This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2018.2836440, IEEE Transactions on Knowledge and Data Engineering

A Correlation-based Feature Weighting Filter for Naive Bayes

Liangxiao Jiang, Lungan Zhang, Chaoqun Li, Jia Wu

Abstract—Due to its simplicity, efficiency and efficacy, naive Bayes (NB) has continued to be one of the top 10 algorithms in the data mining and machine learning community. Of numerous approaches to alleviate its conditional independence assumption, feature weighting has placed more emphasis on highly predictive features than those that are less predictive. In this paper, we argue that for NB highly predictive features should be highly correlated with the class (maximum mutual relevance), yet uncorrelated with other features (minimum mutual redundancy). Based on this premise, we propose a correlation-based feature weighting (CFW) filter for NB. In CFW, the weight for a feature is proportional to the difference between the feature-class correlation (mutual relevance) and the average feature-feature intercorrelation (average mutual redundancy). Experimental results show that NB with CFW significantly outperforms NB and all the other existing state-of-the-art feature weighting filters used to compare. Compared to feature weighting wrappers for improving NB, the main advantages of CFW are its low computational complexity (no search involved) and the fact that it maintains the simplicity of the final model. Besides, we apply CFW to text classification and have achieved remarkable improvements.

Index Terms—feature weighting; naive Bayes; correlation; mutual information; mutual relevance; mutual redundancy.

1 Introduction

Bayesian networks [1], [2], [3] are often used for classification problems, in which a learner attempts to construct a classifier from a given set of training instances with class labels. Assume that A_1, A_2, $\cdots$, A_m are m feature variables, a test instance x can be represented by a feature vector $< a_1, a_2, \cdots, a_m >$, where a_i is the value of A_i. Let C represent the class variable and c represent the value that C takes, Bayesian network classifiers (BNC) use Equation 1 to classify x.

$$c(x) = \arg\max_{c \in C} P(c)P(a_1, a_2, \cdots, a_m|c), \tag{1}$$

where $c(x)$ is the class label of x predicted by BNC.

Assume that all features are fully independent given the class, the resulting BNC is called naive Bayes (NB). NB uses Equation 2 to classify x.

$$c(x) = \arg\max_{c \in C} P(c) \prod_{i=1}^{m} P(a_i|c). \tag{2}$$

In NB, each feature variable A_i $(i = 1, 2, \cdots, m)$ has the class variable C as its parent, but does not have any parent from other feature variables. Because the values of the prior probability $P(c)$ and the conditional probability $P(a_i|c)$ can be easily estimated from the given training instances, NB is easy and efficient to learn. It is also, however, surprisingly effective [4], [5], [6], [3], [7]. This fact raises the question of whether an improved NB with less restrictive conditional independence assumptions can perform even better. To answer this question, a mass of enhancements to NB have been proposed to help alleviate its conditional independence assumption. The related improved approaches can be broadly divided into six main categories:

1) Structure extension [3], [8], [9], [10], [11], [12];
2) Feature selection [13], [14], [15], [16], [17], [18];
3) Feature weighting [19], [20], [21], [22], [23], [24];
4) Instance selection [25], [26], [27];
5) Instance weighting [28], [29], [30];
6) Fine Tuning [31], [32], [33].

In this paper, we focus our attention on feature weighting and find that all of the existing feature weighting methods improve indeed the classification accuracy of NB at the expense (to a greater or lesser degree) of computational complexity and/or simplicity of the final model. The purpose of this paper is to propose a simple, efficient and effective method for setting feature weights for use with NB. In addition, since feature weighting has generally placed more emphasis on highly predictive features than those that are less predictive, in this paper we argue that for NB highly predictive features should be highly correlated with the class (maximum mutual relevance), yet uncorrelated with other features (minimum mutual redundancy). Based on these premises, we propose a correlation-based feature weighting (CFW) filter for NB. In CFW, the weight for a feature is proportional to the difference between the feature-class

- *L. Jiang is with Department of Computer Science, China University of Geosciences, Wuhan 430074, China. E-mail: ljiang@cug.edu.cn*
- *L. Zhang is with Hubei Key Laboratory of Intelligent Geo-Information Processing, China University of Geosciences, Wuhan 430074, China. E-mail: lunganzhang@gmail.com*
- *C. Li is with Department of Mathematics, China University of Geosciences, Wuhan 430074, China. E-mail: chqli@cug.edu.cn*
- *J. Wu is with Department of Computing, Macquarie University, Sydney, NSW 2109, Australia. E-mail: jia.wu@mq.edu.au*

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2018.2836440, IEEE Transactions on Knowledge and Data Engineering

correlation (mutual relevance) and the average feature-feature intercorrelation (average mutual redundancy). In other words, in CFW the features with maximum mutual relevance and minimum average mutual redundancy are considered to be highly predictive features and thus have greater feature weights. More important, the extensive empirical studies on a collection of 36 UCI datasets and 15 text datasets validate the effectiveness of the proposed feature weighting filter.

The rest of the paper is organized as follows. Section 2 provides a formal description of the feature weighted NB and a compact survey on the state-of-the-art feature weighting methods including filters and wrappers. Section 3 proposes a correlation-based feature weighting filter. Section 4 illustrates CFW using two artificial examples. Section 5 adapts CFW to text classification. Section 6 describes in detail the experimental setup and results. Section 7 concludes the paper and outlines the main directions for future research.

2 Feature Weighted Naive Bayes

The feature weighting approach weights predictive features differently according to their significance, and the resulting model is called feature weighted naive Bayes (FWNB). FWNB incorporates feature weights (representing the significance of features) into the formula gives:

$$c(x) = \arg\max_{c \in C} P(c) \prod_{i=1}^{m} P(a_i|c)^{W_i}, \tag{3}$$

where $W_i \in R^+$ is the weight of the ith feature A_i.

Now, the only question left to answer is how to define the weight of each predictive feature, which is crucial in constructing FWNB and has attracted more and more attention from researchers. The study of feature weighting is a relatively mature field in the data mining and machine learning community, and a large number of existing feature weighting methods prevent us from presenting them exhaustively. Here we only provide a compact survey on the state-of-the-art feature weighting methods specially designed for naive Bayes.

To the best of our knowledge, the earliest feature weighting method specially designed for naive Bayes is by Ferreira et al. [34]. However, it assigns a weight to each feature value rather than each feature and therefore is not strictly a feature weighting method but a feature value weighting method. In order to strictly perform feature weighting, Zhang and Sheng [19] proposed a gain ratio-based feature weighting method (GRFW) at first. They argued that a feature with higher gain ratio deserves a larger weight and therefore set the weight of each feature to the gain ratio of the feature relative to the average gain ratio across all features. The formula is shown in Equation 4.

$$W_i = \frac{GR(A_i)}{\frac{1}{m}\sum_{i=1}^{m} GR(A_i)}, \tag{4}$$

where m is the number of features, $GR(A_i)$ is the gain ratio of using feature A_i to partition the given training instances, which is then simply the information gain of A_i divided by its split information.

Hall [20] proposed a decision tree-based feature weighting method (DTFW). In DTFW, the weight for a feature is inversely proportional to the minimum depth at which it is tested in the built unpruned decision tree, and then the estimated weights are stabilized by averaging across 10 decision trees learned on data samples generated by bootstrapping 50% from the training data. As a result, DTFW assigns the weight to feature A_i as:

$$W_i = \frac{1}{T}\sum_{t=1}^{T} \frac{1}{\sqrt{d_{ti}}}, \tag{5}$$

where d_{ti} is the minimum depth at which feature A_i is tested in the built unpruned decision tree t, and T is the total number of the built decision trees. Note that the root node of the tree has depth 1 and the weight of a feature is set to zero if it does not appear in the built unpruned decision tree.

Lee et al. [21] proposed a Kullback-Leibler measure-based feature weighting method (KLMFW). KLMFW assumes that when a certain feature value a_i is observed, it gives a certain amount of information to the class variable C and then uses the Kullback-Leibler measure to calculate the information content of a feature value a_i as: $KL(C|a_i) = \sum_c P(c|a_i) \log \frac{P(c|a_i)}{P(c)}$. Then, the weight of feature A_i can be defined as the weighted average of the Kullback-Leibler measures across all feature values of A_i. In order to keep their ranges realistic, the learned feature weights are finally normalized by their mean. The formula is shown in Equation 6.

$$\begin{aligned} W_i &= \frac{1}{Z}\sum_{a_i} P(a_i) KL(C|a_i) \\ &= \frac{1}{Z}\sum_{a_i} P(a_i) \sum_{c} P(c|a_i) \log \frac{P(c|a_i)}{P(c)} \\ &= \frac{1}{Z}\sum_{a_i}\sum_{c} P(a_i) P(c|a_i) \log \frac{P(c|a_i)}{P(c)} \\ &= \frac{1}{Z}\sum_{a_i}\sum_{c} P(a_i, c) \log \frac{P(a_i, c)}{P(a_i)P(c)}, \end{aligned} \tag{6}$$

where $Z = \frac{1}{m}\sum_{i=1}^{m} w_i$ is a normalization constant. Although the authors present two implementations in their paper, above feature weighting version without split information presents the best performance according to their experimental results.

Jiang et al. [24] proposed a deep feature weighting method (DFW). DFW uses a correlation-based feature selection (CFS) filter [35] to select the best feature subset from the whole space of features and then assigns larger weights to the features in the selected feature subset and smaller weights to others. For simplicity, in their paper they simply set the weights of the selected features to 2

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2018.2836440, IEEE Transactions on Knowledge and Data Engineering

and 1 to others. The formula is shown in Equation 7.

$$W_i = \begin{cases} 2, & if\ A_i\ is\ selected \\ 1, & otherwise \end{cases} . \quad (7)$$

Different from all the other existing feature weighting methods, DFW is the only one to incorporate the learned feature weights not only into the classification formula of naive Bayes but also into its conditional probability estimates. Note that, the authors applied the proposed deep feature weighting method to some state-of-the-art naive Bayes text classifiers and have also achieved remarkable improvements. To our knowledge, this kind of deep feature weighting method is more suitable for naive Bayes text classifiers such as multinomial naive Bayes [36] rather than standard naive Bayes.

It can be seen that all above feature weighting methods directly compute feature weights according to the heuristics based on general data characteristics before the feature weighted naive Bayes is run and therefore are feature weighting filters. In addition to them, there exists another category of feature weighting methods which use the performance feedback from the feature weighted naive Bayes to optimize feature weights as a whole. We call them feature weighting wrappers. For example, Wu and Cai [22] proposed a differential evolution-based feature weighting wrapper (DEFW), which conducts a differential evolution search to optimize feature weights by maximizing the classification accuracy of the final model. Zaidi et al. [23] proposed a conditional log likelihood-based feature weighting wrapper (CLLFW) and a mean squared error-based feature weighting wrapper (MSEFW), which employ gradient descent searches to optimize feature weights by maximizing the conditional log likelihood or minimizing the mean squared error of the final model, respectively.

3 Correlation-based Feature Weighting

In feature weighting methods, each feature is assigned a different weight according to its importance. This kind of feature weighting corresponds to stretching the axes in the feature space, and the amount by which each axis is stretched is determined by the importance of each feature. This process of stretching the axes in order to optimize the performance of naive Bayes (NB) provides a mechanism for inspiring more relevant features and suppressing less relevant features.

Appropriate feature weights can reduce the error that results from violations of the feature independence assumption required by NB. Obviously, if a set of training data include a set of features that are identical to one another, the error due to the violation of the feature independence assumption can be removed by assigning weights that sum to 1.0 to the set of features in the set. For example, the weight for one of the features, A_i could be set to 1.0, and that of the remaining features that are identical to A_i set to 0.0. This is equivalent to removing the remaining features from the training data, namely feature selection. Feature weighting is strictly more powerful than feature selection, as it is possible to obtain identical results to feature selection by setting the weights of selected features to 1.0 and of unselected features to 0.0, and assignment of other weights can result in NB that cannot be expressed using feature selection. In a word, feature weighting assigns a continuous positive weight to each feature, and thus is a more flexible approach than feature selection.

Since feature weighting methods alleviate naive Bayes' feature independence assumption assumption by assigning greater weights to highly predictive features than those that are less predictive. In this paper, we argue that for naive Bayes highly predictive features should be highly correlated with the class (maximum mutual relevance), yet uncorrelated with other features (minimum mutual redundancy). Based on this premise, we propose a correlation-based feature weighting (CFW) filter for naive Bayes. In CFW, the weight for a feature is proportional to the difference between the feature-class correlation (mutual relevance) and the average feature-feature intercorrelation (average mutual redundancy). Thus it can be seen that how to measure (define) the correlation between each pair of random variables is crucial and therefore our research should start from answering this question.

For simplicity, here we restrict our discussion to discrete (categorical) variables. For continuous (numeric) random variables, we discretize them using the Fayyad & Irani's MDL method [37] implemented in the WEKA platform [38]. Then we use mutual information to measure the correlation between each pair of random discrete variables, and therefore the feature-class correlation and the feature-feature intercorrelation can be respectively defined as:

$$I(A_i; C) = \sum_{a_i} \sum_{c} P(a_i, c) \log \frac{P(a_i, c)}{P(a_i)P(c)}, \quad (8)$$

$$I(A_i; A_j) = \sum_{a_i} \sum_{a_j} P(a_i, a_j) \log \frac{P(a_i, a_j)}{P(a_i)P(a_j)}, \quad (9)$$

where C is the class variable, A_i and A_j are two different feature variables, c, a_i and a_j represent the values that they take, respectively.

Since a highly predictive feature should be highly correlated with the class, yet uncorrelated with other features, its weight should be proportional to the difference between the feature-class correlation and the average feature-feature intercorrelation defined by Equation 10.

$$D_i = \underbrace{NI(A_i; C)}_{relevance} - \underbrace{\frac{1}{m-1} \sum_{j=1 \wedge j \neq i}^{m} NI(A_i; A_j)}_{average\ redundancy}, \quad (10)$$

where $NI(A_i; C)$ is the normalized $I(A_i; C)$ representing mutual relevance and $NI(A_i; A_j)$ is the normalized $I(A_i; A_j)$ representing mutual redundancy. Note that

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2018.2836440, IEEE Transactions on Knowledge and Data Engineering

normalization is implemented for the calculations of the mutual information values to keep their ranges realistic and the consistency between possible features. The formulas are shown in Equations 11 and 12, respectively.

$$NI(A_i; C) = \frac{I(A_i; C)}{\frac{1}{m}\sum_{i=1}^{m} I(A_i; C)}. \quad (11)$$

$$NI(A_i; A_j) = \frac{I(A_i; A_j)}{\frac{1}{m(m-1)}\sum_{i=1}^{m}\sum_{j=1\wedge j\neq i}^{m} I(A_i; A_j)}. \quad (12)$$

Because the difference D_i defined by Equation 10 may be negative and yet at the same time the weights required by feature weighted naive Bayes are positive, we finally employ a standard logistic sigmoid function to transform its value into the range $(0, 1)$. Thus, the final form of the weight of feature A_i is:

$$W_i = \frac{1}{1 + e^{-D_i}}. \quad (13)$$

Now, the detailed learning algorithm for our correlation-based feature weighting (CFW for short) can be briefly depicted as Algorithm 1:

Algorithm 1 : Correlation-based Feature Weighting

Input: a training dataset
1: **for** each feature A_i $(i = 1, 2, \cdots, m)$ **do**
2: Compute $I(A_i; C)$ by Equation 8
3: **end for**
4: **for** each pair of features A_i and A_j $(j \neq i)$ **do**
5: Compute $I(A_i; A_j)$ by Equation 9
6: **end for**
7: **for** each feature A_i $(i = 1, 2, \cdots, m)$ **do**
8: Compute $NI(A_i; C)$ by Equation 11
9: **end for**
10: **for** each pair of features A_i and A_j $(j \neq i)$ **do**
11: Compute $NI(A_i; A_j)$ by Equation 12
12: **end for**
13: **for** each feature A_i $(i = 1, 2, \cdots, m)$ **do**
14: Compute D_i by Equation 10
15: Compute W_i by Equation 13
16: **end for**
Output: all feature weights W_i $(i = 1, 2, \cdots, m)$

Once all feature weights are obtained by Algorithm 1, we can then use them to construct a feature weighted naive Bayesian classifier. The formula is the same as Equation 3. We repeat it here for convenience:

$$c(x) = \arg\max_{c \in C} P(c) \prod_{i=1}^{m} P(a_i|c)^{W_i}, \quad (14)$$

where the prior probability $P(c)$ and the conditional probability $P(a_i|c)$ are estimated by the same m-estimation employed by CLLFW and MSEFW [23]:

$$P(c) = \frac{\sum_{j=1}^{n} \delta(c_j, c) + 1/k}{n + 1}, \quad (15)$$

TABLE 1
Feature weighting methods used to compare.

Name	Type	Relevance	Redundancy	Search
CFW	Filter	Yes	Yes	No
KLMFW	Filter	Yes	No	No
GRFW	Filter	Yes	No	No
DTFW	Filter	Yes	Yes	No
DFW	Filter	Yes	Yes	Yes
DEFW	Wrapper	Yes	Yes	Yes
CLLFW	Wrapper	Yes	Yes	Yes
MSEFW	Wrapper	Yes	Yes	Yes

$$P(a_i|c) = \frac{\sum_{j=1}^{n} \delta(a_{ji}, a_i)\delta(c_j, c) + 1/n_i}{\sum_{j=1}^{n} \delta(c_j, c) + 1}, \quad (16)$$

where k is the number of classes, n is the number of training instances, n_i is the number of values of A_i, c_j is class label of the *j*th training instance, a_{ji} is the *i*th feature value of the *j*th training instance, and the indicator function $\delta(\alpha, \beta)$ is one if $\alpha = \beta$ and zero otherwise.

From the algorithm above, we know that the training process of naive Bayes with our correlation-based feature weighting is very similar to standard naive Bayes, except for some additional training time to compute all feature weights w_i $(i = 1, 2, \cdots, m)$. According to Algorithm 1, the execution time for computing these weights is proportional to $mkv + m^2v^2 + m + m^2 + m$, where v the average number of values for a feature. If we only take the highest order term, the training time complexity for obtaining these weights is $O(m^2v^2)$ only. Thus it can be seen that our correlation-based feature weighting is indeed simple and efficient. More important, the experimental results in the next section validate its effectiveness.

Table 1 briefly summarizes all feature weighting methods used to compare. Compared to KLMFW and GRFW, CFW takes the mutual redundancy between each pair of features into account. Compared to DTFW, CFW does not have to construct multiple unpruned decision trees to compute feature weights. Compared to DFW, DEFW, CLLFW and MSEFW, CFW maintains the computational complexity and simplicity (no search involved) that characterizes naive Bayes. In a word, our work provides a simple, efficient and effective filter method for setting feature weights for use with NB.

4 Two Artificial Examples

Before discussing the performance of our proposed correlation-based feature weighting (CFW) on standard benchmark datasets, it could be helpful to get some intuitive feeling and illustrative results on CFW through two artificial examples. Thus, this section compares the behaviour of our proposed correlation-based feature weighting (CFW) to that of standard naive Bayes (NB) on two artificially generated datasets. In particular, we are interested in how sensitive the proposed CFW is to different levels of correlated features. We also experimentally argue that for naive Bayes highly predictive

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2018.2836440, IEEE Transactions on Knowledge and Data Engineering

TABLE 2
Description of two Monk's problems.

Feature information:	
class:	0, 1
A_1:	1, 2, 3
A_2:	1, 2, 3
A_3:	1, 2
A_4:	1, 2, 3
A_5:	1, 2, 3, 4
A_6:	1, 2
Target concepts:	
Monk-1:	$(A_1 = A_2)$ or $(A_5 = 1)$
Monk-2:	Exactly two of $\{A_1 = 1, A_2 = 1, A_3 = 1, A_4 = 1, A_5 = 1, A_6 = 1\}$
Size of the training set / Size of the test set:	
Monk-1:	124/432
Monk-2:	169/432

TABLE 3
The relevance, average redundancy and weight of each feature computed by CFW.

	Monk-1					
	A_1	A_2	A_3	A_4	A_5	A_6
relevance:	1.1293	0.0876	0.0706	0.3948	4.3063	0.0114
redundancy:	1.6903	1.2032	0.1634	1.1302	1.4003	0.4126
weight:	0.3633	0.2468	0.4768	0.3240	0.9481	0.4010
	Monk-2					
	A_1	A_2	A_3	A_4	A_5	A_6
relevance:	0.4851	0.3175	0.1364	2.0229	2.2312	0.8068
redundancy:	1.1709	1.4525	0.5098	1.0838	1.4699	0.3132
weight:	0.3350	0.2432	0.4077	0.7189	0.6817	0.6210

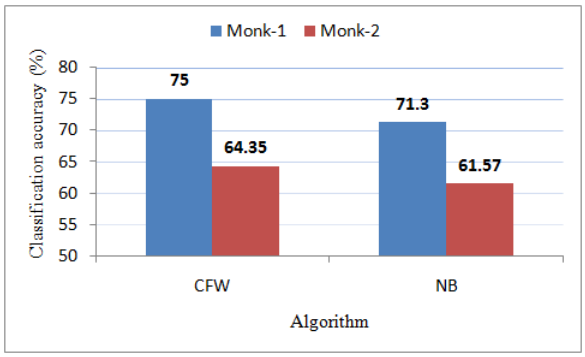


Fig. 1. Classification accuracy comparisons for CFW versus NB on two Monk's problems.

features should be highly correlated with the class, yet uncorrelated with other features.

In our experiments, two Monk's problems [39], [40] are chosen to generate two artificial datasets. These two Monk's problems are challenging artificial domains that have been widely used to compare the performance of machine learning algorithms. They involve irrelevant features and high degrees of feature interaction. Each Monk's problem uses the same representation and contains 432 instances described by six nominal features. For each problem there is a standard training and test set. Table 2 shows a brief description of them.

Table 3 shows the relevance, average redundancy and weight of each feature computed by CFW. Figure 1 graphically shows the detailed results comparing classification accuracy on two Monk's problems. From Table 2, Table 3 and Figure 1, we can see that:

1) The Monk-1 problem is difficult due to the interaction between the first two features and the fact that only one value of A_5 is useful. Note that A_3, A_4 and A_6 are irrelevant to the concept.
2) The Monk-2 problem is difficult due to the pairwise feature interactions and the fact that only one value of each feature is useful. Note that all six features are relevant to the concept.
3) In the Monk-1 problem, although A_1 and A_2 are all relevant to the concept, their average redundancies are high as well (1.6903 and 1.2032, respectively), and thus their weights are relatively low (0.3633 and 0.2468, respectively). A_5 is highly relevant to the concept (its relevance reaches to 4.3063.), and thus it gains the biggest weight (0.9481). The classification accuracy of CFW is 75%, which is markedly higher than that of NB (71.3%).
4) In the Monk-2 problem, All six features are relevant to the concept, but they involve high degrees of feature interaction. The average redundancies of A_1 and A_2 reach to 1.1709 and 1.4525, respectively, and thus their weights are the lowest (0.3350 and 0.2432, respectively). A_4 and A_5 is highly relevant to the concept (its relevances reach to 2.0229 and 2.2312, respectively), and thus they achieve the biggest weight (0.7189 and 0.6817, respectively). The classification accuracy of CFW is 64.35%, which is remarkably higher than that of NB (61.57%).
5) These illustrative results suggest that the proposed CFW is really sensitive to different levels of correlated features. We also experimentally prove that for naive Bayes highly predictive features are indeed highly correlated with the class, yet uncorrelated with other features as much as possible.

5 Adapting Correlation-based Feature Weighting for Text Classification

In principle, the proposed CFW can be developed as a meta learning method and then be used to improve most of other feature weighted models. In this section, we adapt it for improving two state-of-the-art naive Bayes text classifiers [41], [42], [43]: multinomial naive Bayes (MNB) [36] and complement naive Bayes (CNB) [44]. For simplicity, we denote the resulting models CFWMNB and CFWCNB, respectively.

Given a test document x, represented by a word frequency vector $< f_1, f_2, \cdots, f_m >$, CFWMNB uses Equation 17 to classify it.

$$c(d) = \arg\max_{c \in C} [\log P(c) + \sum_{i=1}^{m} W_i f_i \log P(w_i|c)], \quad (17)$$

where m is the vocabulary size in the text collection (the number of different words in all of the documents), w_i

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2018.2836440, IEEE Transactions on Knowledge and Data Engineering

$(i = 1, 2, \cdots, m)$ is the ith word that occurs in x, f_i is the frequency count of w_i, W_i $(i = 1, 2, \cdots, m)$ is the weight of w_i, the prior probability $P(c)$ and the conditional probability $P(w_i|c)$ can be estimated by Equations 18 and 19, respectively.

$$P(c) = \frac{\sum_{j=1}^{n} \delta(c_j, c) + 1/k}{n + 1}, \tag{18}$$

$$P(w_i|c) = \frac{\sum_{j=1}^{n} f_{ji}\delta(c_j, c) + 1/m}{\sum_{i=1}^{m}\sum_{j=1}^{n} f_{ji}\delta(c_j, c) + 1}, \tag{19}$$

where n is the number of training documents, k is the number of classes, c_j is the class label of the jth training document, f_{ji} is the frequency count of w_i in the jth training document, and $\delta(c_j, c)$ is a binary function defined as:

$$\delta(c_j, c) = \begin{cases} 1, & if\ c_j = c \\ 0, & otherwise \end{cases}. \tag{20}$$

CFWCNB uses Equation 21 to classify x.

$$c(d) = \arg\max_{c \in C}[-\log P(\overline{c}) - \sum_{i=1}^{m} W_i f_i \log P(w_i|\overline{c})], \tag{21}$$

where $\overline{c}$ is the complement classes of the class c (all classes except the class c), the prior probability $P(\overline{c})$ and the conditional probability $P(w_i|\overline{c})$ can be estimated by Equations 22 and 23, respectively.

$$P(\overline{c}) = \frac{\sum_{j=1}^{n} \delta(c_j, \overline{c}) + 1/k}{n + 1}, \tag{22}$$

$$P(w_i|\overline{c}) = \frac{\sum_{j=1}^{n} f_{ji}\delta(c_j, \overline{c}) + 1/m}{\sum_{i=1}^{m}\sum_{j=1}^{n} f_{ji}\delta(c_j, \overline{c}) + 1}, \tag{23}$$

where $\delta(c_j, \overline{c})$ is a binary function defined as:

$$\delta(c_j, \overline{c}) = \begin{cases} 1, & if\ c_j \in \overline{c}, namely\ c_j \neq c \\ 0, & otherwise \end{cases}. \tag{24}$$

Note that, in Equations 17 and 21, W_i $(i = 1, 2, \cdots, m)$ is the weight of word w_i learned by our proposed CFW. Now, the only question left to answer is how to use mutual information to measure the feature-class correlation and the feature-feature intercorrelation. To the best of our knowledge, the feature values (word frequencies) required by mutual information are generally nominal, while the feature values in text classification data are integral. A standard text classification data set is a collection of documents, where each document is represented by a word frequency vector. The text classification data is often a sparse matrix because the vocabulary of the words bag is quite vast. Each feature value in this matrix is zero or a positive integer. Moreover, most of these feature values are zero and the values greater than one are quite few. Therefore, in our definition of mutual information, we assume that all features only have two values of zero and nonzero. Then, we adapt Equations 8 and 9 as:

$$I(w_i; C) = \sum_{f_i \in \{0, \overline{0}\}} \sum_{c} P(f_i, c) \log \frac{P(f_i, c)}{P(f_i)P(c)}, \tag{25}$$

$$I(w_i; w_j) = \sum_{f_i \in \{0, \overline{0}\}} \sum_{f_j \in \{0, \overline{0}\}} P(f_i, f_j) \log \frac{P(f_i, f_j)}{P(f_i)P(f_j)}, \tag{26}$$

where C is the class variable, w_i and w_j are two different feature variables (words), c, f_i and f_j represent the values that they take, respectively. $f_i = 0$ indicates the absence of word w_i, and $f_i = \overline{0}$ indicates the presence of word w_i; $f_j = 0$ indicates the absence of word w_j, and $f_j = \overline{0}$ indicates the presence of word w_j.

6 Experiments and Results

The purpose of this section is to validate the effectiveness of our proposed CFW on a collection of 36 benchmark datasets and 15 text datasets, which all represent a wide range of domains and data characteristics.

6.1 Experiments on 36 UCI datasets

This subsection validates the effectiveness of our proposed CFW on a collection of 36 benchmark datasets shown in Table 4. Please note that, the missing values (%) column shows the percentage of a dataset's entries (number of features $\times$ number of instances) that have missing values [27], [35]. In our experiments, missing feature values were replaced with the modes of the nominal feature values and the means of the numerical feature values from the available data. Numerical feature values were discretized using the Fayyad & Irani's MDL method [37] implemented in the WEKA platform [38]. Besides, we manually deleted three useless features in advance: "Hospital Number" in "colic.ORIG", "instance name" in "splice" and "animal" in "zoo".

We conducted a group of experiments to compare CFW to NB, KLMFW [21], GRFW [19], DTFW [20], DFW [24], DEFW [22], CLLFW and MSEFW [23] in terms of classification accuracy (the percentage of test instances correctly classified). We implemented CFW, KLMFW, GRFW, DFW and DEFW on the Waikato Environment for Knowledge Analysis (WEKA) platform [38] and used the existing implementation of NB on the WEKA platform and the implementations of DTFW, CLLFW and MSEFW kindly provided by the original authors.

Table 5 shows the detailed comparison results in terms of classification accuracy. All classification accuracy estimates are obtained by averaging the results from 10 separate runs of stratified 10-fold cross-validation. Then, we conduct corrected paired two-tailed t-tests at 95% significance level [45] to compare CFW to each of its competitors: NB, KLMFW, GRFW, DTFW, DFW, DEFW, CLLFW and MSEFW. The symbols $\bullet$ and $\circ$ in the table denote statistically significant improvement or degradation over its competitors, respectively. The averages and the $Win/Tie/Lose$ $(W/T/L)$ values are summarized at the bottom of the table. The average (arithmetic mean) of each algorithm across all datasets provides a gross indicator of the relative performance in addition to the other statistics. Each entry's $W/T/L$ in the table implies

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2018.2836440, IEEE Transactions on Knowledge and Data Engineering

TABLE 4
36 UCI datasets used in our experiments.

Dataset name	Instance number	Numeric features	Nominal features	Class number	Missing values (%)
anneal	898	6	32	5	0.0
anneal.ORIG	898	6	32	5	63.3
audiology	226	0	69	24	2.0
autos	205	15	10	7	1.1
balance-scale	625	4	0	3	0.0
breast-cancer	286	0	9	2	0.3
breast-w	699	9	0	2	0.3
colic	368	7	15	2	23.8
colic.ORIG	368	7	20	2	18.7
credit-a	690	6	9	2	0.6
credit-g	1000	7	13	2	0.0
diabetes	768	8	0	2	0.0
Glass	214	9	0	7	0.0
heart-c	303	6	7	5	0.2
heart-h	294	6	7	5	20.4
heart-statlog	270	13	0	2	0.0
hepatitis	155	6	13	2	5.6
hypothyroid	3772	23	6	4	6.0
ionosphere	351	34	0	2	0.0
iris	150	4	0	3	0.0
kr-vs-kp	3196	0	36	2	0.0
labor	57	8	8	2	3.9
letter	20000	16	0	26	0.0
lymph	148	3	15	4	0.0
mushroom	8124	0	22	2	1.4
primary-tumor	339	0	17	21	3.9
segment	2310	19	0	7	0.0
sick	3772	7	22	2	6.0
sonar	208	60	0	2	0.0
soybean	683	0	35	19	9.8
splice	3190	0	61	3	0.0
vehicle	846	18	0	4	0.0
vote	435	0	16	2	5.6
vowel	990	10	3	11	0.0
waveform-5000	5000	40	0	3	0.0
zoo	101	1	16	7	0.0

that, compared to its competitors, CFW wins on W datasets, ties on T datasets, and loses on L datasets. From these results, we can see that:

1) Compared to NB, CFW is significantly better on 12 datasets and significantly worse on none. This suggests that feature dependencies captured in CFW is useful when setting weights for NB.
2) Compared to KLMFW, GRFW and DFW, CFW is significantly more accurate on 15, 14, and 8 datasets, respectively, and significantly less accurate on none as well.
3) Compared to DTFW, CFW for determining weights results in significantly higher classification accuracy on eight datasets and significantly lower classification accuracy on three only.
4) Compared to other three wrappers, CFW almost ties DEFW (two wins and five losses), and is a little worse than CLLFW (one win and eight losses) and MSEFW (zero win and seven losses).
5) For additional insight into the results, we observed the performance of CFW on the datasets "kr-vs-kp", "letter" and "mushroom", within which strong and high-order feature dependencies have been observed [25], [40], [46]. We can see that, on these datasets, the classification accuracies of CFW (93.58%, 75.22% and 99.19%) are notably better than those of NB (87.79%, 74.00% and 95.52%, respectively).

Yet at the same time, based on the accuracy results presented in Table 5, we can take advantage of KEEL Data-Mining Software Tool [47] [1] or R [2] or MATLAB [3] to complete the Wilcoxon signed-ranks test [48], [49] for thoroughly comparing each pair of algorithms. Table 6 summarizes the detailed comparison results. In Table 6, $\circ$ indicates that the algorithm in the column improves the algorithm in the corresponding row, and $\bullet$ indicates that the algorithm in the row improves the algorithm in the corresponding column. Lower diagonal level of significance $\alpha = 0.05$; Upper diagonal level of significance $\alpha = 0.1$. From these comparisons, we can see that:

1) The improvements achieved by KLMFW and GRFW are very limited in terms of classification accuracy. KLMFW performs even worse than NB. This suggests that only taking the mutual relevance between each feature and the class variable into account is not enough.
2) Another two feature weighting filters DTFW and DFW are notably better than KLMFW and GRFW. This suggests that capturing the mutual redundancy between each pair of features is very useful when setting feature weights for NB.
3) CFW significantly outperforms NB and other four existing state-of-the-art filters: KLMFW, GRFW, DTFW and DFW. This suggests that feature dependencies captured in CFW is more accurate than all the other existing filters used to compare.
4) Compared to the other three wrappers, only CLLFW and MSEFW are significant. The main advantages of CFW compared to wrappers are its low computational complexity (no search involved) and the fact that it maintains the simplicity of the final model. Thus, when high model performance is the sole concern, CLLFW and MSEFW should be appropriate alternatives. However, when the computational complexity and the simplicity of the final model are also important, CFW should be considered at first. Besides, CFW is a non-parametric learning algorithm, which makes CFW a very attractive alternative to parametric learning algorithms, such as DEFW, that require fine-tuning of some parameters to achieve good results.

Besides, in our experiments, we have also observed the performance of our proposed method in terms of the conditional log likelihood (CLL) [50], [51], [9], [52], and the area under the ROC curve (AUC) [53], [54], [55], [56]. Tables 7-10 show the detailed comparison results. From these comparison results, we can find that our proposed feature weighting method (CFW) is also very promising

1. http://sci2s.ugr.es/keel/
2. http://www.r-tutor.com/elementary-statistics/
3. http://cn.mathworks.com/help/stats/signrank.html

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2018.2836440, IEEE Transactions on Knowledge and Data Engineering

TABLE 5
Accuracy comparisons for CFW versus NB, KLMFW, GRFW, DTFW, DFW, DEFW, CLLFW and MSEFW.

Dataset	CFW	NB	KLMFW	GRFW	DTFW	DFW	DEFW	CLLFW	MSEFW
anneal	98.50	96.13 •	87.85 •	97.14 •	90.27 •	97.42 •	98.06	98.60	98.69
anneal.ORIG	94.60	92.66 •	85.77 •	88.74 •	93.59	91.66 •	93.72	93.29 •	93.69
audiology	74.22	71.40	70.82	71.62	70.11 •	73.83	72.75	78.08 ○	78.08
autos	77.95	72.30 •	75.28	72.77 •	75.23	76.18	76.78	80.43	80.09
balance-scale	73.76	71.08	71.27	71.32	72.38	71.99	69.26 •	71.08	71.08
breast-cancer	72.46	72.94	70.63	72.24	72.29	71.89	71.91	71.00	71.35
breast-w	97.14	97.25	97.43	97.30	97.20	97.31	96.74	96.84	96.51
colic	83.34	81.39	83.81	81.80	84.29	83.15	82.45	84.13	83.72
colic.ORIG	73.70	73.62	70.98	72.50	75.79	74.46	74.05	73.87	73.87
credit-a	86.99	86.25	85.99	85.54	87.23	86.54	86.65	86.59	86.23
credit-g	75.70	75.43	68.28 •	69.78 •	75.49	74.28	75.16	75.61	75.59
diabetes	78.01	77.85	76.81	77.82	78.72	78.70	77.88	78.54	78.48
glass	73.37	74.39	73.01	70.99	73.74	70.84	75.89	73.92	73.82
heart-c	82.94	83.60	83.90	82.98	83.20	82.71	82.41	83.11	83.73
heart-h	83.82	84.46	81.99	82.77	83.18	83.44	81.23	84.94	84.39
heart-statlog	83.44	83.74	84.33	84.37	83.11	83.33	83.70	83.48	84.74
hepatitis	85.95	84.22	83.69	83.81	85.52	84.41	86.26	86.74	86.61
hypothyroid	98.56	98.48	96.27 •	97.12 •	99.07 ○	98.06 •	99.10 ○	99.37 ○	99.37 ○
ionosphere	91.82	90.77	90.80	91.48	92.22	91.20	91.34	92.85	92.73
iris	94.40	94.47	94.33	94.33	94.93	94.47	94.47	94.60	94.33
kr-vs-kp	93.58	87.79 •	90.83 •	89.67 •	95.08 ○	91.88 •	94.31 ○	93.43	93.92
labor	92.10	93.13	90.53	91.10	87.33	94.00	93.70	95.97	95.60
letter	75.22	74.00 •	74.27 •	74.30 •	73.87 •	74.99	74.78 •	75.52 ○	75.55 ○
lymphography	84.81	84.97	80.07 •	80.22 •	79.40 •	82.56	84.62	84.30	84.48
mushroom	99.19	95.52 •	98.52 •	98.92 •	97.97 •	98.91 •	99.15	99.90 ○	99.90 ○
primary-tumor	47.20	47.20	45.64	45.76	45.28	44.55	47.17	47.52	48.53
segment	93.47	91.71 •	90.20 •	91.14 •	93.63	93.09	94.80 ○	95.15 ○	95.24 ○
sick	97.36	97.10	96.55 •	96.55 •	97.47	96.83 •	97.61	97.46	97.47
sonar	82.56	85.16	81.12	80.84	81.66	83.81	84.83	83.67	83.85
soybean	93.66	92.20 •	91.74 •	90.60 •	92.64	92.47	92.99	93.92	93.75
splice	96.19	95.42 •	94.50 •	94.00 •	96.03	95.99	95.83	96.05	96.28
vehicle	62.91	62.52	61.25 •	62.66	63.61	62.60	66.02 ○	68.20 ○	68.57 ○
vote	92.11	90.21 •	93.01	93.01	95.40 ○	92.62	95.10 ○	95.56 ○	95.52 ○
vowel	68.84	65.23 •	63.25 •	67.47	65.15 •	67.42	67.05	68.22	68.19
waveform-5000	83.11	80.72 •	79.64 •	80.39 •	81.06 •	80.96 •	83.73	84.58 ○	84.65 ○
zoo	95.96	93.98	93.87	94.17	88.04 •	91.30 •	95.75	95.85	95.75
Average	84.41	83.31	82.17	82.70	83.37	83.61	84.37	85.07	85.12
$W/T/L$	-	12/24/0	15/21/0	14/22/0	8/25/3	8/28/0	2/29/5	1/27/8	0/29/7

TABLE 6
Accuracy summary of the Wilcoxon test.

Algorithm	CFW	NB	KLMFW	GRFW	DTFW	DFW	DEFW	CLLFW	MSEFW
CFW	-	•	•	•	•	•		○	○
NB	○	-	•	•			○	○	○
KLMFW	○	○	-	○	○	○	○	○	○
GRFW	○			-	○	○	○	○	○
DTFW	○		•	•	-		○	○	○
DFW	○		•	•		-	○	○	○
DEFW		•	•	•		•	-	○	○
CLLFW	•	•	•	•	•	•	•	-	
MSEFW	•	•	•	•	•	•	•		-

in terms of the conditional log likelihood (CLL) and the area under the ROC curve (AUC).

Finally, we have also compared our proposed method with its existing competitors in terms of the elapsed training time (in milliseconds). Our experiments were conducted on a Linux machine with 3.2 GHz processor and 8 GB of RAM. The detailed comparison results are shown in Tables 11 and 12. Note that the meaning of the t-test results in Table 11 are opposite to those in Tables 5, 7, and 9. For the elapsed training time, a small number is better than a large number. Thus, in terms of the elapsed training time, the symbols ○ and • in the table denote statistically significant improvement or degradation over its competitors, respectively. From these comparison results, we can find that our proposed CFW is only slower than NB, KLMFW and GRFW and much faster than DTFW, DFW, DEFW, CLLFW and MSEFW.

6.2 Experiments on 15 text datasets

This subsection validates the effectiveness of our proposed CFW on a collection of 15 text datasets shown in Table 13. We designed two groups of experiments to compare the original versions MNB and CNB with our correlation-based feature weighted versions CFWMNB

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2018.2836440, IEEE Transactions on Knowledge and Data Engineering

TABLE 7
CLL comparisons for CFW versus NB, KLMFW, GRFW, DTFW, DFW, DEFW, CLLFW and MSEFW.

Dataset	CFW	NB	KLMFW	GRFW	DTFW	DFW	DEFW	CLLFW	MSEFW
anneal	-8.93	-15.16 •	-84.75 •	-17.57 •	-24.75 •	-18.92 •	-12.03 •	-7.22 ○	-7.41 ○
anneal.ORIG	-22.65	-24.77	-193.23 •	-75.18 •	-27.95 •	-40.89 •	-27.78 •	-20.49 ○	-20.74 ○
audiology	-45.89	-95.09 •	-90.26 •	-88.57 •	-33.64 ○	-94.59 •	-59.44 •	-36.87 ○	-36.87 ○
autos	-25.22	-45.41 •	-60.22 •	-51.62 •	-19.56 ○	-52.07 •	-24.32	-20.55	-20.75
balance-scale	-63.66	-53.86 ○	-54.79 ○	-54.79 ○	-58.59 ○	-51.02 ○	-64.93	-53.80 ○	-53.80 ○
breast-cancer	-24.32	-26.51 •	-37.03 •	-39.57 •	-23.52	-39.76 •	-23.44	-25.36	-24.90
breast-w	-14.62	-28.15 •	-32.27 •	-30.14 •	-14.33	-57.86 •	-15.71	-10.12 ○	-14.67
colic	-24.24	-39.45 •	-71.58 •	-85.32 •	-22.44 ○	-48.40 •	-23.43	-21.72 ○	-22.17
colic.ORIG	-26.11	-27.57	-68.96 •	-46.53 •	-23.89	-30.94 •	-25.32	-28.27 •	-27.93 •
credit-a	-34.27	-44.53 •	-107.00 •	-114.21 •	-33.58	-63.98 •	-33.89	-32.50	-33.34
credit-g	-72.75	-75.22	-186.05 •	-142.66 •	-72.17	-94.20 •	-72.13	-72.41	-72.61
diabetes	-51.28	-53.25	-81.86 •	-66.66 •	-51.04	-67.60 •	-51.25	-50.63	-50.66
glass	-21.44	-23.52	-28.05 •	-26.94 •	-22.27 •	-39.23 •	-22.62	-21.94	-23.23
heart-c	-15.97	-20.09 •	-30.12 •	-28.78 •	-16.52 •	-33.82 •	-17.08	-16.29	-16.36
heart-h	-15.46	-17.98 •	-52.00 •	-47.60 •	-16.82	-29.09 •	-17.96 •	-14.69	-14.78
heart-statlog	-14.34	-17.47 •	-25.58 •	-25.45 •	-14.83	-28.71 •	-14.73	-14.83	-14.81
hepatitis	-7.38	-10.56 •	-20.72 •	-19.92 •	-7.39	-19.41 •	-7.10	-7.51	-7.29
hypothyroid	-25.83	-29.68 •	-675.10 •	-774.81 •	-22.74 ○	-51.50 •	-23.14	-19.86 ○	-19.28 ○
ionosphere	-32.57	-64.58 •	-68.18 •	-65.52 •	-12.54 ○	-89.07 •	-29.52	-11.21 ○	-15.82 ○
iris	-3.04	-3.61	-4.03	-4.02	-3.35	-5.82	-4.12	-3.07	-2.97
kr-vs-kp	-126.60	-134.86 •	-107.45 ○	-110.83 ○	-157.12 •	-93.23 ○	-136.61 •	-117.43 ○	-118.35 ○
labor	-1.37	-1.00 ○	-2.25	-1.56	-2.78 •	-1.29	-1.42	-1.13	-1.23
letter	-2564.35	-3236.23 •	-4042.52 •	-3782.26 •	-2785.11 •	-5055.97 •	-2720.98 •	-2483.89 ○	-2515.85 ○
lymphography	-7.95	-8.96	-16.67 •	-14.09 •	-9.52 •	-15.38 •	-8.60	-8.17	-8.30
mushroom	-34.21	-152.60 •	-162.73 •	-141.09 •	-38.15	-119.07 •	-43.71	-7.59 ○	-8.59 ○
primary-tumor	-90.22	-94.58 •	-100.56 •	-99.06 •	-90.12	-138.40 •	-91.70	-94.46 •	-95.25 •
segment	-74.66	-149.03 •	-258.06 •	-222.09 •	-67.25 ○	-174.27 •	-54.22 ○	-43.92 ○	-48.58 ○
sick	-50.27	-62.51 •	-827.59 •	-807.15 •	-46.96 ○	-99.13 •	-47.41 ○	-44.25 ○	-44.18 ○
sonar	-11.18	-13.81	-71.88 •	-61.22 •	-11.48	-26.14 •	-9.98	-9.60	-9.52
soybean	-19.52	-37.87 •	-60.82 •	-45.23 •	-26.58 •	-69.61 •	-22.83 •	-15.11 ○	-14.87 ○
splice	-53.64	-67.13 •	-274.02 •	-303.24 •	-61.35 •	-94.28 •	-59.97 •	-57.20 •	-58.42 •
vehicle	-134.04	-227.39 •	-356.06 •	-303.87 •	-138.93 •	-339.12 •	-98.03 ○	-85.94 ○	-86.47 ○
vote	-17.68	-39.31 •	-47.02 •	-46.46 •	-9.99 ○	-39.37 •	-11.57 ○	-8.17 ○	-8.07 ○
vowel	-121.90	-131.33 •	-214.74 •	-134.89 •	-139.62 •	-146.34 •	-148.70 •	-119.29	-119.72
waveform-5000	-280.38	-521.19 •	-1305.62 •	-1192.89 •	-351.47 •	-939.56 •	-270.21 ○	-247.31 ○	-247.62 ○
zoo	-1.88	-1.76	-2.17	-1.62	-5.95 •	-3.39	-2.43 •	-1.72	-1.87
Average	-114.99	-155.45	-272.83	-252.04	-124.01	-230.87	-119.40	-106.51	-107.98
$W/T/L$	-	25/9/2	31/3/2	31/3/2	14/13/9	31/3/2	10 /21/5	3/16/17	3/18/15

TABLE 8
CLL summary of the Wilcoxon test.

Algorithm	CFW	NB	KLMFW	GRFW	DTFW	DFW	DEFW	CLLFW	MSEFW
CFW	-	•	•	•		•		○	○
NB	○	-	•	•	○	•	○	○	○
KLMFW	○	○	-	○	○	○	○	○	○
GRFW	○	○	•	-	○		○	○	○
DTFW		•	•	•	-	•		○	○
DFW	○	○	•		○	-	○	○	○
DEFW		•	•	•		•	-	○	○
CLLFW	•	•	•	•	•	•	•	-	•
MSEFW	•	•	•	•	•	•	•	○	-

and CFWCNB, respectively. We used the existing implementations of MNB and CNB in the WEKA platform [38] and implemented CFWMNB and CFWCNB on the WEKA platform. Figure 2 graphically shows the detailed comparison results. From our results, we can see that our correlation-based feature weighted versions CFWMNB and CFWCNB are also the best against the original versions MNB and CNB, respectively.

7 Conclusions and Further Research

Since NB makes the conditional independence assumption, we want to assign greater weights to those features that are highly correlated with the class, yet uncorrelated with other features. Based on this premise, we propose a correlation-based feature weighting (CFW) filter for NB. In CFW, the weight for a feature is proportional to the difference between the feature-class correlation (mutual relevance) and the average feature-feature intercorrelation (average mutual redundancy). The extensive experimental results show that CFW has a better overall performance compared to NB and all the other existing state-of-the-art feature weighting filters used to compare. Considering its computational complexity and the simplicity of the final model, CFW is a promising

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2018.2836440, IEEE Transactions on Knowledge and Data Engineering

TABLE 9
AUC comparisons for CFW versus NB, KLMFW, GRFW, DTFW, DFW, DEFW, CLLFW and MSEFW.

Dataset	CFW	NB	KLMFW	GRFW	DTFW	DFW	DEFW	CLLFW	MSEFW
anneal	99.49	99.20 •	99.04 •	98.91 •	98.86 •	99.39	99.38	99.57	99.57
anneal.ORIG	98.28	97.78	97.44 •	97.63 •	98.01	98.08	97.71	98.41	98.40
audiology	97.90	96.93 •	96.95 •	97.48	96.88 •	97.38	96.77 •	97.89	97.88
autos	94.36	92.34 •	94.11	93.39 •	94.56	94.39	92.97 •	94.67	94.40
balance-scale	87.39	87.74	87.37	87.39	87.36	88.13	84.14 •	87.73	87.73
breast-cancer	69.28	70.18	69.95	70.19	69.97	69.95	69.07	66.69	67.02
breast-w	99.26	99.25	99.17	99.19	99.23	99.23	99.23	99.26	99.24
colic	87.78	84.97 •	88.46	88.57	88.25	86.73 •	87.17	88.34	88.59
colic.ORIG	82.93	82.32	81.36	83.26	83.63	83.25	81.82	82.71	82.90
credit-a	92.91	92.21	92.60	92.60	93.06	92.75	92.86	93.34	93.16
credit-g	78.34	78.99	77.31	77.58	78.81	78.20	78.93	79.01	78.95
diabetes	84.40	84.64	83.56	84.28	84.68	84.97	84.46	84.70	84.76
glass	91.06	91.10	90.39	90.12	91.04	90.51	90.28	91.09	90.97
heart-c	91.69	91.27	90.90	90.82	91.43	90.71 •	90.99	91.61	91.50
heart-h	91.46	91.44	90.62	90.57	91.10	91.63	89.48	92.26	92.12
heart-statlog	91.48	91.47	90.82	90.82	91.02	90.84	91.18	91.14	91.16
hepatitis	91.12	91.25	90.17	90.09	89.04	90.78	90.43	89.71	90.56
hypothyroid	99.69	99.66	99.50 •	99.49 •	99.76	99.65	99.76	99.81 ○	99.82 ○
ionosphere	97.57	97.23	97.10 •	97.25	97.01	97.08	97.36	97.17	97.13
iris	98.95	98.95	98.97	98.97	99.01	98.97	98.55	98.82	99.07
kr-vs-kp	98.19	95.19 •	97.95 •	98.11	98.91 ○	97.74 •	98.65 ○	98.44	98.76 ○
labor	99.17	99.17	99.13	99.50	95.62	99.13	97.88	99.29	98.88
letter	98.12	97.89 •	97.90 •	97.89 •	97.80 •	98.06 •	97.96 •	98.22 ○	98.24 ○
lymphography	92.52	92.89	91.40	90.50	91.89	92.80	92.12	92.49	92.60
mushroom	99.96	99.80 •	99.95 •	99.89 •	99.96	99.88 •	99.91	100.00 ○	100.00 ○
primary-tumor	83.10	82.79	82.74	82.79	83.01	82.45	82.63	83.37	83.43
segment	99.41	99.15 •	98.86 •	98.99 •	99.46 ○	99.40	99.56 ○	99.67 ○	99.65 ○
sick	96.14	96.01	95.54	95.68	96.00	96.20	96.00	96.23	96.17
sonar	92.19	93.34	89.63 •	90.20 •	91.21	92.69	93.36	93.77	93.83
soybean	99.63	99.48 •	99.52 •	99.39 •	99.57	99.54 •	99.55	99.68	99.69
splice	99.51	99.35 •	99.30 •	99.23 •	99.47	99.44 •	99.41 •	99.47	99.44 •
vehicle	86.66	86.23 •	84.57 •	85.15 •	86.66	86.16 •	87.36 ○	89.27 ○	89.27 ○
vote	97.99	97.15 •	98.05	98.01	98.79 ○	98.10	98.18	99.03 ○	99.06 ○
vowel	96.16	95.41 •	94.54 •	95.84 •	95.33 •	95.75 •	95.25 •	96.15	96.12
waveform-5000	96.41	95.94 •	95.82 •	96.07 •	96.15 •	96.05 •	96.14 •	96.41	96.40
zoo	99.94	99.94	99.88	99.94	99.57	99.83	99.92	99.90	99.89
Average	93.35	93.02	92.79	92.94	93.11	93.22	92.96	93.48	93.51
$W/T/L$	-	14/22/0	15/21/0	13/23/0	5/28/3	10/26/0	7/26/3	0/30/6	1/28/7

TABLE 10
AUC summary of the Wilcoxon test.

Algorithm	CFW	NB	KLMFW	GRFW	DTFW	DFW	DEFW	CLLFW	MSEFW
CFW	-	•	•	•		•	•	○	○
NB	○	-	•					○	○
KLMFW	○	○	-	○	○	○		○	○
GRFW	○			-	○	○		○	○
DTFW			•	•	-			○	○
DFW	○		•	•		-		○	○
DEFW	○						-	○	○
CLLFW	•	•	•	•	•	•	•	-	
MSEFW	•	•	•	•	•	•	•		-

feature weighting method that could be used in many real-world applications.

As already pointed out, for simplicity, we assume that all features have discrete (nominal) values and have not missing values in the current version and thus all continuous features are discretized and all missing values are replaced with the modes (means) from the available data. However, in many real-world applications, continuous features and missing values are widespread and, therefore, extending it to handle applications with continuous features and missing values is a main direction for our future research. Besides, further enhancing the proposed CFW using some sophisticated methods such as weight adjustment is another direction for our future research.

ACKNOWLEDGEMENTS

Many thanks to Mark Hall and Nayyar A. Zaidi for kindly providing us with the implementations of their feature weighting methods. The work was partially supported by the Chenguang Program of Science and Technology of Wuhan (2015070404010202) and the Open Research Project of Hubei Key Laboratory of Intelligent Geo-Information Processing (KLIGIP201601).

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2018.2836440, IEEE Transactions on Knowledge and Data Engineering

TABLE 11
Elapsed training time comparisons for CFW versus NB, KLMFW, GRFW, DTFW, DFW, DEFW, CLLFW and MSEFW.

Dataset	CFW	NB		KLMFW		GRFW		DTFW		DFW		DEFW		CLLFW		MSEFW	
anneal	3.59	0.54	•	3.05		1.15	•	15.88		7.24		10426.64	○	164.30		223.52	○
anneal.ORIG	2.69	0.16	•	0.96	•	0.85	•	10.95	○	3.49		11040.85	○	102.71	○	208.25	○
audiology	5.28	0.03		0.83		0.61		8.88		3.79		16379.62	○	88.44	○	229.21	○
autos	0.72	0.08	•	0.38		0.26	•	3.49	○	1.56		1756.59	○	15.89	○	24.90	○
balance-scale	0.09	0.14		0.26		0.26		1.75	○	0.43		624.76	○	2.96	○	1.52	○
breast-cancer	0.18	0.01		0.34		0.19		2.25	○	0.51		336.76	○	3.89	○	4.23	○
breast-w	0.21	0.08		0.33		0.32		2.26	○	0.73	○	829.29	○	5.89	○	12.09	○
colic	0.53	0.01	•	0.32		0.25		4.03	○	0.84		935.45	○	7.67	○	9.54	○
colic.ORIG	0.99	0.10	•	0.39	•	0.34	•	4.66	○	1.18		1076.93	○	14.68	○	12.19	○
credit-a	0.46	0.11		0.48		0.33		5.63	○	0.97		1181.14	○	14.57	○	23.51	○
credit-g	1.00	0.07	•	0.68		0.88		11.75	○	1.60	○	2325.14	○	11.54	○	16.55	○
diabetes	0.23	0.08		0.38		0.37		4.14	○	0.55		876.09	○	4.13	○	6.17	○
glass	0.19	0.05		0.22		0.17		1.53	○	0.33		810.31	○	3.33	○	5.90	○
heart-c	0.22	0.14		0.26		0.20		2.48	○	0.53		1050.66	○	6.30	○	12.11	○
heart-h	0.25	0.02		0.14		0.17		2.05	○	0.49		1091.09	○	5.46	○	11.20	○
heart-statlog	0.11	0.05		0.20		0.18		2.15	○	0.54		452.20	○	3.30	○	4.44	○
hepatitis	0.25	0.04		0.26		0.18		1.86		0.66		356.54	○	4.05	○	5.33	○
hypothyroid	5.86	0.74	•	2.61	•	2.56	•	18.85	○	7.13		24199.29	○	333.55	○	565.26	○
ionosphere	1.69	0.07	•	0.39	•	0.29	•	4.02	○	2.57	○	1225.54	○	8.24	○	24.44	○
iris	0.12	0.02		0.12		0.17		0.52	○	0.27		156.78	○	1.83	○	2.64	○
kr-vs-kp	6.82	0.72	•	2.24	•	2.54	•	34.46	○	9.32	○	12620.29	○	154.08	○	236.20	○
labor	0.13	0.05		0.25		0.09		0.55		0.41		129.02	○	2.06	○	2.48	○
letter	29.13	3.14	•	11.92	•	17.76	•	420.78	○	65.53	○	416849.82	○	1141.41	○	3036.12	○
lymphography	0.35	0.05		0.21		0.16		1.97	○	0.70		551.69	○	5.50	○	8.69	○
mushroom	12.08	1.22	•	4.82	•	5.89	•	28.55	○	16.13	○	20103.70	○	120.74	○	244.91	○
primary-tumor	0.43	0.06	•	0.28		0.26		7.03	○	0.96	○	6115.86	○	14.22	○	31.73	○
segment	4.41	0.27	•	1.07	•	1.19	•	15.67	○	4.20		15097.75	○	89.93	○	168.87	○
sick	5.28	0.66	•	2.34	•	2.68	•	47.86	○	7.97	○	13409.19	○	237.50	○	330.69	○
sonar	1.59	0.06	•	0.45	•	0.33	•	4.53	○	5.11	○	1480.83	○	8.56	○	10.63	○
soybean	2.54	0.14	•	0.62	•	0.60	•	13.04	○	5.25	○	22120.94	○	140.66	○	373.59	○
splice	39.64	1.00	•	3.31	•	3.72	•	76.43	○	25.13	•	25772.12	○	259.12	○	400.95	○
vehicle	0.91	0.08	•	0.46	•	0.40	•	11.67	○	1.75	○	3046.69	○	13.84	○	28.97	○
vote	0.26	0.06		0.26		0.27		2.27	○	0.48		795.18	○	6.17	○	8.74	○
vowel	0.83	0.11	•	0.34		0.37		9.86	○	1.10		7346.63	○	16.39	○	54.35	○
waveform-5000	17.40	1.21	•	4.08	•	4.82	•	95.72	○	28.40	○	33108.75	○	110.53	○	191.49	○
zoo	0.27	0.05		0.16		0.12		1.00	○	0.54		618.34	○	3.59	○	6.52	○
Average	4.08	0.32		1.26		1.41		24.46		5.79		18230.51		86.86		181.61	
$W/T/L$	-	0/16/20		0/22/14		0/20/16		32/4/0		12/23/1		36/0/0		35/1/0		36/0/0	

TABLE 12
Elapsed training time summary of the Wilcoxon test.

Algorithm	CFW	NB	KLMFW	GRFW	DTFW	DFW	DEFW	CLLFW	MSEFW
CFW	-	○	○	○	●	●	●	●	●
NB	●	-	●	●	●	●	●	●	●
KLMFW	●	○	-		●	●	●	●	●
GRFW	●	○		-	●	●	●	●	●
DTFW	○	○	○	○	-	○	●	●	●
DFW	○	○	○	○	●	-	●	●	●
DEFW	○	○	○	○	○	○	-	○	○
CLLFW	○	○	○	○	○	○	●	-	●
MSEFW	○	○	○	○	○	○	●	○	-

REFERENCES

[1] J. Pearl, *Probabilistic Reasoning in Intelligent Systems*. San Francisco: Morgan Kaufmann, 1988.

[2] T. M. Mitchell, *Machine Learning*, 1st ed. New York: McGraw-Hill, 1997.

[3] N. Friedman, D. Geiger, and M. Goldszmidt, "Bayesian network classifiers," *Machine learning*, vol. 29(2-3), pp. 131–163, 1997.

[4] I. Kononenko, "Comparison of inductive and naive bayesian learning approaches to automatic knowledge acquisition," in *Current Trends in Knowledge Acquisition*, B. Wielinga, Ed. IOS Press, 1990.

[5] P. Langley, W. Iba, and K. Thomas, "An analysis of bayesian classifiers," in *Proceedings of the Tenth National Conference of Artificial Intelligence*. AAAI Press, 1992, pp. 223–228.

[6] P. Domingos and M. Pazzani, "Beyond independence: Conditions for the optimality of the simple bayesian classifier," *Machine Learning*, vol. 29, pp. 103–130, 1997.

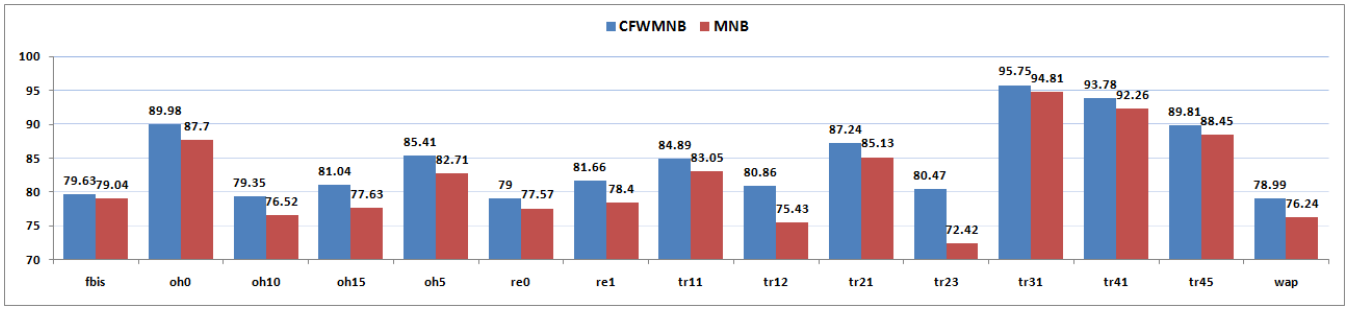
[7] X. Wu, V. Kumar, J. R. Quinlan, J. Ghosh, and Q. Yang, "Top 10 algorithms in data mining," *Knowledge and Information Systems*, vol. 14(1), pp. 1–37, 2008.

[8] E. Keogh and M. Pazzani, "Learning augmented bayesian classifiers: A comparison of distribution-based and classification-based approaches," in *Proceedings of the seventh international workshop on artificial intelligence and statistics*, 1999, pp. 225–230.

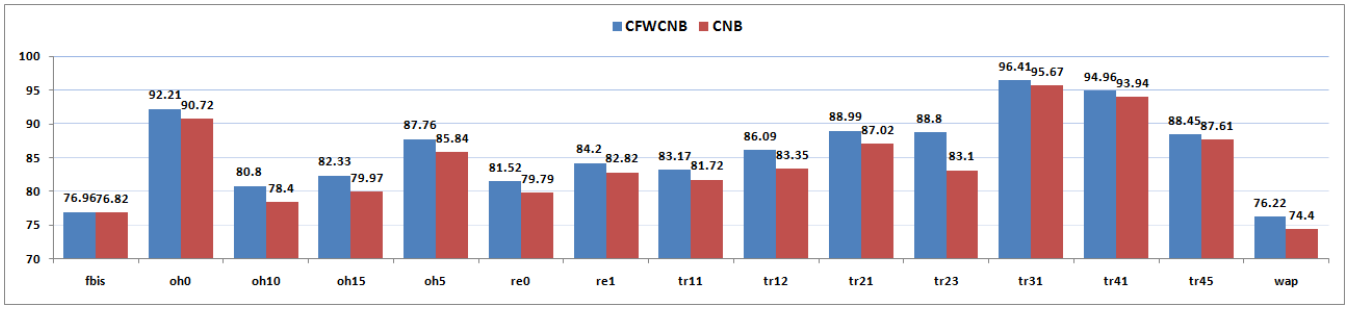
[9] L. Jiang, H. Zhang, and Z. Cai, "A novel bayes model: Hidden naive bayes," *IEEE Transactions on Knowledge and Data Engineering*, vol. 21, pp. 1361–1371, 2009.

[10] N. Li, Y. Yu, and Z. H. Zhou, "Semi-naive exploitation of one-dependence estimators," in *Proceedings of the 9th IEEE International Conference on Data Mining*. Miami, FL: IEEE, 2009, pp. 278–287.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2018.2836440, IEEE Transactions on Knowledge and Data Engineering



(a) CFWMNB versus MNB



(b) CFWCNB versus CNB

Fig. 2. Classification accuracy comparisons on 15 text datasets.

TABLE 13
15 text datasets used in our experiments.

Dataset	Documents number	Words number	Classes number
fbis	2463	2000	17
oh0	1003	3182	10
oh10	1050	3238	10
oh15	913	3100	10
oh5	918	3012	10
re0	1657	3758	25
re1	1504	2886	13
tr11	414	6429	9
tr12	313	5804	8
tr21	336	7902	6
tr23	204	5832	6
tr31	927	10128	7
tr41	878	7454	10
tr45	690	8261	10
wap	1560	8460	20

[11] C. Qiu, L. Jiang, and C. Li, "Not always simple classification: Learning superparent for class probability estimation," *Expert Systems with Applications*, vol. 42(13), pp. 5433–5440, 2015.
[12] J. Wu, S. Pan, X. Zhu, P. Zhang, and C. Zhang, "SODE: Self-adaptive one-dependence estimators for classification," *Pattern Recognition*, vol. 51, pp. 358–377, 2016.
[13] P. Langley and S. Sage, "Induction of selective bayesian classifiers," in *Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence*, 1994, pp. 339–406.
[14] C. A. Ratanamahatana and D. Gunopulos, "Feature selection for the naive bayesian classifier using decision trees," *Applied Artificial Intelligence*, vol. 17, pp. 475–487, 2003.
[15] L. Jiang, H. Zhang, Z. Cai, and J. Su, "Evolutional naive bayes," in *Proceedings of the 1st International Symposium on Intelligent Computation and its Applications*, 2005, pp. 344–350.
[16] H. Peng, F. Long, and C. Ding, "Feature selection based on mutual information: Criteria of max-dependency, max-relevance, and min-redundancy," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27(8), pp. 1226–1238, 2005.
[17] L. Jiang, Z. Cai, H. Zhang, and D. Wang, "Not so greedy: Randomly selected naive bayes," *Expert Systems with Applications*, vol. 39(12), pp. 11 022–11 028, 2012.
[18] B. Tang, S. Kay, and H. He, "Toward optimal feature selection in naive bayes for text categorization," *IEEE Transactions on Knowledge and Data Engineering*, vol. 28(9), pp. 2508–2521, 2016.
[19] H. Zhang and S. Sheng, "Learning weighted naive bayes with accurate ranking," in *Proceedings of the 4th International Conference on Data Mining*. Brighton, UK: IEEE, 2004, pp. 567–570.
[20] M. Hall, "A decision tree-based attribute weighting filter for naive bayes," *Knowledge-Based Systems*, vol. 20(2), pp. 120–126, 2007.
[21] C. H. Lee, F. Gutierrez, and D. Dou, "Calculating feature weights in naive bayes with kullback-leibler measure," in *Proceedings of the 11th IEEE International Conference on Data Mining*. Vancouver, BC: IEEE, 2011, pp. 1146–1151.
[22] J. Wu and Z. Cai, "Attribute weighting via differential evolution algorithm for attribute weighted naive bayes (wnb)," *Journal of Computational Information Systems*, vol. 7(5), pp. 1672–1679, 2011.
[23] N. A. Zaidi, J. Cerquides, M. J. Carman, and G. I. Webb, "Alleviating naive bayes attribute independence assumption by attribute weighting," *Journal of Machine Learning Research*, vol. 14, pp. 1947–1988, 2013.
[24] L. Jiang, C. Li, S. Wang, and L. Zhang, "Deep feature weighting for naive bayes and its application to text classification," *Engineering Applications of Artificial Intelligence*, vol. 52, pp. 26–39, 2016.
[25] R. Kohavi, "Scaling up the accuracy of naive-bayes classifier: A decision-tree hybrid," in *Proceedings of the second international conference on Knowledge Discovery and Data Mining*. AAAI Press, 1996, pp. 202–207.
[26] Z. Xie, W. Hsu, Z. Liu, and M. Lee, "A selective neighborhood based naive bayes for lazy learning," in *Proceedings of the Sixth Pacific Asia Conference on KDD*. Berlin: Springer, 2002, pp. 104–114.
[27] E. Frank, M. Hall, and B. Pfahringer, "Locally weighted naive bayes," in *Proceedings of the Nineteenth Conference on Uncertainty in Artificial Intelligence*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2003, pp. 249–256.
[28] C. Elkan, "Boosting and Naive Bayesian learning," University of California, San Diego, Tech. Rep. CS97-557, 1997.
[29] L. Jiang, Z. Cai, and D. Wang, "Improving naive bayes for classification," *International Journal of Computers and Applications*, vol. 32(3), pp. 328–332, 2010.
[30] L. Jiang, D. Wang, and Z. Cai, "Discriminatively weighted naive

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TKDE.2018.2836440, IEEE Transactions on Knowledge and Data Engineering

bayes and its application in text classification," *International Journal on Artificial Intelligence Tools*, vol. 21(1), p. 1250007, 2012.

[31] K. E. Hindi, "Fine tuning the naive bayesian learning algorithm," *AI Communications*, vol. 27(2), pp. 133–141, 2014.

[32] A. Alhussan and K. E. Hindi, "Selectively fine-tuning bayesian network learning algorithm," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 30(8), p. 1651005, 2016.

[33] D. M. Diab and K. E. Hindi, "Using differential evolution for fine tuning naive bayesian classifiers and its application for text classification," *Applied Soft Computing*, vol. 54, pp. 183–199, 2016.

[34] J. T. A. S. Ferreira, D. G. T. Denison, and D. J. Hand, "Weighted naive bayes modelling for data mining," In Dept. of Mathematics, Imperial College, London, UK, Tech. Rep., 2001.

[35] M. Hall, "Correlation-based feature selection for discrete and numeric class machine learning," in *Proceedings of the 17th International Conference on Machine Learning*, Stanford, CA, USA, 2000, pp. 359–366.

[36] A. McCallum and K. A. Nigam, "A comparison of event models for naive bayes text classification," in *Working Notes of the 1998 AAAI/ICML Workshop on Learning for Text*. Madison, Wisconsin, USA: AAAI Press, 1998, pp. 41–48.

[37] U. M. Fayyad and K. B. Irani, "Multi-interval discretization of continuous-valued attributes for classification learning," in *Proceedings of the 13th International Joint Conference on Articial Intelligence*, Chambery, France, 1993, pp. 1022–1027.

[38] I. H. Witten, E. Frank, and M. A. Hall, *Data Mining: Practical Machine Learning Tools and Techniques*. San Francisco, California, USA: Morgan Kaufmann, 2011.

[39] S. B. Thrun, J. Bala, E. Bloedorn, I. Bratko, B. Cestnik, J. Cheng, K. De Jong, S. Dzeroski, S. E. Fahlman, D. Fisher, R. Hamann, K. Kaufman, S. Keller, I. Kononenko, J. Kreuziger, R. S. Michalski, T. Mitchell, P. Pachowicz, Y. Reich, H. Vafaie, W. Van de Welde, W. Wenzel, J. Wnek, and J. Zhang, "The monk's problems: A performance comparison of different learning algorithms," Carnegie Mellon University, Tech. Rep. Technical Report CS-CMU-91-197, 1991.

[40] M. A. Hall, "Correlation-based feature selection for machine learning," The University of Waikato, Tech. Rep., 1999.

[41] L. Jiang, Z. Cai, H. Zhang, and D. Wang, "Naive bayes text classifiers: a locally weighted learning approach," *Journal of Experimental & Theoretical Artificial Intelligence*, vol. 25(2), pp. 273–286, 2013.

[42] L. Zhang, L. Jiang, C. Li, and G. Kong, "Two feature weighting approaches for naive bayes text classifiers," *Knowledge-Based Systems*, vol. 100, pp. 137–144, 2016.

[43] S. Wang, L. Jiang, and C. Li, "Adapting naive bayes tree for text classification," *Knowledge and Information Systems*, vol. 44(1), pp. 77–89, 2015.

[44] J. D. Rennie, L. Shih, J. Teevan, and D. R. Karger, "Tackling the poor assumptions of naive bayes text classifiers," in *Proceedings of the Twentieth International Conference on Machine Learning*. Morgan Kaufmann, 2003, pp. 616–623.

[45] C. Nadeau and Y. Bengio, "Inference for the generalizaiton error," *Machine Learning*, vol. 52(3), pp. 239–281, 2003.

[46] J. Su and H. Zhang, "A fast decision tree learning algorithm," in *Proceedings of the Twenty-First National Conference on Artificial Intelligence*. AAAI Press, 2006, pp. 500–505.

[47] J. Alcalá-Fdez, A. Fernandez, J. Luengo, J. Derrac, S. García, L. Sánchez, and F. Herrera, "KEEL data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework," *Journal of Multiple-Valued Logic and Soft Computing*, vol. 17(2-3), pp. 255–287, 2011.

[48] J. Demsar, "Statistical comparisons of classifiers over multiple data sets," *Journal of Machine Learning Research*, vol. 7, pp. 1–30, 2006.

[49] S. Garcia and F. Herrera, "An extension on statistical comparisons of classifiers over multiple data sets for all pairwise comparisons," *Journal of Machine Learning Research*, vol. 9, pp. 2677–2694, 2008.

[50] D. Grossman and P. Domingos, "Learning bayesian network classifiers by maximizing conditional likelihood," in *Proceedings of the Twenty-First International Conference on Machine Learning*. ACM Press, 2004, pp. 361–368.

[51] Y. Guo and R. Greiner, "Discriminative model selection for belief net structures," in *Proceedings of the Twentieth National Conference on Artificial Intelligence*. AAAI Press, 2005, pp. 770–776.

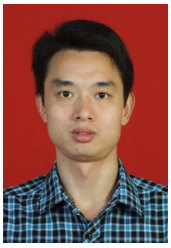
[52] L. Jiang, Z. Cai, and D. Wang, "Improving tree augmented naive bayes for class probability estimation," *Knowledge-Based Systems*, vol. 26, pp. 239–245, 2012.

[53] P. Bradley and P. Andrew, "The use of the area under the roc curve in the evaluation of machine learning algorithms," *Pattern Recognition*, vol. 30(7), pp. 1145–1159, 1997.

[54] D. Hand and R. Till, "A simple generalisation of the area under the roc curve for multiple class classification problems," *Machine Learning*, vol. 45(2), pp. 171–186, 2001.

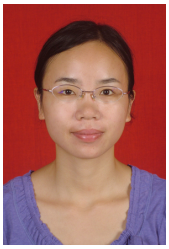
[55] L. Jiang, "Random one-dependence estimators," *Pattern Recognition Letters*, vol. 32(3), pp. 532–539, 2011.

[56] C. X. Ling, J. Huang, and H. Zhang, "AUC: a statistically consistent and more discriminating measure than accuracy," in *Proceedings of the 18th International Joint Conference on Artificial Intelligence*. Morgan Kaufmann, 2003, pp. 329–341.



Liangxiao Jiang received his PhD degree from China University of Geosciences, Wuhan, China, in 2009. He is currently a professor with the Department of Computer Science, China University of Geosciences (Wuhan). His current research interests include data mining and machine learning. Since 2005, he has published over 60 refereed journal and conference papers in the above areas.

Lungan Zhang received his B.Sc. degree from China University of Geosciences in June 2015. Currently, he is a M.Sc. student at the Department of Computer Science, China University of Geosciences. His research interests include data mining and machine learning. Since 2016, he has published over 10 refereed journal and conference papers in the above areas.



Chaoqun Li received her PhD degree from China University of Geosciences, Wuhan, China, in 2012. She is currently an associate professor with the Department of Mathematics, China University of Geosciences (Wuhan). Her current research interests include data mining and machine learning. Since 2006, he has published over 30 refereed journal and conference papers in the above areas.

Jia Wu received his PhD degree in computer science from the University of Technology Sydney, Sydney, NSW, Australia. He is currently a lecturer with the Department of Computing, Macquarie University. His current research interests include data mining and machine learning. Since 2009, he has published over 40 refereed journal and conference papers in the above areas.