Inference-Based Naïve Bayes: Turning Naïve Bayes Cost-Sensitive

Xiao Fang

Abstract—A fundamental challenge for developing a cost-sensitive Naïve Bayes method is how to effectively classify an instance based on the cost-sensitive threshold computed under the assumption of knowing the instance's true classification probabilities and the highly biased estimations of these probabilities by the Naïve Bayes method. To address this challenge, we develop a cost-sensitive Naïve Bayes method from a novel perspective of inferring the order relation (e.g., greater than or equal to, less than) between an instance's true classification probability of belonging to the class of interest and the cost-sensitive threshold. Our method learns and infers the order relation from the training data and classifies the instance based on the inferred order relation. We empirically show that our proposed method significantly outperforms major existing methods for turning Naïve Bayes cost-sensitive through experiments with UCI data sets and a real-world case study.

Index Terms—Cost-sensitive classification, Naïve Bayes, classification

1 Introduction

Classification has significant impact on a wide variety of important applications from marketing [27], [28] to consumer credit scoring [21], and medical diagnosis [42]. Given its importance, a number of classification methods have been developed, including decision tree induction methods [2], [34], neural networks [30], support vector machines [40], and Bayesian classification methods [9]. Among classification methods, the Naïve Bayes method has many attractive properties such as simplicity, low time, and memory requirements [6]. Despite its simplicity, the Naïve Bayes method is comparable or even favorable to more complicated classification methods in terms of the classification accuracy [6], [14]. The method thus remains popular for classification tasks over years [24], [35]. The Naïve Bayes method, like other classification methods, aims at maximizing the classification accuracy, assuming that all misclassifications are equally costly [43]. Nonetheless, the assumption is not true for many real-world applications. Considering a direct marketing application, the cost of not marketing to a potential buyer is normally much higher than the cost of marketing to a nonbuyer. It is therefore necessary to develop a cost-sensitive Naïve Bayes method with the objective of minimizing the misclassification cost.

Threshold adjustment is generally a promising way of transforming a classification method into cost-sensitive one [10]. It adjusts the classification threshold used by an accuracy-maximization classification method to a cost-sensitive threshold to turn the method cost-sensitive. For two-class classification, an optimal cost-sensitive threshold that minimizes the expected misclassification cost can be calculated using the Bayes decision rule, assuming that true classification probability of an instance belonging to each class is known [4], [9], [14]. Ideally, an instance is optimally classified as one class if its true probability of belonging to the class is greater than or equal to the optimal cost-sensitive threshold; otherwise it is optimally classified as the other class [10], [14]. However, true classification probabilities are unknown in reality and they are substituted with their estimations when classifying an instance [14]. Moreover, classification probabilities estimated by the Naïve Bayes method are highly biased [14]. That is, probability estimation errors by the Naïve Bayes method (i.e., absolute differences between true and estimated classification probabilities) are large. Therefore, a fundamental challenge for developing a cost-sensitive Naïve Bayes method is how to effectively classify an instance based on the cost-sensitive threshold computed under the assumption of knowing the instance's true classification probabilities and the highly biased estimations of these probabilities by the Naïve Bayes method.

To address this challenge, a straightforward solution is to reduce probability estimation errors by the Naïve Bayes method. However, reducing probability estimation errors do not necessarily lead to better classification performance and often can make it worse because of different bias-variance decomposition between probability estimation errors and classification errors [14]. We observe that an instance can be optimally classified as long as we know the order relation (e.g., greater than or equal to, less than) between its true classification probability of belonging to the class of interest and the cost-sensitive threshold, while knowing this true classification probability is not necessary. Hence, effective cost-sensitive classifications could be realized by inferring the order relation from the training data, rather than by reducing probability estimation errors. We develop a cost-sensitive Naïve Bayes method from this novel perspective. Our method learns and infers the order relation between an

• *The author is with the Department of Operations and Information Systems, David Eccles School of Business, University of Utah, 1655 East Campus Center Drive, Salt Lake City, UT 84112-9301. E-mail: xiao.fang@business.utah.edu.*

Manuscript received 12 June 2011; revised 21 Nov. 2011; accepted 30 Aug. 2012; published online 3 Oct. 2012.
Recommended for acceptance by G. Karypis.
For information on obtaining reprints of this article, please send e-mail to: tkde@computer.org, and reference IEEECS Log Number TKDE-2011-06-0347.
Digital Object Identifier no. 10.1109/TKDE.2012.196.

instance's true classification probability of belonging to the class of interest and the cost-sensitive threshold from the training data and classifies the instance based on the inferred order relation. In particular, our method infers the order relation based on the oversmoothing nature of the Naïve Bayes method [14], according to which classification probabilities estimated by the Naïve Bayes method tend to be shrunk toward the mean output value of the training data [14]. This study makes the following contributions. First, we propose how to infer the order relation from relevant order relations learned from the training data. Second, we develop a cost-sensitive Naïve Bayes method based on the inference of the order relation. Third, we empirically show that our proposed cost-sensitive Naïve Bayes method significantly outperforms major existing methods for turning Naïve Bayes cost-sensitive.

The rest of the paper is organized as follows. We review related work in Section 2. We propose a cost-sensitive Naïve Bayes method in Section 3. The performance of the proposed method is empirically compared to that of major existing methods for turning Naïve Bayes cost-sensitive in Section 4. A case study of using our proposed method in a direct marketing scenario is presented in Section 5. The paper is concluded with future research directions in Section 6.

2 Related Work

Prior research related to this study can be broadly categorized as distribution adjustment and threshold adjustment methods for turning a classification method cost-sensitive. Distribution adjustment methods alter prior class probabilities in the training data according to given costs of misclassifying an instance. Threshold adjustment methods, on the other hand, use the as-is training data but change the classification threshold to reflect given misclassification costs. In the following, we review major methods in these two categories.

Breiman et al. [2, pp. 114-115] define altered prior class probability as a function of its current probability and misclassification costs such that instances with higher misclassification cost are less likely to be misclassified. Altering prior class probabilities is usually realized through stratification, i.e., undersampling or oversampling, where undersampling removes instances from the training data while oversampling adds instances to the training data. In particular, random undersampling randomly eliminates instances in accordance with altered prior class probabilities; and intelligent undersampling selects certain instances to remove. For example, one-sided selection [26], an intelligent undersampling method, identifies borderline, noisy, or redundant instances and then undersamples the training data by removing these identified instances. Similarly, random oversampling randomly duplicates instances according to altered prior class probabilities while intelligent oversampling employs heuristics to add instances to the training data. For example, SMOTE [5], an intelligent oversampling method, generates and adds synthetic instances to the training data.

Altering prior class probabilities can also be achieved through instance weighting. Specifically, instances in the training data are weighted in proportion to the costs of misclassifying them, which implicitly changes prior class probabilities in the training data. Ting [39] proposes an instance weighting method for inducing cost-sensitive decision trees. By assigning high weights to instances with high misclassification costs, the proposed method influences decision tree induction to focus on these instances, which in turn reduces the number of high-cost errors and the total misclassification cost. The instance weighting schema proposed in [39] is generic and applicable to the Naïve Bayes method. Gama [17] develops an iterative Naïve Bayes method, which iteratively updates class-attribute distributions learned by the Naïve Bayes method with a predefined increment. As a result, instance weights in the training data are changed. By incorporating misclassification costs into the increment, the method can be employed to turn the Naïve Bayes method cost-sensitive [17]. MetaCost [7] is another distribution adjustment method. Different from the methods just reviewed, MetaCost alters prior class probabilities in the training data by relabeling instances rather than adding (or removing) instances or weighting instances differently. MetaCost assumes that, if instances in the training data were relabeled to their optimal classes according to given misclassification costs, an accuracy-maximization classification method applied to the modified training data could produce an optimal cost-sensitive classifier [7]. Based on the assumption, MetaCost computes the optimal class for each instance in the training data and relabels the instance if its current class is different from the computed optimal class.

Distribution adjustment methods have their own limitations. For example, oversampling increases the learning time and the risk of overfitting whereas undersampling reduces useful information for learning [7]. Ting [38] shows that MetaCost is inferior to its internal classifier for turning a classification method cost-sensitive and questions the effectiveness of the method. Moreover, simply altering prior class probabilities in the training data have limited effect on classifiers learned by the Naïve Bayes method [10]. Therefore, distribution adjustment methods seem less promising in turning the Naïve Bayes method cost-sensitive than threshold adjustment methods [10].

Prior studies have shown that threshold adjustment methods are more effective in turning classification methods cost-sensitive than stratification methods [33], [46]. Threshold adjustment methods consider misclassification costs and move the classification threshold used by an accuracy-maximization classification method to a cost-sensitive classification threshold. For example, Sheng and Ling [36] develop a method that searches for the best threshold minimizing the total misclassification cost in the training data and uses the threshold to classify instances in the test data. The method proposed by Zadrozny and Elkan [43] considers instance-dependent misclassification costs, instead of commonly used class-dependent misclassification costs. Margineantu [31] develops a method to compute confidence estimates for classification probabilities and the method classifies instances based on confidence estimates. Zhou and Liu [46] propose a threshold adjustment method for training cost-sensitive neural networks and show that the proposed method outperforms stratification methods in terms of the misclassification cost and the number of high cost errors.

TABLE 1
Feature Comparison: Our Method versus Prior Methods

Feature		Distribution Adjustment Methods (e.g. [5], [7], [26])	Threshold Adjustment + Modified Naïve Bayes (e.g. [16], [25])	Our Method
training data	as-is		√	√
	distribution adjusted	√		
threshold	accuracy-maximization threshold	√		
	cost-sensitive threshold		√	√
learning classification probability with	standard Naïve Bayes	√		√
	modified Naïve Bayes		√	
inferring the order relation	Yes			√
	No	√	√	

While threshold adjustment is in general promising for turning a classification method cost-sensitive, applying it to the Naïve Bayes method faces a fundamental challenge discussed in Section 1: how to effectively classify an instance based on the cost-sensitive threshold and the highly biased estimations of its classification probabilities by the Naïve Bayes method. One possible solution is to combine threshold adjustment with a modified Naïve Bayes method that can reduce probability estimation errors. In this vein, a number of methods have been proposed to reduce probability estimation errors by relaxing the Naïve Bayes's conditional independence assumption to some extent. Friedman et al. [15] propose the tree augmented Naïve Bayes method, which relaxes the assumption by allowing one attribute to be dependent on at most one other attribute (i.e., one-dependence). Similar to TAN, aggregating one-dependence estimators (AODE) enhances the Naïve Bayes method by allowing one-dependence and the method aggregates the estimations of all possible one-dependences [49], [50]. Webb et al. [51] extend AODE to AnDE, which allows one attribute to be dependent on $n \geq 1$ other attributes. The hidden Naïve Bayes method [25] is also a one-dependence method, which allows one attribute to be dependent on one hidden attribute that combines influences from all other attributes. Relaxing the conditional independence assumption has also been achieved by weighting attributes differently [52], [53] or by weighting instances in the training data differently [16]. Rather than reducing probability estimation errors, our method infers the order relation between an instance's true classification probability of belonging to the class of interest and the cost-sensitive threshold and classifies the instance based on the inferred order relation. Next section discusses our method in details. As the summary of this section, Table 1 compares between our method and major prior methods along a set of key features, including training data used, threshold used, method for learning classification probability, and whether to infer the order relation.

3 Inference-based Naïve Bayes

For two-class classification, let T be the training data of classified instances, where each classified instance is described by a vector of its attributes $X = \langle X_1, X_2, \ldots, X_n \rangle$ and its class label $Y \in \{0, 1\}$. For brevity, we refer to instance x as an instance with attribute value vector $x = \langle x_1, x_2, \ldots, x_n \rangle$; i.e., $X_l = x_l$ for $l = 1, 2, \ldots, n$. Let C be the matrix of misclassification costs, where its element $C(i, j)$ denotes the cost of classifying an instance as class i while the instance actually belongs to class j, $i, j \in \{0, 1\}$. Given T and C, the objective is to learn a cost-sensitive classifier from T to minimize the cost of misclassifying unclassified instances.

Let $p(x)$ be the true classification probability of an unclassified instance x belonging to class 1; that is,

$$p(x) = P(Y = 1|x). \tag{1}$$

By the Bayes decision rule [9], the expected misclassification cost $EC(i|x)$ of classifying instance x as class $i \in \{0, 1\}$ is given by

$$\begin{aligned} EC(i|x) &= P(Y = 0|x)C(i, 0) + P(Y = 1|x)C(i, 1) \\ &= (1 - p(x))C(i, 0) + p(x)C(i, 1). \end{aligned} \tag{2}$$

It is optimal to classify instance x as class 1 if $EC(1|x) \leq EC(0|x)$ or class 0 otherwise [9], [10]. Solving the inequality with (2), the optimal classification Y^* for instance x is

$$Y^* = \begin{cases} 1 & \text{if } p(x) \geq p^*, \\ 0 & \text{if } p(x) < p^*, \end{cases} \tag{3}$$

where the cost-sensitive threshold p^* is given by [10]

$$p^* = \frac{C(1, 0) - C(0, 0)}{C(1, 0) - C(1, 1) + C(0, 1) - C(0, 0)}. \tag{4}$$

Both equations (4) and (3) are derived under the assumption that true classification probability $p(x)$ is known [10]. In reality, $p(x)$ is generally unknown and it is substituted with its estimation $\hat{p}(x)$ learned from the training data T [14]. Noises in the training data as well as limitations of learning methods might cause $\hat{p}(x)$ deviating from $p(x)$. Such deviation (or estimation error) in turn could cause the classification based on $\hat{p}(x)$ and the cost-sensitive threshold p^* different from the optimal classification [14].

Applying the Naïve Bayes method to the training data T, we can compute $\hat{p}(x)$ as following [32]:

$$\hat{p}(x) = \frac{\pi_1 \prod_{l=1}^{n} \hat{P}(X_l = x_l|Y = 1)}{\pi_0 \prod_{l=1}^{n} \hat{P}(X_l = x_l|Y = 0) + \pi_1 \prod_{l=1}^{n} \hat{P}(X_l = x_l|Y = 1)}. \tag{5}$$

In (5),$\pi_i = \hat{P}(Y = i), i \in \{0, 1\}$ is the estimated prior probability of class i, which can be calculated as the frequency of class i instances in T [32]. $\hat{P}(X_l = x_l|Y = i)$ represents the estimated conditional probability of $X_l = x_l$ given $Y = i$. It is computed as the frequency of attribute X_l taking value x_l in class i instances for discrete-valued X_l or by assuming a Gaussian distribution for X_l for continuous-valued X_l [32]. In the experiments reported in Section 4, we follow this standard procedure to estimate $\hat{P}(X_l = x_l|Y = i)$.

It is well known that $\hat{p}(x)$ computed using the Naïve Bayes method is a highly biased estimation of $p(x)$ [14],

primarily due to the method's unrealistic assumption that X_l are conditionally independent given Y, $l = 1, 2, \ldots, n$ [14]. Thus, a fundamental challenge for developing a cost-sensitive Naïve Bayes method is how to effectively classify an instance x based on p^* and highly biased $\hat{p}(x)$ estimated by the Naïve Bayes method. One possible solution is to enhance the accuracy of $\hat{p}(x)$, i.e., reducing probability estimation errors. A number of methods have been developed to reduce probability estimation errors of Naïve Bayes by relaxing the conditional independence assumption to some extent [15], [16], [25], [41]. We may employ one of these methods to produce more accurate estimation of $p(x)$, which in turn is compared to p^* for cost-sensitive classification according to (3). However, reducing probability estimation errors do not necessarily lead to better classification performance and often can make it worse because the bias-variance decomposition determining probability estimation errors is different from that determining classification errors [14]. Moreover, by equation (3), optimal classification of an instance depends on both its classification probability and the cost-sensitive threshold. Therefore, solely reducing probability estimation errors is not adequate for effective classification.

Classifying an instance x using (3) does not require the exact value of $p(x)$ but the order relation (e.g., greater than or equal to, less than) between $p(x)$ and p^*. Hence, rather than improving the accuracy of estimating $p(x)$, it could be more effective for cost-sensitive classifications to infer the order relation between $p(x)$ and p^* from relevant order relations learned with the training data such as that between $\hat{p}(x)$ and p^*. We take this perspective to develop a cost-sensitive Naïve Bayes method and name it inference-based Naïve Bayes because it classifies instances based on the inferred order relation between $p(x)$ and p^*. Our method employs the oversmoothing nature of the Naïve Bayes method [14]. According to this nature [14], the probability $\hat{p}(x)$ estimated by the Naïve Bayes method tends to be shrunk toward the mean output value $\bar{Y}$, where

$$\bar{Y} = \frac{1}{N}\sum_{i=1}^{N} Y_i,$$

N is the number of instances in the training data, and $Y_i \in \{0, 1\}$ is the class label of instance i in the training data. And, $\hat{p}(x)$ can be modeled as [14]

$$\hat{p}(x) = (1 - \alpha(x))p(x) + \alpha(x)\bar{Y}, \tag{6}$$

where $\alpha(x)$ denotes the oversmoothing coefficient, which is instance dependent and $0 < \alpha(x) < 1$.[1] By (6), if $\bar{Y} = p^*$, $\hat{p}(x) \geq p^*$ implies $p(x) \geq p^*$ and $\hat{p}(x) < p^*$ suggests $p(x) < p^*$. Hence, an instance can be effectively classified by substituting $p(x)$ in (3) with its Naïve Bayes estimation $\hat{p}(x)$ when $\bar{Y} = p^*$.[2] However, this strong condition seldom holds in reality. It is therefore necessary to analyze situations when $\bar{Y} \neq p^*$.

Lemma 1. *Given* $\bar{Y} > p^*$,

- *if* $\hat{p}(x) \leq p^*$, *we have* $p(x) < p^*$;
- *if* $\hat{p}(x) \geq \bar{Y}$, *we have* $p(x) > p^*$.

Proof. Let us first prove (a). By (6), $\hat{p}(x) \leq p^*$ means

$$(1 - \alpha(x))p(x) + \alpha(x)\bar{Y} \leq p^*.$$

Given $\bar{Y} > p^*$ and $\alpha(x) > 0$, we have

$$(1 - \alpha(x))p(x) + \alpha(x)p^* < p^*.$$

Considering $0 < \alpha(x) < 1$, we have $p(x) < p^*$. We then prove (b). By (6), $\hat{p}(x) \geq \bar{Y}$ indicates

$$(1 - \alpha(x))p(x) + \alpha(x)\bar{Y} \geq \bar{Y}.$$

The above inequality implies $p(x) \geq \bar{Y}$ because $0 < \alpha(x) < 1$. Given $\bar{Y} > p^*$, we have $p(x) > p^*$. □

For the situation of $\bar{Y} > p^*$, by Lemma 1 and (3), instance x is labeled as class 0, if its Naïve Bayes estimated probability $\hat{p}(x)$ is less than or equal to the cost-sensitive threshold p^*. On the other hand, if $\hat{p}(x)$ is greater than or equal to $\bar{Y}$, instance x is classified as class 1. However, for an instance with $\hat{p}(x)$ falling between p^* and $\bar{Y}$, its class cannot be determined by applying Lemma 1 directly.

Lemma 2. *Given* $\bar{Y} < p^*$,

- *if* $\hat{p}(x) \leq \bar{Y}$, *we have* $p(x) < p^*$;
- *if* $\hat{p}(x) \geq p^*$, *we have* $p(x) > p^*$.

Proof. We first prove (a). By (6), $\hat{p}(x) \leq \bar{Y}$ indicates

$$(1 - \alpha(x))p(x) + \alpha(x)\bar{Y} \leq \bar{Y}.$$

The above inequality implies $p(x) \leq \bar{Y}$ because $0 < \alpha(x) < 1$. Given $\bar{Y} < p^*$, we have $p(x) < p^*$. We prove (b) next. By (6), $\hat{p}(x) \geq p^*$ means

$$(1 - \alpha(x))p(x) + \alpha(x)\bar{Y} \geq p^*.$$

Given $\bar{Y} < p^*$ and $\alpha(x) > 0$, we have

$$(1 - \alpha(x))p(x) + \alpha(x)p^* > p^*.$$

Hence, $p(x) > p^*$ because $0 < \alpha(x) < 1$. □

For the situation of $\bar{Y} < p^*$, by Lemma 2 and (3), it is effective to predict instance x as class 0, if its Naïve Bayes estimated $\hat{p}(x)$ probability is less than or equal to $\bar{Y}$. If $\hat{p}(x)$ is greater than or equal to the cost-sensitive threshold p^*, it is effective to predict instance x as class 1. However, for an instance with $\hat{p}(x)$ falling between $\bar{Y}$ and p^*, its class cannot be determined by applying Lemma 2 directly.

Let U be the set of unclassified instances whose class cannot be determined by directly applying Lemma 1 or 2. Obviously, U consists of instances with $\hat{p}(x)$ falling between $\bar{Y}$ and p^*. Their class labels need to be further learned with appropriate training data. We construct training data for U using nearest neighbors of each instance in U. Nearest neighbors of an instance are highly informative for classifying the instance. According to [4], half classification

1. $\alpha(x) = 0$ means that $\hat{p}(x)$ is the perfect estimation of $p(x)$, which is unlikely for the Naïve Bayes method. Thus, $\alpha(x) = 0$ is not considered in this study. On the other hand, $\hat{p}(x)$ is an estimation of $p(x)$, although highly biased. Hence, $\alpha(x) = 1$ is not considered either.

2. When $\bar{Y} \neq p^*$, to effectively classify an instance by simply substituting $p(x)$ in (3) with its Naïve Bayes estimation, a seemingly viable way is to adjust prior class probabilities in the training data. However, such way is not practical because appropriate level of adjustment is unknown and inappropriate adjustment could result in worse performance than no adjustment [14].

INB (p^*, L, T_L , k)

p^*: cost-sensitive threshold
L: set of unclassified instances
T_L : training data for classifying instances in L; $T_L = T$ when INB firstly invoked
k: number of nearest neighbors

$U = \phi$.
Estimate $\hat{P}(Y = i), i \in \{0,1\}$, $\hat{P}(X_l = x_l \mid Y = i)$, l=1,2,…,n, and calculate $\overline{Y}$ from T_L .
For each unclassified instance x in L
  Calculate $\hat{p}(x)$ using (5).
  If ($\overline{Y} = p^*$ and $\hat{p}(x) < p^*$) or ($\overline{Y} > p^*$ and $\hat{p}(x) \le p^*$) or ($\overline{Y} < p^*$ and $\hat{p}(x) \le \overline{Y}$)
    Classify instance x as class 0. //By Lemmas 1,2 and equation (3)
  Else if ($\overline{Y} = p^*$ and $\hat{p}(x) \ge p^*$) or ($\overline{Y} > p^*$ and $\hat{p}(x) \ge \overline{Y}$) or ($\overline{Y} < p^*$ and $\hat{p}(x) \ge p^*$)
    Classify instance x as class 1. //By Lemmas 1,2 and equation (3)
  Else
    $U = U \cup \{x\}$. //Add x to U
  End if
End for
If ($U = \phi$)
  Terminate.
Else if ($|U| = |L|$) //Hard instances
  Classify hard instances using (3).
  Terminate.
Else
  $L = U$.
  T_L = NN(T,U, k). //Construct training data for U
  INB (p^*, L, T_L , k).
End if

Fig. 1. The inference-based Naïve Bayes algorithm.

information of an instance is contained in its nearest neighbor. When identifying nearest neighbors, we follow commonly applied procedures to calculate the distance between instances. For example, the distance between instances with continuous-valued attributes are measured with the Euclidean distance, where the attributes are standardized before applying the Euclidean distance function [20]; and the distance between instances with discrete-valued attributes are gauged as the ratio of mismatched attributes [20], [24].

For each instance in U, its k nearest neighbors are k instances in the original training data with least distance to it, $k \ge 1$. The training data for U contain k nearest neighbors of each instance in U. Once the training data for U are constructed, we apply the Naïve Bayes method to the data to estimate probability $\hat{p}(x)$ for each instance in U as well as $\bar{Y}$ and employ the Lemmas to classify the instance. It is possible that there are still instances whose class cannot be determined. We thus repeat the above described procedure of constructing training data for these unclassified instances and then classifying them using the Lemmas until there are no unclassified instances left or there are remaining instances, called hard instances, that can never be classified using the Lemmas. Hard instances are finally classified according to their probabilities $\hat{p}(x)$ estimated during the last run of the procedure and (3) by substituting $p(x)$ in (3) with $\hat{p}(x)$. Empirical results reported in Section 4.2 show that few instances are left as hard instances.

Fig. 1 summarizes the inference-based Naïve Bayes (INB) algorithm. The inputs of the algorithm include the cost-sensitive threshold p^*, which is calculated from the cost matrix C using (4), the set of unclassified instances L, training data T_L for classifying instances in L, where T_L equals T when INB is first invoked, and the number of nearest neighbors k. The algorithm classifies instances according to the Lemmas and calls itself recursively until termination. The most computationally expensive component of INB is the procedure of constructing training data by finding nearest neighbors, i.e., procedure NN() in Fig. 2. For each instance in L whose class cannot be determined by applying Lemma 1 or 2, it is necessary to find its nearest neighbors. Hence, the more instances in L left undetermined after applying the Lemmas the more time is required to run INB. INB could be adapted to run in an online fashion, i.e., unclassified instances in L being presented to INB one-by-one. In this case, a presented instance is first attempted to be classified by the Lemmas. If its class label cannot be determined by the Lemmas, training data are constructed for the instance. Using the newly constructed training data, the instance's label is determined either by the Lemmas or by (3) (i.e., hard instance). One potential advantage of running INB in an online fashion is that the true label of the presented instance could be observed after its classification and the instance with its true label could be added to the original training data to classify the rest instances in L.

Choice of k is important to the performance of the proposed algorithm. A common approach to choosing k is to evaluate various k values by applying leave-one-out

```
NN (T, U, k)
    T: training data of classified instances
    U: set of unclassified instances
    k: number of nearest neighbors

T_U = φ.  // T_U : training data for U
For each instance in U
    Find its k nearest neighbors in T.
    For each identified nearest neighbor nn
        If ( nn ∉ T_U )  //Ensure no duplications in T_U
            T_U = T_U ∪ {nn}.
        End if
    End for
End for
Return T_U .
```

Fig. 2. Procedure NN().[3]

cross validation to the training data [18], [22]. In particular, for a k value, each instance in the training data is classified by its k nearest neighbors in the data; and the chosen k value is the best one according to some evaluation criteria such as misclassification rate [18], [22]. Another more direct and effective approach to finding k is to run INB using cross validation for different k values and selects k value with the lowest misclassification cost. Conceivably, such approaches are highly time consuming for large size training data. Unfortunately, it is quite common to have large size training data in real-world classification tasks. Hall et al. [19] discover a theoretical property characterizing optimal k. For two sample data sets from the same distribution, T_1 and T_2, T_1 on average has u class 0 instances and v class 1 instances; T_2 on average has ru class 0 instances and rv class 1 instances, $r > 0$. Let the optimal number of nearest neighbors for T_1 and T_2 are k_1 and k_2, respectively. According to the property [19], one has[3]

$$\frac{k_1}{k_2} = r^{-4/(n+4)}, \tag{7}$$

where n is the number of attributes describing an instance. The theoretical property depicted in (7) reveals the relationship between optimal k for two data sets of different sizes, assuming that the number of class 0 instances and the number of class 1 instances in each data set follow Poisson distribution, respectively [19]. Utilizing the property, we can develop a simple heuristic to find k: creating a small sample from a large size training data set, finding the optimal number of nearest neighbors for the sample using cross validation (which is not computational expensive because of its small size), and calculating the number of nearest neighbors for the training data set with (7). Specifically, we construct several equally sized small samples from the training data T, where the numbers of class 0 and class 1 instances in a sample are z times of their original numbers in T and $0 < z < 1$. For each sample, its optimal number k_s of nearest neighbors is found through cross validation. Let $\overline{k_s}$ be the average optimal number of nearest neighbors across samples. By (7), k for the training data T can be approximated as $\overline{k_s} \times z^{-4/(n+4)}$.

3. For implementation, nearest neighbors identified over the initial run of procedure NN() may be stored, which saves the computation time of reidentifying these nearest neighbors during the following runs of procedure NN().

TABLE 2
Experimental Data Sets

Data set	# of instances	# of attributes	Attribute type+
adult	45222	14	m
australian	690	14	m
blood	748	4	c
breast-c	286	9	d
breast-w (original)	699	9	d
diabetes	768	8	c
german	1000	20	m
haberman	306	3	c
horse	368	23	m
ionosphere	351	34	c
labor	57	16	m
liver	345	6	c
monk	432	6	d
pima	768	8	c
sick	3772	29	m
sonar	208	60	c
spambase	4601	57	c
spect	267	22	d
statlog-heart	270	13	m
wine-quality(red)	1599	11	c
libras	360	90	c
musk	476	166	c
semeion	1593	256	c

+*c—all continuous-valued attributes; d—all discrete-valued attributes; m—mixed.*

4 Experimental Study

We conducted experiments to empirically evaluate the performance of the inference-based Naïve Bayes method. This section describes the experimental setup and reports the experimental results.

4.1 Experimental Setup

The experiments employed 23 binary-class data sets from the UCI machine learning repository [12]. These data sets, as described in Table 2, represent a broad range in size and dimensionality. We benchmarked the inference-based Naïve Bayes method against 11 methods, including major methods for transforming Naïve Bayes cost-sensitive and the standard Naïve Bayes method as a baseline. Four of the benchmark methods are stratification methods: random undersampling, random oversampling, one-sided selection [26], and SMOTE [5], which have been popularly employed as benchmarks for evaluating the performance of cost-sensitive classification methods, e.g., [7], [8], [46]. When implementing stratification methods, we followed the guideline by Breiman et al. [2] to change prior class probabilities. Let $P(j)$ be the prior probability of class $j \in 0, 1$ in the training data. In consideration of misclassification costs, the altered prior probability $P'(j)$ of class j is given by [2, pp. 114-115]

$$P'(j) = \frac{C(j)P(j)}{\sum_j C(j)P(j)}, \quad (8)$$

where

$$C(j) = \sum_i C(i,j), i \in 0,1. \quad (9)$$

Another benchmark method is instance weighting. Following the weighting schema in [39], the weight $w(j)$ of a class $j \in \{0,1\}$ instance is calculated as

$$w(j) = \frac{C(j)N}{\sum_i C(i)N_i}, \quad (10)$$

where N and N_i denote the total number of instances and the number of class $i \in \{0,1\}$ instances in the training data respectively, and $C(j)$ is given in (9). We also compared our proposed method with MetaCost [7] and MetaCost_NBC, which differs from MetaCost only in that MetaCost_NBC employs the Naïve Bayes committee (NBC) procedure [45] while MetaCost uses Bagging. In our experiments, MetaCost was implemented by following the parameter settings suggested in [7], with Naïve Bayes as the classification method.

In addition to distribution adjustment methods, we compared our proposed method to threshold adjustment methods as well. One benchmark method is the standard threshold adjustment method [10]. It computes the cost-sensitive threshold using (4) and classifies instances using (3) with $p(x)$ substituted with its Naïve Bayes estimation. As discussed in Section 3, a number of methods have been proposed to enhance the accuracy of estimating $p(x)$ by Naïve Bayes. Hence, an interesting benchmark method is the combination of threshold adjustment and accuracy enhancement. Specifically, more accurate estimation of $p(x)$ produced by an accuracy enhancement method is plugged into (3) for cost-sensitive classification. The accuracy enhancement methods used in our experiments are the locally weighted Naïve Bayes method [16] and the hidden Naïve Bayes method [25], the performance of which has been shown favorable or comparable to other representative accuracy enhancement methods [16], [25]. The last benchmark method is the standard Naïve Bayes method [32]. This benchmark method also serves as a baseline and a cost-sensitive method is expected to outperform the baseline in terms of the misclassification cost. Table 3 lists the methods compared in the experiments. In the remainder of the paper, we refer to these methods using their abbreviations.

We adopted two different cost models to produce different types of cost matrices for our experiments. Cost model 1 randomly generates costs for a cost matrix. Cost model 2 emulates a situation frequently encountered in real world and it purposely sets the cost of misclassifying a minority instance (i.e., an instance belonging to a less frequent class) higher than that of misclassifying a majority instance (i.e., an instance belonging to a more frequent class). Taking direct marketing as an example, the cost of misclassifying a potential buyer (i.e., a minority instance) as a nonbuyer is normally higher than that of misclassifying a nonbuyer (i.e., a majority instance) as a potential buyer. In our experiments, cost model 1 set $C(i,j)$ to be 0 if $i = j$ and randomly chose a real number from interval (0,10) for $C(i,j)$ if $i \neq j$. Under cost model 2, $C(i,j)$ was also set to be 0 if $i = j$. Similarly, a real number h was chosen from interval (0,10) for $C(i,j)$ if $i \neq j$ and j was the majority class. However, $C(i,j)$ was set to be gh if $i \neq j$ and j was the minority class, where g was randomly generated from interval (1,10).

TABLE 3
Methods Compared in the Experiments

Method	Abbreviation	Note
inference-based Naïve Bayes	INB	proposed method
random undersampling	RUS	benchmark
random oversampling	ROS	benchmark
one-sided selection	OSS	benchmark
SMOTE	SMOTE	benchmark
instance weighting	WGT	benchmark
MetaCost	MetaCost	benchmark
MetaCost_NBC	MetaCost_NBC	benchmark
threshold adjustment	THR	benchmark
threshold adjustment + locally weighted Naïve Bayes (LWNB)	t-LWNB	benchmark
threshold adjustment + hidden Naïve Bayes (HNB)	t-HNB	benchmark
Naïve Bayes	NB	benchmark, baseline

4.2 Experimental Results and Analysis

We conducted experiments with the methods summarized in Table 3 using the data sets listed in Table 2 and cost matrices generated according to the cost models. An experimental run consisted of the following steps. First, a cost matrix was generated according to one of the cost models. A data set was next divided into two parts, by randomly selecting 2/3 of its instances as the training data and the remaining instances as the test data. Each of the methods was then applied to the training data to produce a classifier, which in turn, classified instances in the test data. For each method, its cost of misclassifying instances in the test data was calculated according to the initially generated cost matrix. Finally, the methods were ranked according to their misclassification costs. We followed the ranking schema described in [47] to rank the methods. Specifically, the method incurring the least misclassification cost was assigned rank 1 and the method incurring the second least misclassification cost was assigned rank 2 and so on. For methods incurring the same misclassification cost, average ranks were assigned. The above described experimental run was conducted 100 times for every data set under each cost model. Tables 4 and 5 report average ranks over 100 experimental runs for each method with every data set, under cost models 1 and 2, respectively. Interestingly, WGT and THR incur exactly the same cost across all data sets under both cost models.[4] This phenomenon is rigorously explained in the online Appendix, which can be found on the Computer Society Digital Library at http://doi. ieeecomputersociety. org/10.1109/TKDE.2012.196. During each experimental

4. This phenomenon is also observed in [44].

TABLE 4
Average Rank by Misclassification Cost (Cost Model 1)

Data Set	INB	RUS	ROS	OSS	SMOTE	WGT/THR	MetaCost	MetaCost_NBC	t-LWNB	t-HNB	NB
adult	3.00	5.34	5.86	2.79	6.71	5.59	11.00	8.47	12.00	4.63	7.03
australian	5.18	5.76	6.78	5.57	6.28	7.19	9.19	8.19	6.03	1.01	9.65
blood	3.32	5.91	6.13	5.17	5.33	6.26	9.75	10.23	6.43	4.52	8.70
breast-c	4.34	5.32	6.52	8.39	6.70	6.77	4.82	4.98	7.21	8.05	8.16
breast-w (original)	2.44	7.62	6.09	7.30	5.74	6.16	6.06	7.63	6.47	9.23	7.11
diabetes	2.72	5.87	5.73	6.89	4.63	5.27	9.06	7.82	9.75	5.67	9.34
german	2.74	4.92	4.82	5.91	8.14	4.87	9.79	9.45	8.50	5.05	8.97
haberman	2.67	6.06	5.92	7.42	5.52	5.84	5.59	5.33	8.62	11.09	8.14
horse	5.27	5.07	7.08	5.57	7.91	6.67	10.40	8.59	6.30	1.40	7.10
ionosphere	4.40	5.43	4.60	7.09	3.71	4.38	9.44	9.47	9.25	11.60	4.28
labor	3.64	6.95	4.88	8.35	5.81	4.96	10.61	5.94	10.30	6.54	5.08
liver	3.37	6.92	7.05	8.63	6.43	5.92	8.55	6.46	3.79	5.82	9.17
monk	4.54	6.47	5.91	6.29	5.90	5.94	7.61	9.91	2.29	10.25	6.98
pima	2.60	5.72	6.06	6.62	4.91	5.51	9.29	7.37	9.69	6.14	8.60
sick	4.50	7.36	6.32	4.51	8.48	6.84	11.71	6.28	1.64	8.22	5.33
sonar	5.27	5.39	7.07	5.65	7.95	6.70	10.09	9.58	1.81	3.75	8.07
spambase	5.05	5.16	5.13	6.71	10.24	4.75	10.01	8.96	7.99	4.97	4.29
spect	6.76	7.27	7.32	8.44	4.88	7.30	8.34	4.48	2.79	4.09	9.06
statlog-heart	3.32	5.82	5.58	5.95	7.49	5.65	8.85	8.25	7.12	7.73	6.62
wine-quality(red)	3.67	6.80	6.09	7.82	6.28	6.29	9.13	8.15	5.02	2.30	10.20
libras	4.16	5.97	5.20	7.09	6.42	4.78	9.53	10.10	7.11	8.25	4.63
musk	3.87	5.30	4.35	8.90	4.35	4.58	10.96	9.30	7.09	9.13	5.61
semeion	3.65	6.26	4.52	5.57	8.56	4.89	4.89	9.56	12.00	6.55	6.69

run, we also recorded for our proposed method: the number of hard instances, the number of recursive calls of itself, and the running time. Under both cost models and across all data sets, on average 1 percent of the test instances are left as hard instances, the average number of recursive calls is 1.7, and the average running time is 5.9 sec on a machine with 1.7 GHz processor and 6 GB RAM. Parameter k of INB was estimated using the heuristic described in Section 3. The average k value for a data set (averaged across 100 experimental runs and both cost models) ranges from 1.20 to 38.93.

TABLE 5
Average Rank by Misclassification Cost (Cost Model 2)

Data Set	INB	RUS	ROS	OSS	SMOTE	WGT/THR	MetaCost	MetaCost_NBC	t-LWNB	t-HNB	NB
adult	3.16	4.81	5.33	2.81	7.04	5.20	11.00	8.51	12.00	4.17	8.79
australian	5.52	4.85	6.87	6.55	6.77	6.82	10.04	7.37	5.07	1.00	10.35
blood	3.52	6.56	6.48	4.29	5.07	6.23	9.39	10.21	6.14	5.14	8.77
breast-c	3.91	5.82	6.20	8.26	6.57	6.51	4.75	4.74	8.29	8.23	8.26
breast-w (original)	2.93	7.40	4.99	7.55	5.36	5.20	7.12	7.52	7.93	9.91	6.91
diabetes	2.42	6.48	5.54	7.05	5.03	5.36	9.14	7.26	9.46	5.37	9.55
german	2.29	5.42	4.88	5.65	7.87	4.75	9.41	8.92	9.51	5.20	9.38
haberman	2.63	6.20	6.14	7.40	5.74	6.04	5.40	5.05	8.11	10.53	8.75
horse	5.25	4.84	6.71	6.10	8.84	6.74	10.01	8.40	6.40	1.56	6.44
ionosphere	4.36	6.14	4.31	7.98	2.99	4.26	9.09	9.28	9.70	11.53	4.12
labor	3.43	7.48	4.57	8.85	5.56	4.91	10.14	6.01	10.67	6.53	4.97
liver	3.44	6.76	5.82	8.48	6.08	6.23	7.83	5.59	4.53	6.44	10.60
monk	3.87	6.96	5.50	6.26	5.56	5.59	7.87	10.97	2.88	9.83	7.16
pima	2.84	6.25	5.65	6.60	4.61	5.94	8.95	6.92	9.16	5.93	9.24
sick	4.37	7.64	6.87	3.91	7.93	7.06	11.45	7.25	1.82	9.56	3.11
sonar	5.49	5.98	6.74	5.43	7.04	6.89	9.84	9.16	2.01	4.10	8.46
spambase	5.37	4.12	4.34	5.38	9.23	4.82	10.48	10.09	9.61	5.75	4.00
spect	6.89	7.87	7.33	9.29	4.47	7.42	8.17	3.45	2.15	3.25	10.32
statlog-heart	3.41	6.07	5.47	5.46	6.97	5.80	9.22	8.77	6.75	7.54	6.76
wine-quality(red)	3.37	7.66	5.61	8.42	6.43	6.05	8.92	6.88	5.72	2.51	10.40
libras	3.84	5.66	5.07	7.59	5.62	4.56	9.33	12.00	7.05	8.57	4.16
musk	4.11	3.71	4.45	8.05	4.76	4.67	10.72	9.60	9.36	7.92	6.00
semeion	4.06	5.50	4.79	4.85	6.81	4.79	5.39	11.39	11.60	8.02	6.03

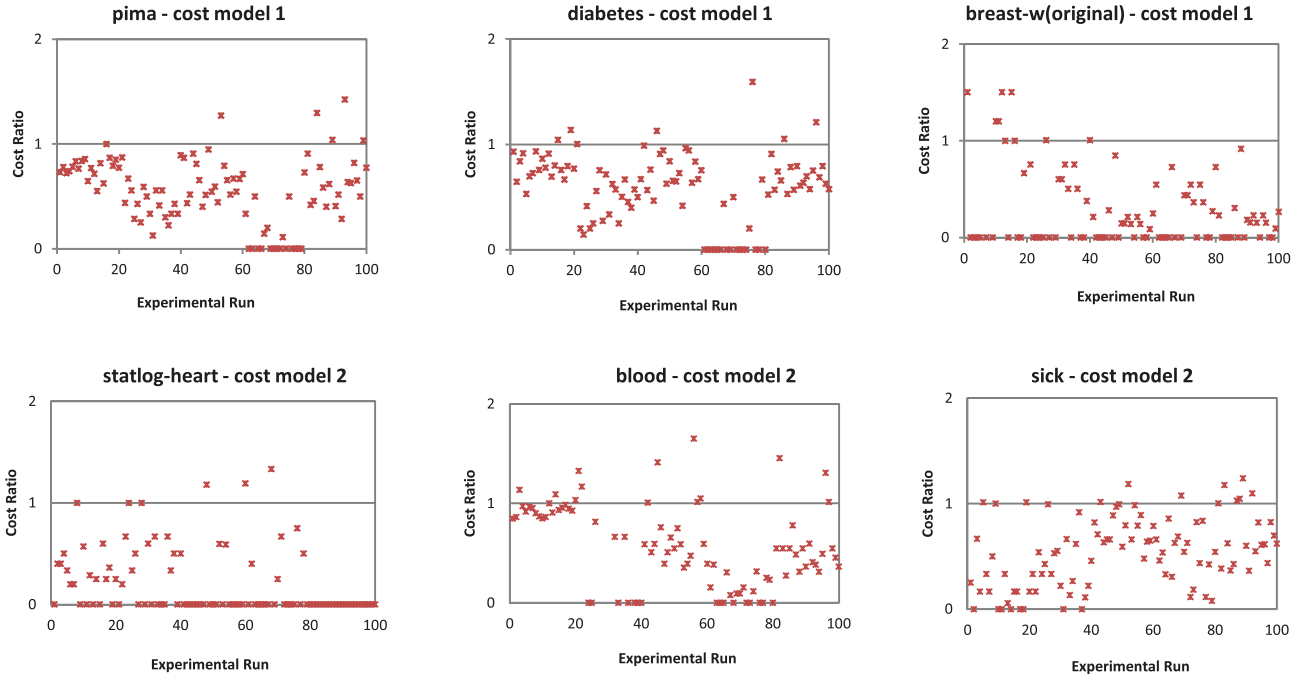

Fig. 3. Ratio of the cost of misclassifying instances with Naïve Bayes estimated probability between p^* and $\bar{Y}$ by INB to that by THR.

A suitable statistical test to analyze our experimental results is the Friedman test with the corresponding Bonferroni-Dunn post-hoc test because our experiments involve more than two classification methods over multiple data sets [47]. The Friedman test with the corresponding Bonferroni-Dunn post-hoc test consists of two tests: the main Friedman test and the post-hoc Bonferroni-Dunn test [47]. The main Friedman test reveals whether there is performance difference among all classification methods including the proposed one and the benchmark methods [47]. If there is difference according to the main test, the Bonferroni-Dunn test further suggests whether the proposed method outperforms each of the benchmark methods [47]. We applied the statistical test to average ranks in Tables 4 and 5. The main Friedman test suggests that there is performance difference among the methods at 95 percent significance level under both cost models; and the Bonferroni-Dunn test further suggests that the proposed method outperforms each of the benchmark methods at 95 percent significance level under both cost models, except that the proposed method outperforms ROS at 90 percent significance level under cost model 2. Statistical testing results reveal the superiority of the proposed method over the benchmark methods, which is analyzed next. Since THR and WGT produce the same classification, we focus on analyzing the performance comparison between INB and one of the two methods—THR. The superiority of INB over THR is primarily attributed to the following factor. According to Lemmas 1 and 2, instances with Naïve Bayes estimated probability falling between p^* and $\bar{Y}$ need to be carefully classified rather than being arbitrarily classified all as class 1 or all as class 0 by THR. In aware of this, INB constructs appropriate training data to classify these instances iteratively. As a result, INB incurs less cost of misclassifying instances with Naïve Bayes estimated probability falling between p^* and $\bar{Y}$ than THR, which in turn leads to the superior performance of INB over THR. According to our experimental results, compared to THR, INB reduces the cost of misclassifying instances with Naïve Bayes estimated probability falling between p^* and $\bar{Y}$ by 42 percent under cost model 1 and by 41 percent under cost model 2, averaged over all data sets. Taking several data sets as examples, Fig. 3 plots the ratio of the cost of misclassifying instances with Naïve Bayes estimated probability falling between p^* and $\bar{Y}$ by INB to that by THR across 100 experimental runs. As shown in the figure, the cost incurred by INB is clearly lower that by THR, i.e., most cost ratios below 1.

We analyze stratification methods next. Elkan [10] shows that changing prior class probabilities in the training data has limited effect on the Naïve Bayes method and suggests threshold adjustment (e.g., THR) a more effective way of transforming Naïve Bayes cost-sensitive than stratification. Since INB is superior to THR, INB also outperforms stratification methods. Benchmark methods t-LWNB and t-HNB employ the locally weighted Naïve Bayes (LWNB) method [16] and the hidden Naïve Bayes method (HNB) [25], respectively, which generally produce more accurate estimation of $p(x)$ than standard Naïve Bayes [16], [25]. t-LWNB and t-HNB aim at achieving better classification performance through more accurate probability estimation. However, the bias-variance decomposition determining probability estimation errors is different from that determining classification errors [14]. As a result, enhanced probability estimation does not necessarily lead to improved classification performance and oftentimes degrades the performance [14]. For example, compared to THR, t-LWNB increases the cost of misclassifying instances with Naïve Bayes estimated probability falling between p^* and $\bar{Y}$ by 8.4 percent under cost model 1 and by 12.7 percent under cost model 2, averaged over all data sets. In aware of that improving classification performance through enhanced

probability estimation may be misguided [14], our proposed method (INB) relies on inferring the order relation between $p(x)$ and p^*, which has a direct impact on classification decisions according to (3), rather than improving the accuracy of probability estimation. Thus, INB outperforms t-LWNB and t-HNB according to our experimental results.

We finally study MetaCost. Evaluating the performance of MetaCost on transforming Naïve Bayes cost-sensitive is identified as an important future research topic by the inventor of MetaCost [7]. Our experimental results show that MetaCost is among the worst performing benchmark methods. The underperformance of MetaCost on turning Naïve Bayes cost-sensitive has been explained from two different standpoints. Domingos [7] predicts the ineffectiveness of MetaCost on turning Naïve Bayes cost-sensitive, attributes the ineffectiveness to the Bagging procedure [3] employed by MetaCost, and recommends a remedy of replacing the Bagging procedure with the NBC procedure [45]. Ting [38] argues with strong empirical evidences that MetaCost is not an effective cost-sensitive method. Hence, replacing Bagging with NBC may not rescue MetaCost for turning Naïve Bayes cost-sensitive. According to our experimental results, MetaCost_NBC still lags behind most other benchmark methods and INB. Overall, our experimental results suggest that neither MetaCost nor MetaCost_NBC is effective in turning Naïve Bayes cost-sensitive.

5 A Case Study

We study the performance of the proposed method using a real world direct marketing application at an online retailer. The online retailer regularly mails catalogs to its current customers (i.e., those who purchased from the retailer before), with the hope that some of them will repurchase from the retailer in the future. For each of its current customers, the retailer must decide whether or not to mail him or her a catalog. To study the performance of the proposed method, we phrase the decision problem as a cost-sensitive classification problem. In particular, a customer is classified as a purchaser or a nonpurchaser by a classification method and catalogs are only mailed to classified purchasers. The cost of misclassifying a nonpurchaser as a purchaser is set as \$0.78, which is the fee of mailing and printing a catalog according to domain experts working at the retailer. On the other hand, the cost of misclassifying a purchaser (i.e., those who would purchase if receiving a catalog) as a nonpurchaser is the loss of the profit that could be generated from the purchaser. According to domain experts, the cost of misclassifying a purchaser as a nonpurchaser is set as \$12.07, which is the average profit generated from a purchase.

We analyzed shopping behaviors of 18,578 customers at the retailer and constructed a data set of these customers. For each customer, data describing his or her shopping behavior until the end of June 2011 were extracted from the online retailer's database. A customer was labeled as a purchaser if he or she repurchased in June 2011 whereas a customer was labeled as a nonpurchaser otherwise. Among the 18,578 customers, 1,376 customers repurchased in June 2011, yielding a repurchase rate of 7.4 percent. A customer's shopping history until the end of May 2011 was used to predict his or her repurchase in June 2011. We followed the RFM (recency, frequency, monetary) method to compute measures from a customer's shopping history data. The RFM method is widely used to analyze a customer's shopping history [48], which generally reveals how recently, how often, and how much a customer purchased. In consultation with domain experts, we compute from shopping history data (until the end of May 2011) the following RFM measures: time elapsed since the last purchase, time elapsed since the last visit to the retailer's Web site, time elapsed since the customer's registration, frequency of purchase, frequency of visiting the retailer's site, total amount of purchase, number of coupons used, and number of campaigns participated. In addition to the RFM measures, domain experts helped us identify the following attributes that could be useful in predicting a customer's repurchase: gender, age, time elapsed since the last product return, frequency of product return, total refund amount, and total return fee paid. In sum, we collected a data set of 18,578 records, each of which described a customer with the attributes summarized in Table 6.

TABLE 6
Customer Attributes

Attribute	Category
repurchase in June 2011 or not	class label
time elapsed since the last purchase	recency
time elapsed since the last visit to the retailer's Web site	
time elapsed since the customer's registration	
frequency of purchase	frequency
frequency of visiting the retailer's site	
total amount of purchase	monetary
number of coupons used	
number of campaigns participated	
gender	demographics
age	
time elapsed since the last product return	product return
frequency of product return	
total refund amount	
total return fee paid	

Using the customer data set, we conducted experiments to compare our proposed method and the benchmark methods listed in Table 3. In an experiment, the data set was randomly divided into two equally sized parts: one part as the training data and the other as the test data. Each of the methods was then applied to the training data to produce a classifier, which in turn, classified customers in the test data as purchaser or nonpurchaser. The misclassification cost of each method was finally calculated according to the costs given at the start of the case study. The above described experimental procedure was repeated for 10 times. We apply the Friedman test with the corresponding Bonferroni-Dunn post-hoc test [47] to misclassification costs collected in the experiments. The test results reveal that our proposed method outperforms each of the benchmark methods at 95 percent significance level. Table 7 lists, for each method,

TABLE 7
Average and Standard Deviation of Misclassification Costs for a Direct Marketing Application: INB versus Benchmark Methods

Method	Average	Standard Deviation
INB	$2,731.31	$130.60
RUS	$2,901.14	$124.53
ROS	$2,907.37	$134.41
OSS	$3,017.34	$106.81
SMOTE	$2,913.55	$104.56
WGT/THR	$2,901.18	$129.61
MetacCost	$2,977.05	$153.77
MetacCost_NBC	$3,480.01	$143.68
t-LWNB	$4,148.28	$125.00
t-HNB	$2,930.51	$129.87
NB	$3,092.01	$146.89

the average and the standard deviation of misclassification costs over the experiments. Across all 10 experiments, the average cost reduction by our method over a benchmark method ranges from 5.9 to 34.2 percent.

6 Conclusion

In this paper, we develop a cost-sensitive Naïve Bayes method from a novel perspective of inferring the order relation between an instance's true classification probability of belonging to the class of interest and the cost-sensitive threshold. Employing the oversmoothing nature of the Naïve Bayes method, we propose how to infer the order relation from the training data. A cost-sensitive Naïve Bayes method is then developed based on the inference of the order relation. We conduct experiments to benchmark the proposed method against major existing methods for turning Naïve Bayes cost-sensitive. The experimental results show that our proposed method significantly outperforms existing methods.

Our research could be extended along several important directions. One area merit worth of exploration is to extend the proposed method to solve classification problems with more than two classes, i.e., polychotomous classification problems. A possible solution is to decompose a polychotomous classification problem into multiple binary classification problems and solve each binary classification problem using the proposed method. In this vein, methods on how to decompose a polychotomous classification problem into binary classification problems as well as how to combine binary classification results have been well studied [1], [23] and they could be used in the solution. In addition, ensemble methods such as boosting [13] have been employed to boost the performance of cost-sensitive classifications [11], [37]. Hence, how to integrate our proposed method and a proper ensemble method to further enhance the performance of the proposed method might be another area worth of studying. Third, it is interesting to extend our proposed method to consider situations where the exact value of the misclassification cost $C(i, j)$ is unknown. For example, one may only know that $C(i, j)$ is bounded within an interval [29]. Toward that end, recent work [29] on learning with cost intervals and cost distributions provides useful directions. Finally, our preliminary experimental study shows that INB significantly outperforms NB under the 0/1 loss function.[5] Future work should analyze and leverage the advantage of INB over NB in this noncost-sensitive scenario.

References

[1] E.L. Allwein, R.E. Schapire, and Y. Singer, "Reducing Multiclass to Binary: A Unifying Approach for Margin Classifiers," *J. Machine Learning Research,* vol. 1, pp. 113-141, 2000.

[2] L. Breiman, J.H. Friedman, R.A. Olshen, and C.J. Stone, *Classification and Regression Trees.* Chapman and Hall, 1984.

[3] L. Breiman, "Bagging Predictors," *Machine Learning,* vol. 24, pp. 123-140, 1996.

[4] T.M. Cover and P.E. Hart, "Nearest Neighbor Pattern Classification," *IEEE Trans. Information Theory,* vol. IT-13, no. 1, pp. 21-27, Jan. 1967.

[5] N.V. Chawla, K.W. Bowyer, L.O. Hall, and W.P. Kegelmeyer, "SMOTE: Synthetic Minority Over-Sampling Technique," *J. Artificial Intelligence Research,* vol. 16, pp. 321-357, 2002.

[6] P. Domingos and M. Pazzani, "On the Optimality of the Simple Bayesian Classifier Under Zero-One Loss," *Machine Learning,* vol. 29, pp. 103-130, 1997.

[7] P. Domingos, "MetaCost: A General Method for Making Classifiers Cost-Sensitive," *Proc. Fifth Int'l Conf. Knowledge Discovery Data Mining,* pp. 155-164, 1999.

[8] C. Drummond and R.C. Holte, "C4.5, Class Imbalance, and Cost Sensitivity: Why Under-Sampling Beats Over-Sampling," *Proc. Workshop Learning Imbalanced Datasets (ICML'03),* 2003.

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

[10] C. Elkan, "The Foundation of Cost-Sensitive Learning," *Proc. 17th Int'l Joint Conf. Artificial Intelligence,* pp. 973-978, 2001.

[11] W. Fan, S.J. Stolfo, J. Zhang, and P. Chan, "AdaCost: Misclassification Cost-Sensitive Boosting" *Proc. 17th Int'l Conf. Machine Learning,* pp. 97-105, 1999.

[12] A. Frank and A. Asuncion, *UCI Machine Learning Repository,* School of Information and Computer Science, Univ. of California, http://archive.ics.uci.edu/ml, 2010.

[13] Y. Freund and R.E. Schapire, "A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting," *J. Computer System Sciences,* vol. 55, no. 1, pp. 119-139, 1997.

[14] J.H. Friedman, "On Bias, Variance, 0/1-Loss, and the Curse-of-Dimensionality," *Data Mining Knowledge Discovery,* vol. 1, pp. 55-77, 1997.

[15] N. Friedman, D. Geiger, and M. Goldszmidt, "Bayesian Network Classifiers," *Machine Learning,* vol. 29, nos. 2/3, pp. 131-163, 1997.

[16] E. Frank, M. Hall, and B. Pfahringer, "Locally Weighted Naive Bayes," *Proc. Conf. Uncertainty Artificial Intelligence,* pp. 249-256, 2003.

[17] J. Gama, "Iterative Bayes," *Intelligent Data Analysis,* vol. 4, pp. 475-488, 2000.

[18] A.K. Ghosh, "On Optimum Choice of k in Nearest Neighbor Classification," *Computational Statistics Data Analysis,* vol. 50, no. 11, pp. 3113-3123, 2006.

[19] P. Hall, B.U. Park, and R. Samworth, "Choice of Neighbor Order in Nearest-Neighbor Classification," *Annals Statistics,* vol. 36, no. 5, pp. 2135-2152, 2008.

[20] J. Han and M. Kamber, *Data Mining: Concepts and Techniques,* second ed. Morgan Kaufmann, 2005.

[21] D.J. Hand and W.E. Henley, "Statistical Classification Methods in Consumer Credit Scoring: A Review," *J. Royal Statistical Soc.: Series A,* vol. 160, no. 3, pp. 523-541, 1997.

[22] D.J. Hand, H. Mannila, and P. Smyth, *Principles of Data Mining.* MIT Press, 2001.

5. Because of space consideration, detailed experimental results are not provided.

[23] T. Hastie and R. Tibshirani, "Classification by Pairwise Coupling," *Annals Statistics,* vol. 26, no. 2, pp. 451-471, 1998.
[24] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning.* Springer, 2001.
[25] L. Jiang, H. Zhang, and Z. Cai, "A Novel Bayes Model: Hidden Naïve Bayes," *IEEE Trans. Knowledge Data Eng.,* vol. 21, no. 10, pp. 1361-1371, Oct. 2009.
[26] M. Kubat and S. Matwin, "Addressing the Curse of Imbalanced Training Sets: One-Sided Selection," *Proc. 14th Int'l Conf. Machine Learning,* pp. 179-186, 1997.
[27] A. Lemmens and C. Croux, "Bagging and Boosting Classification Trees to Predict Churn," *J. Marketing Research,* vol. 43, no. 2, pp. 276-286, 2006.
[28] C.X. Ling and C. Li, "Data Mining for Direct Marketing: Problems and Solutions," *Proc. Fourth Int'l Conf. Knowledge Discovery Data Mining,* pp. 73-79, 1998.
[29] X.-Y. Liu and Z.-H. Zhou, "Learning with Cost Intervals," *Proc. 16th Int'l Conf. Knowledge Discovery Data Mining,* pp. 403-412, 2010.
[30] R. Lippmann, "Pattern Classification Using Neural Networks," *IEEE Commun. Magazine,* vol. CM-27, no. 11, pp. 47-64, Nov. 1989.
[31] D. Margineantu, "Class Probability Estimation and Cost-Sensitive Classification Decisions," *Proc. 13th European Conf. Machine Learning,* pp. 270-281, 2002.
[32] T.M. Mitchell, *Machine Learning.* McGraw Hill, http://www.cs.cmu.edu/tom/NewChapters.html, 1997.
[33] M. Pazzani, C. Merz, P. Murphy, K. Ali, T. Hume, and C. Brunk, "Reducing Misclassification Costs," *Proc. 11th Int'l Conf. Machine Learning (ICML '94),* pp. 217-225, 1994.
[34] J.R. Quinlan, *C4.5: Programs for Machine Learning.* Morgan Kaufmann, 1993.
[35] A.K. Seewald, "An Evaluation of Naïve Bayes Variants in Content-Based Learning for Spam Filtering," *Intelligent Data Analysis,* vol. 11, pp. 497-524, 2007.
[36] V.S. Sheng and C.X. Ling, "Thresholding for Making Classifiers Cost-Sensitive," *Proc. 21st Nat'l Conf. Artificial Intelligence,* pp. 476-481, 2006.
[37] K.M. Ting, "A Comparative Study of Cost-Sensitive Boosting Algorithms," *Proc. 17th Int'l Conf. Machine Learning (ICML '00),* pp. 983-990, 2000.
[38] K.M. Ting, "An Empirical Study of MetaCost using Boosting Algorithms," *Proc. 11th European Conf. Machine Learning (ECML '00),* pp. 413-425, 2000.
[39] K.M. Ting, "An Instance-Weighting Method to Induce Cost-Sensitive Trees," *IEEE Trans. Knowledge and Data Eng.,* vol. 14, no. 3, pp. 659-665, May/June 2002.
[40] V. Vapnik, *Statistical Learning Theory.* John Wiley, 1998.
[41] G.I. Webb, J.R. Boughton, and Z. Wang, "Averaged One-Dependence Estimators: Preliminary Results," *Proc. First Australasian Data Mining Workshop,* pp. 65-73, 2002.
[42] W.H. Wolberg and O.L. Mangasarian, "Multisurface Method of Pattern Separation for Medical Diagnosis Applied to Breast Cytology," *Proc. Nat'l Academy of Sciences USA,* vol. 87, no. 23, pp. 9193-9196, 1990.
[43] B. Zadrozny and C. Elkan, "Learning and Making Decisions When Costs and Probabilities are Both Unknown," *Proc. Seventh Int'l Conf. Knowledge Discovery Data Mining (KDD '01),* pp. 204-213, 2001.
[44] H. Zhao, "Instance Weighting versus Threshold Adjusting for Cost-Sensitive Classification," *Knowledge Information Systems,* vol. 15, pp. 321-334, 2008.
[45] Z. Zheng, "Naïve Bayesian Classifier Committees," *Proc. 10th European Conf. Machine Learning (ECML '98),* pp. 196-207, 1998.
[46] Z.-H. Zhou and X.-Y. Liu, "Training Cost-Sensitive Neural Networks with Methods Addressing the Class Imbalance Problem," *IEEE Trans. Knowledge and Data Eng.,* vol. 18, no. 1, pp. 63-77, Jan. 2006.
[47] J. Demsar, "Statistical Comparisons of Classifiers over Multiple Data Sets," *J. Machine Learning Research,* vol. 7, pp. 1-30, 2006.
[48] Y. Kim, N. Street, G. Russell, and F. Menczer, "Customer Targeting: A Neural Network Approach Guided by Genetic Algorithms," *Management Science,* vol. 51, no. 2, pp. 264-276, 2005.
[49] G.I. Webb, J.R. Boughton, and Z. Wang, "Not So Naïve Bayes: Aggregating One-Dependence Estimators," *Machine Learning,* vol. 58, pp. 5-24, 2005.
[50] F. Zheng and G.I. Webb, "Semi-Naïve Bayesian Classification," Monash Univ. Working Paper, 2008.
[51] G.I. Webb, J.R. Boughton, F. Zheng, K.M. Ting, and H. Salem, "Decreasingly Naïve Bayes: Aggregating n-Dependence Estimators," Monash Univ. Working Paper, 2010.
[52] H. Zhang and S. Sheng, "Learning Weighted Naive Bayes with Accurate Ranking," *Proc. IEEE Fourth Int'l Conf. Data Mining (ICDM '04),* pp. 567-570, 2004.
[53] M. Hall, "A Decision Tree-Based Attribute Weighting Filter for Naïve Bayes," *Knowledge-Based Systems,* vol. 20, no. 2, pp. 120-126, 2007.

Xiao Fang received the BS and MS degrees from Fudan University, China, and the PhD degree in management information systems from the University of Arizona. He is currently an assistant professor of information systems in the Department of Operations and Information Systems, University of Utah. His current research interests include the areas of business intelligence, data mining, and social computing. He has published in the *ACM Transactions on Information Systems*, *ACM Transactions on Internet Technology*, *Communications of the ACM*, *INFORMS Journal on Computing*, *Journal of Management Information Systems*, *Journal of the American Society for Information Science, and Technology*, among others.

Online Appendix

Xiao Fang

In this Appendix, we first show in Lemma A1 a general condition under which instance weighting produces the same cost-sensitive classification as threshold adjustment.

Lemma A1. For binary classification using Naïve Bayes, instance weighting produces the same cost-sensitive classification as threshold adjustment if the following condition is satisfied by instance weighting:

$$\frac{w(1)}{w(0)} = \frac{C(0,1) - C(1,1)}{C(1,0) - C(0,0)}, \tag{A1}$$

where $w(1)$ and $w(0)$ denote the weight of class 1 instance and that of class 0 instance respectively.

Proof. Let $\hat{P}(Y = i)$ and $\tilde{P}(Y = i)$ be the prior probability of class $i \in \{0,1\}$ estimated from the training data and the instance-weighted training data respectively, where the instance-weighted training data refer to the training data with its instances weighted. We have

$$\frac{\tilde{P}(Y=1)}{\tilde{P}(Y=0)} = \frac{w(1)\hat{P}(Y=1)}{w(0)\hat{P}(Y=0)}. \tag{A2}$$

Let $\hat{p}(x)$ and $\tilde{p}(x)$ be the probability of an unclassified instance x belonging to class 1 estimated from the training data and the instance-weighted training data respectively by the Naïve Bayes method. Conceivably, $1-\hat{p}(x)$ and $1-\tilde{p}(x)$ are the probability of an unclassified instance x belonging to class 0 estimated from the training data and the instance-weighted training data respectively. By (5), we have

$$\frac{\tilde{p}(x)}{1-\tilde{p}(x)} = \frac{\tilde{P}(Y=1)\prod_{l=1}^{n}\tilde{P}(X_l = x_l \mid Y=1)}{\tilde{P}(Y=0)\prod_{l=1}^{n}\tilde{P}(X_l = x_l \mid Y=0)}, \tag{A3}$$

where $\tilde{P}(X_l = x_l \mid Y = i)$ denotes the conditional probability of $X_l = x_l$ given $Y=i$, estimated from the instance-weighted training data. Instance weighting does not change attribute distribution within each class. Hence,

$$\tilde{P}(X_l = x_l \mid Y = i) = \hat{P}(X_l = x_l \mid Y = i), \tag{A4}$$

where $\hat{P}(X_l = x_l \mid Y = i)$ is the conditional probability of $X_l = x_l$ given $Y=i$, estimated from the training data. By (A2) and (A3), we have,

$$\begin{aligned}
\frac{\tilde{p}(x)}{1-\tilde{p}(x)} &= \frac{w(1)\hat{P}(Y=1)\prod_{l=1}^{n}\tilde{P}(X_l = x_l \mid Y=1)}{w(0)\hat{P}(Y=0)\prod_{l=1}^{n}\tilde{P}(X_l = x_l \mid Y=0)} \\
&= \frac{w(1)\hat{P}(Y=1)\prod_{l=1}^{n}\hat{P}(X_l = x_l \mid Y=1)}{w(0)\hat{P}(Y=0)\prod_{l=1}^{n}\hat{P}(X_l = x_l \mid Y=0)} \quad \text{by (A4)} \\
&= \frac{w(1)\hat{p}(x)}{w(0)(1-\hat{p}(x))}. \quad \text{by (5)}
\end{aligned}$$

Solving the above equation for $\tilde{p}(x)$, we have

$$\tilde{p}(x) = \frac{w(1)\hat{p}(x)}{w(0) + (w(1) - w(0))\hat{p}(x)}. \tag{A5}$$

By applying instance weighting, cost-sensitive classification of instance x is expressed as

$$Y = \begin{cases} 1 & \text{if } \tilde{p}(x) \geq 1/2, \\ 0 & \text{if } \tilde{p}(x) < 1/2. \end{cases} \tag{A6}$$

Substituting $\tilde{p}(x)$ in (A6) with (A5), the above instance weighting classification can be rewritten as

$$Y = \begin{cases} 1 & \text{if } \hat{p}(x) \geq \dfrac{w(0)}{w(1)+w(0)}, \\ 0 & \text{if } \hat{p}(x) < \dfrac{w(0)}{w(1)+w(0)}. \end{cases} \tag{A7}$$

If condition (A1) is satisfied, instance weighting classification expressed in (A7) becomes

$$Y = \begin{cases} 1 & \text{if } \hat{p}(x) \geq \dfrac{C(1,0)-C(0,0)}{C(1,0)-C(1,1)+C(0,1)-C(0,0)}, \\ 0 & \text{if } \hat{p}(x) < \dfrac{C(1,0)-C(0,0)}{C(1,0)-C(1,1)+C(0,1)-C(0,0)}. \end{cases} \tag{A8}$$

By comparing (A8) and (3), (4), it is easy to show that, if condition (A1) is satisfied, instance weighting produces the same cost-sensitive classification as threshold adjustment, where threshold adjustment substitutes $p(x)$ in (3) with its Naïve Bayes estimation $\hat{p}(x)$. □

For the instance weighting method compared in the experiments, WGT, we have

$$\frac{w(1)}{w(0)} = \frac{C(0,1)+C(1,1)}{C(1,0)+C(0,0)}. \quad \text{by (10) and (9)}$$

Given $C(1,1) = 0$ and $C(0,0) = 0$ in our experiments, condition (A1) is satisfied by WGT. By Lemma A1, WGT produces the same cost-sensitive classification as THR and consequently incurs the same misclassification cost as THR.