of $\boldsymbol{x}$ predicted by BNCs and $P(c)$ is the prior probability of c. Among numerous BNCs, naive Bayes (NB) (Settouti et al., 2016; Domingos & Pazzani, 1996) requires an assumption that all attributes are independent given the class, simplifying the probability estimation of $P(a_1, a_2, ..., a_j, ..., a_m|c)$. According to the assumption, NB uses Eq. (2) to classify $\boldsymbol{x}$:

$$\hat{c}(\boldsymbol{x}) = \underset{c \in C}{arg\,max}\, P(c) \prod_{j=1}^{m} P(a_j|c), \tag{2}$$

where $P(a_j|c)$ is the conditional probability of a_j given c.

Although NB has continued to be one of the top 10 algorithms in data mining (Wu et al., 2008), it is obvious that the assumption of attribute conditional independence is rarely true since various correlations in the real world result in complex dependencies among attributes. To address these correlations, including attribute-class correlations, attribute-attribute correlations, instance-class correlations, instance-instance correlations, and so on, numerous improved algorithms of NB have been proposed. They can be divided into four categories: structure-oriented, probability-oriented, attribute-oriented and instance-oriented algorithms.

Existing improved algorithms have demonstrated significant effectiveness, however, most of them focus on a single correlation, and none of them address all these correlations simultaneously. To

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

bridge this gap, this paper proposes a novel algorithm called multiple correlation encoder-based naive Bayes (MCENB). In MCENB, we first design a multiple correlation encoder to generate new attributes, where multiple correlations are simultaneously captured and optimized. Specifically, the newly generated attributes are highly correlated with the class, yet uncorrelated with each other. Instances consisting of new attribute values are highly correlated with those in the same class. Subsequently, we augment original attributes by concatenating them with new attributes. Finally, we weight each augmented attribute to alleviate the attribute redundancy and then build NB on the weighted attributes. In summary, the main contributions of this paper can be highlighted as:

- We argue that multiple correlations should be simultaneously optimized to improve NB. Attributes should be highly correlated with the class, yet uncorrelated with each other, and instances in the same class should be highly correlated with each other.
- We design a new multiple correlation encoder (MCE), which can capture multiple correlations and then optimize these correlations to generate new attributes exhibiting higher identification abilities compared to original attributes.
- We propose a novel algorithm called multiple correlations encoder-based naive Bayes (MCENB), which uses the original attributes and the new attributes and thus provides a more comprehensive attribute representation for improving NB.

The rest of this paper is organized as: Section 2 conducts a survey on improved algorithms of NB. Section 3 provides a description of our proposed MCENB. Section 4 conducts experiments on real-world and synthetic datasets. Section 5 concludes this paper and discusses the future work.

2 Related work

To address various correlations, numerous improved algorithms of NB have been proposed, which can be divided into four categories: structure-oriented, probability-oriented, attribute-oriented and instance-oriented algorithms. Structure-oriented algorithms (Friedman et al., 1997; Webb et al., 2005; Jiang et al., 2009; Qiu et al., 2015) extend the network structure of NB to capture attribute-attribute correlations. They add directed edges among attribute vertices to represent attribute dependencies and calculate the conditional probabilities of attributes given both the class vertex and their parent vertices. Probability-oriented algorithms (Hindi, 2014; Diab & Hindi, 2017; Hindi et al., 2020; Zhang & Jiang, 2022) suggest that it is rough to represent attribute-class correlations by the conditional probabilities estimated by NB. Therefore, they claim to iteratively fine tune the conditional probabilities to obtain a more effective correlation representation. Attribute-oriented algorithms include three strategies: attribute generation, attribute selection and attribute weighting. Attribute generation (Ou et al., 2022; He et al., 2023) maps original attributes to another attribute space to reduce attribute-attribute correlations. Attribute selection (Chen et al., 2014; 2020) evaluates attribute-class correlations to remove attributes uncorrelated with the class. Attribute weighting (Zaidi et al., 2013; Jiang et al., 2019) evaluates attribute-class correlations and possibly attribute-attribute correlations to assign different weights for different attributes. Similar to attribute-oriented algorithms, instance-oriented algorithms also include three strategies: instance generation, instance selection and instance weighting. Specifically, instance generation (Jiang & Zhang, 2005; Jiang et al., 2008; Li et al., 2025) models instance-instance correlations to generate new instances. Instance selection (Langley & Sage, 2013; Wang et al., 2015) evaluates instance-class correlations to remove redundant and noisy instances in each class. Instance weighting (Jiang et al., 2012; 2014) evaluates instance-class correlations to assign different weights for different instances.

Based on the above discussion, we summarize the correlations addressed in different improved algorithms in Table 1. Existing improved algorithms focus on attribute-class correlations, attribute-attribute correlations, instance-class correlations and instance-instance correlations, but most address one of these correlations. To address the above four correlations simultaneously, this paper proposes a novel algorithm called multiple correlation encoder-based naive Bayes (MCENB).

3 MCENB

In this paper, we first design a new multiple correlation encoder (MCE), which generates new attributes by capturing and optimizing multiple correlations. Based on the MCE, MCENB leverages

Table 1: Comparisons of correlations addressed in different improved algorithms of NB.

Algorithm	attribute-class	attribute-attribute	instance-class	instance-instance
structure-oriented algorithms	-	✓	-	-
probability-oriented algorithms	✓	-	-	-
attribute-oriented algorithms	✓	✓	-	-
instance-oriented algorithms	-	-	✓	✓
our proposed MCENB	✓	✓	✓	✓

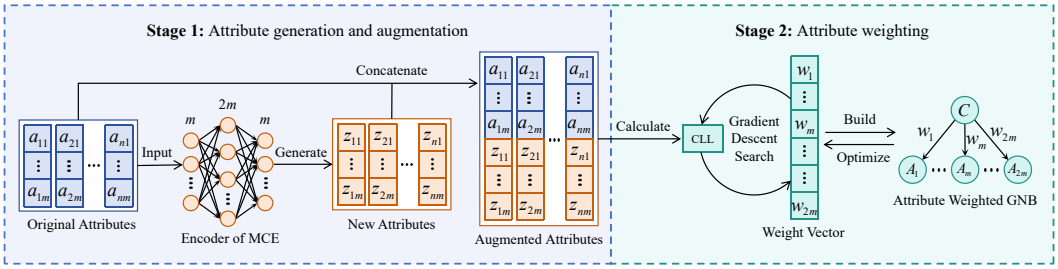


Figure 1: Framework of MCENB.

attribute generation, attribute augmentation and attribute weighting to improve NB. The two-stage framework of MCENB is illustrated in Figure 1. In the first stage, we generate new attributes by inputting original attributes to the encoder of MCE and then augment original attributes by concatenating them with new attributes. In the second stage, we optimize the weight vector by using the gradient descent search to maximize the conditional log-likelihood (CLL) and finally build attribute weighted Gaussian naive Bayes (GNB) on the augmented attributes.

3.1 Attribute Generation and Augmentation

Given a training set with n instances, m attributes and k classes, it can be represented as $\boldsymbol{\mathcal{D}} = \{\mathcal{X}, \boldsymbol{c}\}$, where $\mathcal{X}$ is the set of n original attribute value vectors, $\boldsymbol{c}$ is the class label vector. The i-th attribute value vector in $\mathcal{X}$ is $\boldsymbol{x}_i$, the j-th attribute value of $\boldsymbol{x}_i$ is a_{ij} and the class label of $\boldsymbol{x}_i$ is c_i. For the given $\boldsymbol{\mathcal{D}}$, we design MCE to generate a set of embedding vectors and treat them as new attribute value vectors. The structure of MCE is illustrated in Figure 2. From Figure 2, we can see that MCE consists of an encoder $q(\boldsymbol{z}|\boldsymbol{x})$ and a decoder $p(\boldsymbol{x}|\boldsymbol{z}, \boldsymbol{v})$, where $\boldsymbol{z}$ is the embedding vector generated by $q(\boldsymbol{z}|\boldsymbol{x})$ and $\boldsymbol{v}$ is the one-hot class vector. $q(\boldsymbol{z}|\boldsymbol{x})$ and $p(\boldsymbol{x}|\boldsymbol{z}, \boldsymbol{v})$ are both set to consist of three layers of neurons, with the number of neurons in each layer are m, $2m$, m and m+k, $2m$, m, respectively. During the training process of MCE, original attribute value vectors are processed sequentially. Taking $\boldsymbol{x}_i$ as an example, it is first input to $q(\boldsymbol{z}|\boldsymbol{x})$ to generate an embedding vector $\boldsymbol{z}_i$=$< z_{i1}, z_{i2}, ..., z_{ij}, ..., z_{im} >$, where z_{ij} is the j-th value of $\boldsymbol{z}_i$. Subsequently, to introduce the supervised information into the structure, c_i is transformed into a one-hot class vector $\boldsymbol{v}_i$ and then $\boldsymbol{v}_i$ is concatenated with $\boldsymbol{z}_i$. Finally, the concatenated vector is input to $p(\boldsymbol{x}|\boldsymbol{z}, \boldsymbol{v})$ to generate a reconstructed attribute value vector $\hat{\boldsymbol{x}}_i$, where $\hat{a}_{ij}$ is the j-th reconstructed attribute value of $\hat{\boldsymbol{x}}_i$. Motivated by variational inference (Zhang et al., 2019), MCE is trained by minimizing the negative evidence lower bound (ELBO), which can be formulated as Eq. (3):

$$\mathcal{L}_E = \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x})}[\|\boldsymbol{x} - p(\boldsymbol{x}|\boldsymbol{z}, \boldsymbol{v})\|^2] + KL(\mathcal{Z}||\mathcal{Z}_{prior}), \tag{3}$$

where $\mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x})}[\|\boldsymbol{x} - p(\boldsymbol{x}|\boldsymbol{z}, \boldsymbol{v})\|^2]$ is the expectation of the mean square error between original attribute value vectors and reconstructed attribute value vectors, $KL(\mathcal{Z}||\mathcal{Z}_{prior})$ is the KL divergence between the set of embedding vectors $\mathcal{Z}$ and the variational prior $\mathcal{Z}_{prior}$. We treat the i-th embedding vector $\boldsymbol{z}_i$ in $\mathcal{Z}$ as the i-th new attribute value vector, i.e., the i-th new instance. Meanwhile, we define the j-th new attribute as B_j, and $\mathcal{Z}_{prior}$ requires that each new attribute follows a standard Gaussian distribution given the class.

Based on the structure of MCE, we further improve the identification abilities of new attributes by capturing and optimizing multiple correlations from two perspectives, which are illustrated in Figure 3. From Figure 3, we can see that attribute-class correlations and attribute-attribute correlations

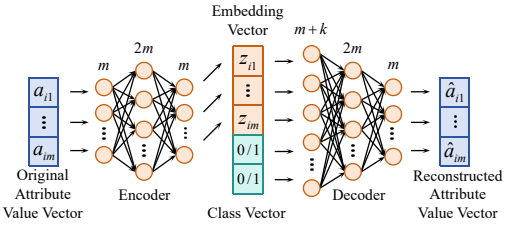


Figure 2: Structure of MCE.

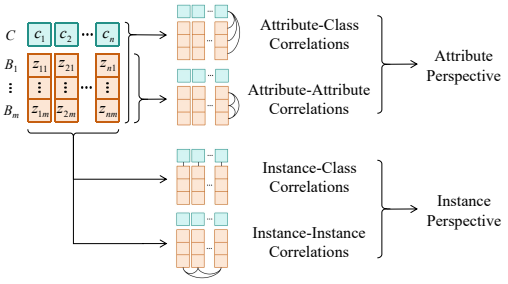


Figure 3: Correlations about new attributes.

can be optimized from the attribute perspective, as well as instance-class correlations and instance-instance correlations from the instance perspective. With these two perspectives, we introduce a new correlation loss $\mathcal{L}_C$, consisting of the attribute correlation loss $\mathcal{L}_A$ and the instance correlation loss $\mathcal{L}_I$. The correlation $\rho(\boldsymbol{\alpha};\boldsymbol{\beta})$ between two vectors $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ is captured by the Pearson coefficient (Cohen & Israel, 2009). The sign of the Pearson coefficient is neglected, as it is uncorrelated with our analysis. Therefore, $\rho(\boldsymbol{\alpha};\boldsymbol{\beta})$ can be formulated as Eq. (4):

$$\rho(\boldsymbol{\alpha};\boldsymbol{\beta}) = \frac{\left|\sum_{l'=1}^{l}\left(\alpha_{l'}-\mu_\alpha\right)\left(\beta_{l'}-\mu_\beta\right)\right|}{\sqrt{\sum_{l'=1}^{l}\left(\alpha_{l'}-\mu_\alpha\right)^2}\sqrt{\sum_{l'=1}^{l}\left(\beta_{l'}-\mu_\beta\right)^2}}, \tag{4}$$

where l is the length of $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, $\alpha_{l'}$ and $\beta_{l'}$ are the l'-th element values of $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, respectively, μ_α and μ_β are the means of element values in $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, respectively.

Attribute perspective. Let $\boldsymbol{b}_j$=$< z_{1j}, z_{2j}, ..., z_{ij}, ..., z_{nj} >$ represent the vector consisting of the attribute values taken by B_j in all new instances, the attribute-class correlation between $\boldsymbol{b}_j$ and $\boldsymbol{c}$ is represented as $\rho(\boldsymbol{b}_j;\boldsymbol{c})$. Similarly, the attribute-attribute correlation between $\boldsymbol{b}_j$ and $\boldsymbol{b}_u$ is represented as $\rho(\boldsymbol{b}_j;\boldsymbol{b}_u)$. To align with the assumption of attribute conditional independence, new attributes should be highly correlated with the class yet uncorrelated with each other. Therefore, we calculate the average attribute-class correlation $\bar{\rho}_{\boldsymbol{bc}}$ and the average attribute-attribute correlation $\bar{\rho}_{\boldsymbol{bb}}$ of new attributes, and then enhance the ratio of $\bar{\rho}_{\boldsymbol{bc}}$ to $\bar{\rho}_{\boldsymbol{bb}}$ by minimizing the attribute correlation loss $\mathcal{L}_A$, which is formulated as Eq. (5):

$$\mathcal{L}_A = -\frac{\bar{\rho}_{\boldsymbol{bc}}}{\bar{\rho}_{\boldsymbol{bb}}} = -\frac{\frac{1}{m}\sum_{j=1}^{m}\rho\left(\boldsymbol{b}_j;\boldsymbol{c}\right)}{\frac{1}{m(m-1)}\sum_{j=1}^{m}\sum_{u=1\wedge u\neq j}^{m}\rho\left(\boldsymbol{b}_j;\boldsymbol{b}_u\right)}. \tag{5}$$

Instance perspective. The instance-instance correlation between $\boldsymbol{z}_i$ and $\boldsymbol{z}_v$ is $\rho(\boldsymbol{z}_i;\boldsymbol{z}_v)$. Since the class label of each instance is a scalar rather than a vector, instance-class correlations can not be explicitly measured by the Pearson coefficient. In this context, we indirectly optimize instance-class correlations through instance-instance correlations. Specifically, new instances in the same class should be highly correlated with each other to tightly aggregate together, thereby enhancing the correlation between each instance and its class. Following the concept of contrastive learning (Khosla et al., 2020), we enhance the ratio between the average instance-instance correlation of instances in each class and the average instance-instance correlation of all instances. This goal is achieved by minimizing the instance correlation loss $\mathcal{L}_I$, which is formulated as Eq. (6):

$$\mathcal{L}_I = -\frac{1}{k}\sum_{c=1}^{k} r_c = -\frac{1}{k}\sum_{c=1}^{k}\frac{\frac{1}{n_c(n_c-1)}\sum_{i=1}^{n_c}\sum_{v=1\wedge v\neq i}^{n_c}\rho(\boldsymbol{z}_i;\boldsymbol{z}_v)}{\frac{1}{n(n-1)}\sum_{i=1}^{n}\sum_{v=1\wedge v\neq i}^{n}\rho(\boldsymbol{z}_i;\boldsymbol{z}_v)}, \tag{6}$$

where r_c is the instance-instance correlation ratio corresponding to the c-th class and n_c is the number of instances in the c-th class.

Considering the negative ELBO $\mathcal{L}_E$ and the correlation loss $\mathcal{L}_C$, we finally minimize the total loss $\mathcal{L}$ to train MCE, which can be formulated as Eq. (7):

$$\mathcal{L} = \mathcal{L}_E + \mathcal{L}_C, \tag{7}$$

where $\mathcal{L}_C = \mathcal{L}_A + \mathcal{L}_I$. We update the parameters $\mathcal{W}$ of MCE over P iterations by Eq. (8):

$$\mathcal{W}_{p+1} = \mathcal{W}_p - \eta \frac{\partial \mathcal{L}}{\partial \mathcal{W}}, \tag{8}$$

where $\mathcal{W}_p$ represents the parameters in the p-th iteration and η represents the learning rate. When p = 1, $\mathcal{W}_p$ represents randomly initialized parameters. And the training process terminates when $\mathcal{L}$ shows no improvement for 10 consecutive epochs.

Finally, we use the trained MCE to generate new attributes. Since MCE optimizes multiple correlations from the attribute and the instance perspectives simultaneously, new attributes exhibit higher identification abilities compared to original attributes. To enhance the identification abilities of original attributes and provide a more comprehensive attribute representation, we augment original attributes by concatenating them with new attributes. For each instance, we define z_{ij} as the $(m+j)$-th attribute value of $\boldsymbol{x}_i$. The length of $\boldsymbol{z}_i$ is the same as that of $\boldsymbol{x}_i$, it is evident that each instance can obtain m new attributes, with the final length expanding to $2m$.

3.2 Attribute weighting

Since new attributes are generated from original attributes, attribute augmentation may potentially cause the attribute redundancy. In this subsection, to alleviate the impact of attribute redundancy, we assign different weights for different augmented attributes. Besides, to align with the numerical attributes generated by MCE, we employ GNB to predict the class label, which is a variant of NB specifically adapted to numerical attributes. Based on the weighted attributes, we build attribute weighted GNB and use Eq. (9) to predict the class label:

$$\hat{c}(\boldsymbol{x}) = \underset{c \in C}{arg\,max}\, P(c) \prod_{j=1}^{2m} P(a_j|c)^{w_j}, \tag{9}$$

where w_j is the weight of A_j. $P(c)$ and $P(a_j|c)$ are estimated by Eq. (10) and Eq. (11):

$$P(c) = \frac{\sum_{i=1}^{n} I(c_i, c) + 1}{n + k}, \tag{10}$$

$$P(a_j|c) = \frac{1}{\sqrt{2\pi}\sigma_{cj}} \exp\left(-\frac{(a_j - \mu_{cj})^2}{2\sigma_{cj}^2}\right), \tag{11}$$

where $I(\cdot)$ represents a binary function, which takes the value 1 if its two parameters are identical and 0 otherwise. And μ_{cj} and σ_{cj} represent the mean and standard deviation of A_j given c, respectively.

After building the attribute weighted GNB, we assign different weights for different augmented attributes. Initially, each weight in the weight vector $\boldsymbol{w}$ is assigned a random value between 0 and 1. Then, these weights are optimized by the gradient descent search. The objective function of optimization is defined to maximize the CLL of the attribute weighted GNB, which can be formulated as Eq. (12):

$$\text{CLL}(\boldsymbol{w}) = \log \hat{P}(C|\boldsymbol{\mathcal{D}}, \boldsymbol{w}) = \sum_{i=1}^{n} \log \hat{P}(c_i|\boldsymbol{x}_i, \boldsymbol{w}), \tag{12}$$

where $\hat{P}(c_i|\boldsymbol{x}_i, \boldsymbol{w})$ is the posterior probability of c_i estimated by the attribute weighted GNB given $\boldsymbol{x}_i$ and $\boldsymbol{w}$, which is formulated as Eq. (13):

$$\hat{P}(c_i|\boldsymbol{x}_i, \boldsymbol{w}) = \frac{\varphi(c_i|\boldsymbol{x}_i, \boldsymbol{w})}{\sum_{c=1}^{k} \varphi(c|\boldsymbol{x}_i, \boldsymbol{w})}, \tag{13}$$

where $\varphi(c|\boldsymbol{x}_i, \boldsymbol{w})$ is the product of $P(c)$ and each $P(a_j|c)^{w_j}$ of $\boldsymbol{x}_i$, which is formulated as Eq. (14):

$$\varphi(c|\boldsymbol{x}_i, \boldsymbol{w}) = P(c) \prod_{j=1}^{2m} P(a_j|c)^{w_j}. \tag{14}$$

The gradient of $\varphi(c|\boldsymbol{x}_i, \boldsymbol{w})$ with respect to w_j can be formulated as Eq. (15):

$$\begin{aligned}\frac{\partial}{\partial w_j}\varphi(c|\boldsymbol{x}_i, \boldsymbol{w}) &= \left(P(c)\prod_{j'=1\wedge j'\neq j}^{2m} P(a_{j'}|c)^{w_{j'}}\right)\frac{\partial}{\partial w_j}P(a_j|c)^{w_j} \\ &= \left(P(c)\prod_{j'=1\wedge j'\neq j}^{2m} P(a_{j'}|c)^{w_{j'}}\right)P(a_j|c)^{w_j}\log P(a_j|c) \\ &= \varphi(c|\boldsymbol{x}_i, \boldsymbol{w})\log P(a_j|c).\end{aligned} \tag{15}$$

Then, the gradient of CLL($\boldsymbol{w}$) with respect to w_j can be formulated as Eq. (16):

$$\begin{aligned}\frac{\partial}{\partial w_j}\text{CLL}(\boldsymbol{w}) &= \frac{\partial}{\partial w_j}\sum_{i=1}^{n}\left(\log\varphi(c_i|\boldsymbol{x}_i, \boldsymbol{w}) - \log\left(\sum_{c=1}^{k}\varphi(c|\boldsymbol{x}_i, \boldsymbol{w})\right)\right) \\ &= \sum_{i=1}^{n}\left(\log P(a_j|c_i) - \frac{\sum_{c=1}^{k}\varphi(c|\boldsymbol{x}_i, \boldsymbol{w})\log P(a_j|c)}{\sum_{c=1}^{k}\varphi(c|\boldsymbol{x}_i, \boldsymbol{w})}\right) \\ &= \sum_{i=1}^{n}\left(\log P(a_j|c_i) - \sum_{c=1}^{k}\hat{P}(c|\boldsymbol{x}_i, \boldsymbol{w})\log P(a_j|c)\right).\end{aligned} \tag{16}$$

In summary, the entire learning algorithm for our MCENB can be partitioned into training (MCENB - training) and classification (MCENB - classification) algorithms. These two algorithms and their time complexity analyses are provided in **Appendix A** and **Appendix B**, respectively.

4 Experiments and Results

The purpose of this section is to validate the effectiveness of MCENB. We first compare the classification accuracy of MCENB with its seven competitors and conduct two groups of ablation studies on 24 real-world datasets. Then, we observe the identification abilities of new attributes, the optimization of correlations and the rationality of MCENB on a synthetic dataset.

4.1 Experiments on 24 real-world datasets

From the real-world datasets published by the KEEL dataset repository[1], we use the whole 24 datasets only containing numerical attributes, which represent a wide range of domains and data characteristics. The descriptions of datasets are provided in **Appendix C**. In our experiments, attributes are normalized by the z-score method (Patro & Sahu, 2015). We compare the classification accuracy of MCENB with its seven competitors by running 10 separate hold-out validations, in which each dataset is split into a training set (80%) and a test set (20%) by stratified sampling.

Classification performance. To verify the classification performance of MCENB, we compare it with GNB and its six state-of-the-art competitors, including attribute grouping-based naive Bayesian classifier (AG-NBC) (He et al., 2023), auto-encoding naive Bayesian classifier (AE-NBC) (Ou et al., 2022), weighting attributes to alleviate naive Bayes' independence assumption (WANBIA) (Zaidi et al., 2013), correlation-based feature weighting filter for naive Bayes (CFWNB) (Jiang et al., 2019), instance correlation graph-based naive Bayes (ICGNB) (Li et al., 2025) and discriminatively weighted naive Bayes (DWNB) (Jiang et al., 2012). Among all competitors, GNB is the baseline. AG-NBC and AE-NBC address attribute-attribute correlations through attribute generation. WANBIA addresses attribute-class correlations through attribute weighting. CFWNB addresses attribute-class correlations and attribute-attribute correlations through attribute weighting. ICGNB addresses instance-instance correlations through instance generation. DWNB addresses instance-class correlations through instance weighting. For algorithms focusing on nominal attributes (WANBIA,

[1] https://sci2s.ugr.es/keel/category.php?cat=clas

Table 2: Classification accuracy (%) comparisons for MCENB versus its competitors.

Dataset	MCENB	AG-NBC	AE-NBC	WANBIA	CFWNB	ICGNB	DWNB	GNB
appendicitis	89.55±7.34	83.64±6.80	83.18±10.77	88.18±7.66	**90.00±6.03**	89.09±7.66	**90.00±6.03**	86.36±6.10
balance	88.88±2.49	**91.68±1.53**	79.84±5.22	90.24±2.84	88.32±3.87	88.00±3.12	91.36±2.23	90.24±2.84
banana	64.44±2.11	**84.80±3.83**	62.41±3.19	62.00±0.91	59.74±1.48	69.53±0.80	61.46±0.96	62.00±0.91
cleveland	**58.17±5.60**	53.67±3.86	55.50±7.82	58.00±4.27	54.83±4.86	57.67±4.84	55.17±7.87	51.67±10.22
ecoli	78.82±6.14	70.00±6.35	78.53±2.81	79.12±6.06	60.29±11.60	**79.12±5.46**	62.50±6.65	60.74±6.61
glass	62.79±7.28	60.70±6.61	60.47±8.06	59.07±7.44	51.40±11.74	**63.02±8.85**	47.44±9.54	47.21±9.70
iris	**96.33±3.14**	90.33±6.90	91.00±7.31	96.00±3.27	95.33±3.71	**96.33±3.14**	**96.33±3.14**	95.33±3.71
led7digit-01	71.80±2.96	71.30±4.73	71.70±3.26	70.40±5.90	64.20±9.11	**72.40±4.27**	61.00±14.66	63.30±12.12
magic	78.85±0.61	75.04±1.33	77.69±1.35	77.05±0.67	74.56±0.56	78.38±0.78	**79.46±0.52**	72.56±0.64
movement_libras	**78.19±5.34**	69.17±6.77	70.97±6.38	62.92±4.97	62.22±4.76	56.11±4.31	62.50±5.27	61.94±5.63
phoneme	76.18±1.42	77.22±1.69	76.91±1.62	75.91±1.40	76.85±1.58	76.46±1.10	**77.74±1.45**	75.97±1.65
pima	**76.43±2.55**	73.12±2.23	73.70±2.94	75.52±2.69	75.06±2.99	75.32±2.29	75.91±2.94	74.61±3.45
ring	97.88±0.26	93.34±1.18	94.73±0.51	97.90±0.20	97.96±0.30	97.98±0.36	**97.89±0.28**	97.92±0.28
segment	**91.00±1.39**	88.01±1.76	83.35±2.60	88.81±1.26	80.52±1.26	90.41±1.59	85.87±1.12	79.42±1.48
sonar	**81.67±5.43**	76.67±7.81	67.15±8.05	78.33±5.05	67.86±5.46	78.57±5.73	78.33±4.70	66.67±5.11
spambase	**93.24±0.55**	86.52±1.83	86.51±0.98	89.99±1.08	83.71±1.26	90.67±1.08	83.32±2.45	82.08±1.25
texture	**98.74±0.34**	94.91±0.78	94.22±0.74	84.47±1.00	78.35±1.38	96.84±0.52	78.38±1.21	77.45±1.39
titanic	77.44±1.23	75.51±1.70	76.98±0.89	77.64±1.21	76.98±0.89	77.41±1.19	**78.07±1.32**	76.98±0.89
twonorm	97.74±0.35	96.37±0.61	95.31±0.91	97.72±0.27	97.71±0.29	97.66±0.23	**97.76±0.32**	97.70±0.28
wdbc	**97.37±1.11**	94.65±1.69	85.53±2.97	96.40±1.54	93.95±1.90	96.32±1.51	95.79±1.56	92.98±2.29
wine	96.67±2.42	97.50±1.94	92.22±5.53	**97.50±1.50**	96.94±1.94	96.94±1.94	96.67±2.42	**97.50±1.50**
winequality-red	**59.56±2.59**	58.84±3.19	57.16±2.93	58.44±1.78	58.47±1.49	59.53±2.56	55.50±2.47	54.72±2.56
winequality-white	**53.29±1.00**	51.02±1.94	51.51±1.43	52.21±1.51	49.23±1.14	52.64±1.27	47.69±1.58	44.38±1.61
yeast	**56.77±2.79**	50.03±3.07	55.49±1.87	54.28±3.20	18.22±3.83	56.53±3.26	15.42±2.42	14.41±3.38
(**W / T / L**)		20/0/4	23/0/1	19/0/5	20/0/4	16/1/7	15/2/7	21/0/3
Average	80.08	77.67	75.92	77.84	73.03	78.87	73.82	71.84

Table 3: Wilcoxon tests for MCENB versus its competitors.

Algorithm	MCENB	AG-NBC	AE-NBC	WANBIA	CFWNB	ICGNB	DWNB	GNB
MCENB	-	○	○	○	○	○	○	○
AG-NBC	●	-			○			○
AE-NBC	●		-	●		●		○
WANBIA	●		○	-	○	●	○	○
CFWNB	●	●		●	-	●		○
ICGNB	●	○	○	○	○	-	○	○
DWNB	●			●		●	-	○
GNB	●	●		●	●	●	●	-

CFWNB and DWNB), we replace NB in them with GNB. In MCENB, P is set to 500 and η is set to 0.01. The parameters of competitors are consistent with those in the original papers. Table 2 shows the classification accuracy (%) and standard deviation, in which the highest classification accuracy on each dataset is bolded. The Win / Tie / Lose (**W / T / L**) values and the averages (arithmetic mean) are summarized at the bottom of the table. The Win / Tie / Lose values imply that MCENB wins on **W** datasets, ties on **T** datasets, and loses on **L** datasets compared to the competitor. Table 3 shows the results of the Wilcoxon signed-ranks test (Demsar, 2006) of each pair of algorithms. In Table 3, symbol ● indicates that the algorithm in the column outperforms the one in the corresponding row, and symbol ○ signifies the opposite. The lower-diagonal significance level is $\alpha = 0.05$, while the upper-diagonal level is $\alpha = 0.1$. Observing these results, the conclusion is evident that MCENB significantly outperforms all the other competitors. Specific conclusions are summarized as:

(1) Compared to AG-NBC, AE-NBC, WANBIA and CFWNB, MCENB is better on 20, 23, 19, 20 datasets and only worse on 4, 1, 5, 4 datasets, respectively. MCENB wins the competitors addressing the correlations from the attribute perspective. This verifies the effectiveness of optimizing the correlations from the instance perspective.

(2) Compared to ICGNB and DWNB, MCENB is better on 16, 15 datasets and worse on 7, 7 datasets. MCENB wins the competitors addressing the correlations from the instance perspective. This verifies the effectiveness of optimizing the correlations from the attribute perspective.

(3) Compared to GNB, MCENB is better on 21 datasets and worse on 3 datasets. This indicates that MCENB effectively enhances the classification performance of GNB, and MCE is powerful.

(4) The average classification accuracy of MCENB (80.08%) is higher than those of AG-NBC (77.67%), AE-NBC (75.92%), WANBIA (77.84%), CFWNB (73.03%), ICGNB (78.87%), DWNB (73.82%) and GNB (71.84%), respectively. This suggests that MCENB performs better than existing state-of-the-art competitors.

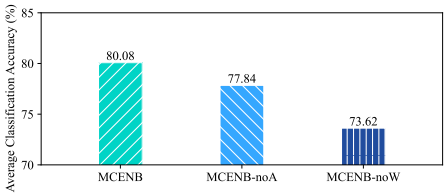


Figure 4: Average classification accuracy (%) of the first group of ablation studies.

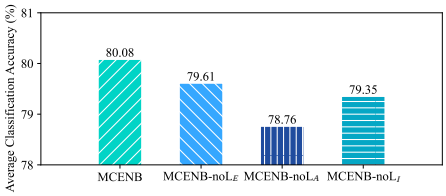


Figure 5: Average classification accuracy (%) of the second group of ablation studies.

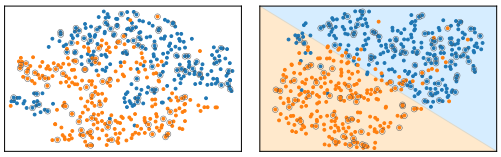
(5) According to the Wilcoxon test results in Table 3, MCENB significantly outperforms all existing state-of-the-art competitors whether α = 0.05 or α = 0.1. This strongly validates the classification performance of MCENB.

Ablation studies. To validate the necessity of each stage in MCENB and each component in the total loss, we perform two groups of ablation studies by comparing MCENB with its ablation variants. The first group of variants are MCENB-noA and MCENB-noW. MCENB-noA removes the stage of attribute generation and augmentation while retaining attribute weighting. MCENB-noW removes the stage of attribute weighting while retaining attribute generation and augmentation. The second group of variants are MCENB-noL$_E$, MCENB-noL$_A$ and MCENB-noL$_I$. MCENB-noL$_E$ removes the negative ELBO $\mathcal{L}_E$ while retaining $\mathcal{L}_A$ and $\mathcal{L}_I$. MCENB-noL$_A$ removes the attribute correlation loss $\mathcal{L}_A$ while retaining $\mathcal{L}_E$ and $\mathcal{L}_I$, and MCENB-noL$_I$ removes the instance correlation loss $\mathcal{L}_I$ while retaining $\mathcal{L}_E$ and $\mathcal{L}_A$. Figures 4 and 5 show the results. Observing the results in Figure 4, we summarize the conclusions as: (1) The accuracy of MCENB is higher than that of MCENB-noA, which confirms that enhancing the identification abilities of the original attributes through attribute generation and augmentation is necessary. (2) The accuracy of MCENB is higher than that of MCENB-noW, which confirms that alleviating the attribute redundancy through attribute weighting is necessary. Observing the results in Figure 5, we summarize the conclusions as: (1) The accuracy of MCENB is higher than that of MCENB-noL$_E$, which confirms that minimizing the negative ELBO through MCE is necessary. (2) The accuracy of MCENB is higher than that of MCENB-noL$_A$, which confirms that optimizing the correlations from the attribute perspective is necessary. (3) The accuracy of MCENB is higher than that of MCENB-noL$_I$, which confirms that optimizing the correlations from the instance perspective is necessary.

4.2 Experiments on a synthetic dataset

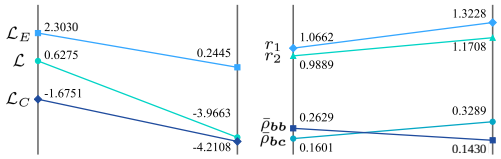
In this subsection, we design a group of experiments on a synthetic dataset. The synthetic dataset contains 2 classes, 500 instances and 20 attributes. Its synthetic process initially creates Gaussian clusters around the vertices of a 10-dimensional hypercube and assigns an equal number of clusters to each class. By sampling from these clusters, instances containing 10 informative attributes can be obtained. Then, 10 redundant attributes are generated by random linear combinations of these informative attributes. On the synthetic dataset, we observe the identification abilities of new attributes through a hold-out validation, the optimization of correlations and the rationality of MCENB through 10 separate hold-out validations.

Identification abilities. To observe the identification abilities of new attributes generated by MCE, we compare the class distributions of original attributes and augmented attributes. We employ the t-SNE algorithm (van der Maaten & Hinton, 2008) to map the dataset to a two-dimensional space. Figure 6 shows the class distributions of original attributes and augmented attributes, where scatters with a circle are test instances and the rest are training instances. In Figure 6a, we can see that the class distribution of original attributes is confusing, and there is no class boundary between the two classes. In Figure 6b, we can see that the class distribution of augmented attributes is significantly more distinguishable compared to that of original attributes. Besides, most test instances are aggregated together with the training instances in the same class, while only a few test instances are around the class boundary. This facilitates predicting the class of each instance accurately. In summary, the class distributions in Figure 6 indicate that augmented attributes exhibit higher identification abilities compared to original attributes.



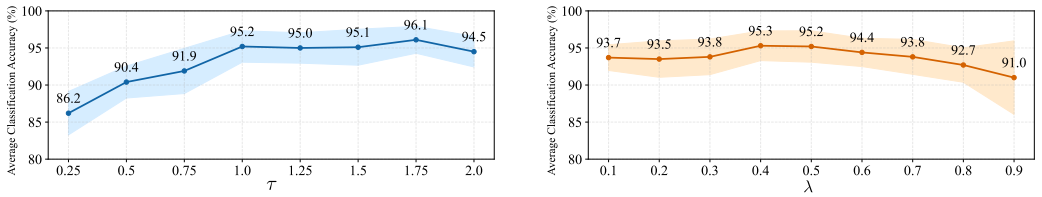
(a) original attributes (b) augmented attributes

Figure 6: Class distributions of original attributes and augmented attributes.



(a) loss (b) correlation indicator

Figure 7: Value Changes of losses and correlation indicators.



(a) ratio of new attributes to original attributes (b) weight of $\mathcal{L}_E$

Figure 8: Average classification accuracy (%) of MCENB with different parameters.

Correlation optimization. To observe whether multiple correlations are effectively optimized through MCE, we show the value changes of losses and correlation indicators in Figure 7. In Figures 7a and 7b, initial values are on the left line and final values are on the right line. From the loss perspective, the final values of $\mathcal{L}$, $\mathcal{L}_E$ and $\mathcal{L}_C$ are smaller than their initial values. This suggests that $\mathcal{L}$ and all its components are effectively optimized. From the attribute perspective, the final value of $\bar{\rho}_{\boldsymbol{bc}}$ is greater than its initial value, while that of $\bar{\rho}_{\boldsymbol{bb}}$ shows the opposite trend. The final value of $\bar{\rho}_{\boldsymbol{bc}}$ is greater than that of $\bar{\rho}_{\boldsymbol{bb}}$. This suggests that new attributes exhibit higher attribute-class correlations and lower attribute-attribute correlations compared to original attributes. From the instance perspective, r_1 and r_2 are the instance-instance correlation ratios corresponding to two classes, respectively. The final values of them are greater than the initial values. This suggests that new attributes exhibit higher instance-instance correlations in the same class compared to original attributes.

Rationality discussion. In MCENB, the number of new attributes is set to be equal to that of original attributes and components of the total loss are summed directly in Eq. (7). To discuss the rationality of the setup, we conduct two experiments by introducing two new parameters. In the first experiment, we introduce the ratio τ of new attributes to original attributes and vary τ from 0.25 to 2.0. In the second experiment, we introduce the weight λ of $\mathcal{L}_E$ and redefine the total loss as $\mathcal{L} = \lambda\mathcal{L}_E + (1-\lambda)\mathcal{L}_C$, then we vary λ from 0.1 to 0.9. Figures 8a and 8b show the results, where the point represents the average classification accuracy (%) and the shaded region represents the standard deviation. Figure 8a shows that the classification performance of MCENB improves as τ increases and stabilizes while $\tau \geqslant 1.0$. Figure 8b shows that MCENB achieves optimal classification performance while $\lambda \in [0.4, 0.5]$. These results indicate the rationality of MCENB (τ = 1.0 and λ = 0.5), ensuring both the simplicity of formulas and the classification performance.

5 Conclusion and Future Work

To obtain highly identifiable attributes, we design a new multiple correlation encoder (MCE) to generate new attributes by capturing and optimizing multiple correlations. Based on it, we propose a novel algorithm called multiple correlation encoder-based naive Bayes (MCENB). Comprehensive experiments demonstrate its effectiveness on both real-world and synthetic datasets. However, MCENB only leverages the absolute value not the sign of the Pearson coefficient, failing to consider the negative correlation. Exploring how to leverage the negative correlation is a promising direction for future work. And there is no negative societal impact of our work.

AI USE STATEMENT

In this work, we used generative AI tools only for minor text editing and formatting assistance. We have not used generative AI tools for research ideation, method development, experiment implementation, data analysis, or result interpretation. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We submit the code and datasets as supplementary materials, and the details of dataset preprocessing and algorithm implementation are provided in the main text. Once our paper is accepted, we will make the code and datasets publicly available on GitHub.

REFERENCES

Shenglei Chen, Ana M. Martínez, and Geoffrey I. Webb. Highly scalable attribute selection for averaged one-dependence estimators. In *Advances in Knowledge Discovery and Data Mining - 18th Pacific-Asia Conference, PAKDD 2014, Tainan, Taiwan, May 13-16, 2014. Proceedings, Part II*, volume 8444 of *Lecture Notes in Computer Science*, pp. 86–97. Springer, 2014.

Shenglei Chen, Geoffrey I. Webb, Linyuan Liu, and Xin Ma. A novel selective naïve bayes algorithm. *Knowl. Based Syst.*, 192:105361, 2020.

Cohen and Israel. Pearson correlation coefficient. *Noise reduction in speech processing.*, (1-4), 2009.

Janez Demsar. Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.*, 7:1–30, 2006.

Diab M. Diab and Khalil M. El Hindi. Using differential evolution for fine tuning naïve bayesian classifiers and its application for text classification. *Appl. Soft Comput.*, 54:183–199, 2017.

Pedro M. Domingos and Michael J. Pazzani. Beyond independence: Conditions for the optimality of the simple bayesian classifier. In *Proceedings of the Thirteenth International Conference on Machine Learning (ICML 1996), Bari, Italy, July 3-6, 1996*, pp. 105–112. Morgan Kaufmann, 1996.

Nir Friedman, Dan Geiger, and Moisés Goldszmidt. Bayesian network classifiers. *Mach. Learn.*, 29 (2-3):131–163, 1997.

Yulin He, Guiliang Ou, Philippe Fournier-Viger, and Joshua Zhexue Huang. Attribute grouping-based naive bayesian classifier. *Sci. China Inf. Sci.*, 2023.

Khalil M. El Hindi. Fine tuning the naïve bayesian learning algorithm. *AI Commun.*, 27(2):133–141, 2014.

Khalil M. El Hindi, Reem R. Aljulaidan, and Hussien AlSalman. Lazy fine-tuning algorithms for naïve bayesian text classification. *Appl. Soft Comput.*, 96:106652, 2020.

Liangxiao Jiang and Harry Zhang. Learning instance greedily cloning naive bayes for ranking. In *Proceedings of the 5th IEEE International Conference on Data Mining (ICDM 2005), 27-30 November 2005, Houston, Texas, USA*, pp. 202–209. IEEE Computer Society, 2005.

Liangxiao Jiang, Dianhong Wang, Harry Zhang, Zhihua Cai, and Bo Huang. Using instance cloning to improve naive bayes for ranking. *Int. J. Pattern Recognit. Artif. Intell.*, 22(6):1121–1140, 2008.

Liangxiao Jiang, Harry Zhang, and Zhihua Cai. A novel bayes model: Hidden naive bayes. *IEEE Trans. Knowl. Data Eng.*, 21(10):1361–1371, 2009.

Liangxiao Jiang, Dianhong Wang, and Zhihua Cai. Discriminatively weighted naive bayes and its application in text classification. *Int. J. Artif. Intell. Tools*, 21(1), 2012.

Liangxiao Jiang, Chaoqun Li, and Shasha Wang. Cost-sensitive bayesian network classifiers. *Pattern Recognit. Lett.*, 45:211–216, 2014.

Liangxiao Jiang, Lungan Zhang, Chaoqun Li, and Jia Wu. A correlation-based feature weighting filter for naive bayes. *IEEE Trans. Knowl. Data Eng.*, 31(2):201–213, 2019.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

Pat Langley and Stephanie Sage. Induction of selective bayesian classifiers. *CoRR*, abs/1302.6828, 2013.

Chengyuan Li, Liangxiao Jiang, Wenjun Zhang, Liangjun Yu, and Huan Zhang. Instance correlation graph-based naive bayes. In *Proceedings of the Seventeenth International Conference on Machine Learning (ICML 2025), Vancouver, Canada, Jul 13-19, 2025*. Morgan Kaufmann, 2025.

Guiliang Ou, Yulin He, Philippe Fournier-Viger, and Joshua Zhexue Huang. Auto-encoding independent attribute transformation for naive bayesian classifier. In *International Joint Conference on Neural Networks, IJCNN 2022, Padua, Italy, July 18-23, 2022*, pp. 1–8. IEEE, 2022.

S. Gopal Krishna Patro and Kishore Kumar Sahu. Normalization: A preprocessing stage. *CoRR*, abs/1503.06462, 2015.

Judea Pearl. *Probabilistic reasoning in intelligent systems - networks of plausible inference*. Morgan Kaufmann series in representation and reasoning. Morgan Kaufmann, 1989.

Chen Qiu, Liangxiao Jiang, and Chaoqun Li. Not always simple classification: Learning superparent for class probability estimation. *Expert Syst. Appl.*, 42(13):5433–5440, 2015.

Nesma Settouti, Mohammed El Amine Bechar, and Mohammed Amine Chikh. Statistical comparisons of the top 10 algorithms in data mining for classi cation task. *Int. J. Interact. Multim. Artif. Intell.*, 4(1):46–51, 2016.

Bo Tang, Steven Kay, and Haibo He. Toward optimal feature selection in naive bayes for text categorization. *IEEE Trans. Knowl. Data Eng.*, 28(9):2508–2521, 2016.

Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9:11, 2008.

Shasha Wang, Liangxiao Jiang, and Chaoqun Li. Adapting naive bayes tree for text classification. *Knowl. Inf. Syst.*, 44(1):77–89, 2015.

Geoffrey I. Webb, Janice R. Boughton, and Zhihai Wang. Not so naive bayes: Aggregating one-dependence estimators. *Mach. Learn.*, 58(1):5–24, 2005.

Xindong Wu, Vipin Kumar, J. Ross Quinlan, Joydeep Ghosh, Qiang Yang, Hiroshi Motoda, Geoffrey J. McLachlan, Angus F. M. Ng, Bing Liu, Philip S. Yu, Zhi-Hua Zhou, Michael S. Steinbach, David J. Hand, and Dan Steinberg. Top 10 algorithms in data mining. *Knowl. Inf. Syst.*, 14(1): 1–37, 2008.

Nayyar Abbas Zaidi, Jesús Cerquides, Mark James Carman, and Geoffrey I. Webb. Alleviating naive bayes attribute independence assumption by attribute weighting. *J. Mach. Learn. Res.*, 14 (1):1947–1988, 2013.

Bohan Zhang, Scott Sanner, Mohamed Reda Bouadjenek, and Shagun Gupta. Bayesian networks for data integration in the absence of foreign keys. *IEEE Trans. Knowl. Data Eng.*, 32(4):803–808, 2020.

Cheng Zhang, Judith Bütepage, Hedvig Kjellström, and Stephan Mandt. Advances in variational inference. *IEEE Trans. Pattern Anal. Mach. Intell.*, 41(8):2008–2026, 2019.

Huan Zhang and Liangxiao Jiang. Fine tuning attribute weighted naive bayes. *Neurocomputing*, 488:402–411, 2022.

A TRAINING ALGORITHM OF MCENB

Algorithm 1 MCENB - training($\boldsymbol{\mathcal{D}}$, P, η)

Input: $\boldsymbol{\mathcal{D}} = \{\mathcal{X}, \boldsymbol{c}\}$ - the training set, P - the number of iterations, η - the learning rate.
Output: $\boldsymbol{w}$ - the weight vector, $q(\boldsymbol{z}|\boldsymbol{x})$ - the encoder of MCE.

1: Initialize the parameters of MCE as $\mathcal{W}_1$;
2: **for** p = 1 to P **do**
3: **for** i = 1 to n **do**
4: Generate the *i*-th embedding vector $\boldsymbol{z}_i$ by $q(\boldsymbol{z}|\boldsymbol{x})$ with $\mathcal{W}_p$;
5: Transform the class label of c_i into the one-hot class vector $\boldsymbol{v}_i$ and concatenate $\boldsymbol{z}_i$ with it;
6: Generate the reconstructed attribute value vector $\hat{\boldsymbol{x}}_i$ by $p(\boldsymbol{x}|\boldsymbol{z}, \boldsymbol{v})$ with $\mathcal{W}_p$;
7: **end for**
8: Calculate the value of the total loss $\mathcal{L}$ by Eqs. (3)-(7);
9: **if** $\mathcal{L}$ shows no improvement for 10 consecutive epochs **then**
10: break;
11: **end if**
12: Use η to update the parameters of MCE by Eq. (8);
13: **end for**
14: **for** i = 1 to n **do**
15: Generate the embedding vector $\boldsymbol{z}_i$ by $q(\boldsymbol{z}|\boldsymbol{x})$;
16: Concatenate $\boldsymbol{x}_i$ with $\boldsymbol{z}_i$;
17: **end for**
18: **for** c = 1 to k **do**
19: Estimate the prior probability $P(c)$ by Eq. (10);
20: **for** j = 1 to $2m$ **do**
21: Estimate the conditional probability $P(a_j|c)$ by Eq. (11);
22: **end for**
23: **end for**
24: Assign random values between 0 and 1 to weights in the weight vector $\boldsymbol{w}$;
25: Optimize the initialized weight vector $\boldsymbol{w}$ by Eqs. (12)-(16);
26: **return** $\boldsymbol{w}$, $q(\boldsymbol{z}|\boldsymbol{x})$.

In **Algorithm 1**, line 1 initializes the parameters in MCE with a time complexity of $O(m+k)$. Lines 2-13 train an MCE with a time complexity of $O(P(nm(m+k)+n^2mk))$. Lines 14-17 generate new attributes and augment original attributes with a time complexity of $O(nm^2)$. Lines 18-23 train a GNB with a time complexity of $O(nmk)$. Lines 24-25 weight augmented attributes with a time complexity of $O(\beta(nmk))$, where $\beta(nmk)$ has a linear relationship with nmk. Due to n usually being greater than m and m usually being greater than k, considering only the highest-order terms, the overall time complexity of **Algorithm 1** is $O(Pn^2mk)$.

B CLASSIFICATION ALGORITHM OF MCENB

Algorithm 2 MCENB - classification($\boldsymbol{x}$, $\boldsymbol{w}$, $q(\boldsymbol{z}|\boldsymbol{x})$)

Input: $\boldsymbol{x}$ - a test instance, $\boldsymbol{w}$ - the weight vector, $q(\boldsymbol{z}|\boldsymbol{x})$ - the encoder of MCE.
Output: $\hat{c}(\boldsymbol{x})$ - the predicted class label of $\boldsymbol{x}$.

1: Generate the embedding vector $\boldsymbol{z}$ by $q(\boldsymbol{z}|\boldsymbol{x})$;
2: Concatenate $\boldsymbol{x}$ with $\boldsymbol{z}$;
3: **for** c = 1 to k **do**
4: Estimate the prior probability $P(c)$ by Eq. (10);
5: **for** j = 1 to $2m$ **do**
6: Estimate the conditional probability $P(a_j|c)$ by Eq. (11);
7: **end for**
8: **end for**
9: Predict the class label $\hat{c}(\boldsymbol{x})$ of $\boldsymbol{x}$ by Eq. (9);
10: **return** $\hat{c}(\boldsymbol{x})$.

In **Algorithm 2**, lines 1-2 generate new attributes and augment original attributes with a time complexity of $O(m^2)$. Lines 3-8 estimate $P(c)$ and each $P(a_j|c)$ with a time complexity of $O(mk)$.

Line 9 predicts $\hat{c}(x)$ with a time complexity of $O(m)$. Considering only the highest-order terms, the overall time complexity of **Algorithm 2** is $O(m^2)$.

C DESCRIPTIONS OF 24 REAL-WORLD DATASETS USED IN EXPERIMENTS

Dataset	#Attributes	#Classes	#Instances
appendicitis	7	2	106
balance	4	3	625
banana	2	2	5300
cleveland	13	5	297
ecoli	7	8	336
glass	9	7	214
iris	4	3	150
led7digit-01	7	10	500
magic	10	2	19020
movement_libras	90	15	360
phoneme	5	2	5404
pima	8	2	768
ring	20	2	7400
segement	19	7	2310
sonar	60	2	208
spambase	57	2	4597
texture	40	11	5500
titanic	3	2	2201
twonorm	20	2	7400
wdbc	30	2	569
wine	13	3	178
winequality-red	11	11	1599
winequality-white	11	11	4898
yeast	8	10	1484

This table provides detailed descriptions of datasets used in our experiments, where #Attributes is the number of attributes, #Classes is the number of classes and #Instances is the number of instances.