IIE Transactions
Publication details, including instructions for authors and subscription information:
http://www.informaworld.com/smpp/title~content=t713772245

A classification procedure for highly imbalanced class sizes

Eunshin Byon [a]; Abhishek K. Shrivastava [b]; Yu Ding [a]
[a] Department of Industrial and Systems Engineering, Texas A&M University, College Station, TX, USA
[b] Department of Manufacturing Engineering and Engineering Management, City University of Hong Kong, Kowloon, Hong Kong

Online publication date: 02 February 2010

To cite this Article Byon, Eunshin, Shrivastava, Abhishek K. and Ding, Yu(2010) 'A classification procedure for highly imbalanced class sizes', IIE Transactions, 42: 4, 288 — 303
To link to this Article: DOI: 10.1080/07408170903228967
URL: http://dx.doi.org/10.1080/07408170903228967

IIE Transactions (2010) **42**, 288–303

ISSN: 0740-817X print / 1545-8830 online
DOI: 10.1080/07408170903228967

A classification procedure for highly imbalanced class sizes

EUNSHIN BYON,[1] ABHISHEK K. SHRIVASTAVA[2] and YU DING[1,*]

[1]*Department of Industrial and Systems Engineering, Texas A&M University, College Station, TX 77843-3131, USA*
E-mail: esbyun@tamu.edu, yuding@iemail.tamu.edu
[2]*Department of Manufacturing Engineering and Engineering Management, City University of Hong Kong, Kowloon, Hong Kong*
E-mail: abhishek.shrivastava@city.edu.hk

Received September 2008 and accepted July 2009

This article develops an effective procedure for handling two-class classification problems with highly imbalanced class sizes. In many imbalanced two-class problems, the majority class represents "normal" cases, while the minority class represents "abnormal" cases, detection of which is critical to decision making. When the class sizes are highly imbalanced, conventional classification methods tend to strongly favor the majority class, resulting in very low or even no detection of the minority class. The research objective of this article is to devise a systematic procedure to substantially improve the power of detecting the minority class so that the resulting procedure can help screen the original data set and select a much smaller subset for further investigation. A procedure is developed that is based on ensemble classifiers, where each classifier is constructed from a resized training set with reduced dimension space. In addition, how to find the best values of the decision variables in the proposed classification procedure is specified. The proposed method is compared to a set of off-the-shelf classification methods using two real data sets. The prediction results of the proposed method show remarkable improvements over the other methods. The proposed method can detect about 75% of the minority class units, while the other methods turn out much lower detection rates.

Keywords: Data reduction, detection power, ensemble classifier, false alarm rate, highly imbalanced classification, resampling, support vector machine

1. Introduction

We are concerned with developing a classification rule for *highly imbalanced* two-class classification problems. That is, the number of records in the minority class is a very small fraction of that in the majority class.

One example of this is in warranty data (Mannar *et al.*, 2006). Thanks to years of quality improvement efforts (Linderman *et al.*, 2003), one should not be surprised to find that only a small fraction of the manufactured units are returned as faulty units by customers, implying that the warranty data are highly imbalanced. We have a warranty data set from a cellphone manufacturer, which has only ten faulty units among a total of 11 899 units. The percentage of the faulty units is only 0.08%. Another instance of a highly imbalanced data set is in a study of abalones (Blake and Merz, 2008), where the abalones are to be grouped into two classes according to their age. The older abalones are of special interest and should be screened out from all the abalones collected. In a data set that had 4141 abalone samples, there were only 36 samples of abalones, or 0.86% of the total, in the old-age class.

The imbalance in classes presents a challenge in developing effective classification methods because conventional classification algorithms are built principally upon the assumption that every class to be predicted has enough representatives in the training set. These classification algorithms are meant to maximize the overall prediction accuracy. When dealing with an imbalanced data set, conventional methods tend to strongly favor the majority class, and largely ignore the minority class. Hence, these methods will likely lead to very low or even no detection of the minority class when directly applied to an imbalanced data set (Kubat *et al.*, 1998; Chen *et al.*, 2005).

Such low detection rates are undesirable in many applications where the detection of the minority class units is very critical, for example, the minority class often represents "faulty units" in warranty problem or "abnormality" in biological exploration. Consider the warranty example. Product quality problems could lead to a large-scale costly recall, followed by legal actions and penalties. Therefore, it is crucial to reduce the chance of dispatching bad-quality products to consumers (Westbrook, 1987; Anderson, 1998). In the study of abalones, although it is possible to precisely decide their age by cutting the shell through the cone, staining it and counting the number of rings under a microscope, it is very time consuming to go through such a manual

*Corresponding author

procedure for every one of the 4000+ abalone samples. Thus, it is essential to establish an automatic procedure for screening the original data set and predicting the likelihood of product return or the age of an abalone from a set of easy-to-measure physical quantities.

The fundamental difficulty associated with highly imbalanced classification problems suggests that developing an automatic classification system working entirely on its own may not be a realistic goal in the immediate future. Rather, we feel that a two-step system will work better: an automatic classification system that serves as a pre-screening tool, which is followed by a manual (or automatic yet more expensive) verification procedure. The first step of the classification system will supposedly have a very high detection ability but with a relatively high false alarm rate. Its mission is to produce, from the original data set, a much smaller subset with substantially higher concentration of the minority class (that is, the class to be detected). The benefit of having the first step is to make the execution of the more expensive verification step (that is, the second step) affordable as the pre-screening step effectively narrows down the data samples that need verification. Suppose that a classification procedure with high detection power and 10% false alarm rate is devised. When used as a pre-screening tool, it can produce a data subset roughly one-tenth of the original set, with roughly the same number of minority class items (precise number depends on the actual detection rate). By contrast, the low detection power of the off-the-shelf methods makes them unsuitable as a pre-screening tool.

The specific goal of this article is to present a classification procedure to be used in the pre-screening step. We present an ensemble-based approach, which we call an Ensemble Classifier for Highly Imbalanced class sizes (ECHI), specifically designed for highly imbalanced class distributions. Ensemble classifiers use a collection of base classifiers, instead of one single classifier, to make predictions. According to Breiman (1996), bagging, one of the ensemble classification methods, reduced the prediction error by 20% on average over various problems. However, the general ensemble approach still has the underdetection problem when applied to class imbalance applications. We are able to bypass this underdetection problem by using a data reduction technique and resizing the original training set to create new training sets with more balanced class sizes. The resizing may involve up-sampling the minority class and down-sampling the majority class. Also, we reduce the size of each training set to be much smaller than the size of the original training set, so that the individual base classifier can be created in a short time. Therefore, the benefit of resizing is not only the boosted detection power but also a gain in computational efficiency that is critical in practice where it may be difficult to handle the entire training data set as a whole because of its huge size.

We believe the major contributions of this article are two-fold.

1. We propose an improved ensemble-based approach, ECHI, for addressing the underdetection problem in class-imbalance classifications. ECHI is built by majority voting of base classifiers like bagging. However, our method for constructing each base classifier achieves better accuracy to detect minority samples as well.
2. For data sets with highly imbalanced class sizes, we advocate reorienting the role of an automatic classification procedure to be a pre-screening tool. When applied to a large data set, ECHI generates a much smaller set which includes only a small, regulated percentage of majority class items that are falsely classified as minority ones, as well as most of minority class items in the original data set. Then, we can apply more accurate, but expensive, verification procedure to this smaller set in order to identify the minority items exactly. We believe this is a practical and attainable goal.

The remainder of the article is organized as follows. We start by presenting the details of the proposed classification method for handling imbalanced data in Section 2. Subsequently in Section 3, several specific issues are discussed in implementing ECHI. Next in Section 4, we describe the data sets used in this study. The prediction results of ECHI and its comparison with the aforementioned off-the-shelf alternatives are presented in Section 5. Finally, we conclude the article in Section 6 with additional discussions and comments.

2. ECHI

2.1. *Problem description*

Our goal is to develop a classification rule $\hat{C}(\mathbf{x}, S)$, with high classification accuracy for the minority class, in high class imbalance applications. Here, $S = \{(y_i, \mathbf{x}_i), i = 1, \ldots, N\}$ is the training data set, where N is the size of the data set, $\mathbf{x}_i = (x_{i1}, \ldots, x_{ip})$ is the vector of p explanatory variables for the ith record in S, and the response variable y_i is the class indicator. Since we are studying two-class problems, y_i takes binary values: when $y_i = 0$, the corresponding case belongs to the *majority class*; and when $y_i = 1$, the corresponding case belongs to the *minority class*. The classification rule $\hat{C}(\mathbf{x}, S)$ is to predict the class to which a future case, with explanatory variable $\mathbf{x}$, belongs.

In a classification problem, the loss function $L(y, \hat{y})$ gives the cost of misclassification between the actual response y and the predicted value $\hat{y}$. Since we have a binary response, we use the 0/1 loss function. In the 0/1 loss function, the loss has a value of one when the predicted class is different from the actual response and is zero otherwise. This 0/1 loss function can be expressed as

$$L(y, \hat{y}) = I(y \neq \hat{y}), \tag{1}$$

where $I(\cdot)$ is the indicator function. Given a loss function, the prediction error (PE), also called the *generalization*

error, of the classification rule $\hat{C}(\mathbf{x}, S)$ is defined as (Tibshirani, 1996):

$$PE(\hat{C}) \equiv E_{0F} E_F[L(Y, \hat{C}(\mathbf{x}, S))], \tag{2}$$

where E_F is the expectation over the training set S, whose members are independently and identically distributed, and E_{0F} is the expectation over the test observations $(Y, \mathbf{x})$.

Since we are interested in those applications whose misclassification costs of majority class units and minority class units are different, we divide the prediction error into two parts: the false alarm rate (FA) and the detection power (DP). The false alarm rate is the expected misclassification rate of classifying majority units as minority units, and the detection power is the expected rate of correctly detecting minority units. Similar to the definition of the prediction error given in Equation (2), the detection power and the false alarm rate of $\hat{C}(\mathbf{x}, S)$ can be defined as

$$DP(\hat{C}) \equiv E_{0F}\{E_F[1 - L(Y, \hat{C}(\mathbf{x}, S))] \big| Y = 1\}, \tag{3}$$

$$FA(\hat{C}) \equiv E_{0F}\{E_F[L(Y, \hat{C}(\mathbf{x}, S))] \big| Y = 0\}. \tag{4}$$

DP is also called *hits*, or *true positive rate* in literature, whereas FA is called *misses* or *false positive rate* (Chan and Stolfo, 1998). In biomedical applications, DP is called *sensitivity* whereas $(1 - FA)$ is called *specificity* (Chen *et al.*, 2005).

Since our aim is to make an automatic classification step that serves as a pre-screening tool, we want to generate a much smaller subset which retains most of the minority records in the original data set. This calls for a much enhanced detection power. It is known that pushing up the detection power of a classifier will generally lead to an increase in its false alarm rate as well. Thus, we need to regulate the false alarm rate to be below a specific percentage value while boosting the detection power substantially. Of course, selecting the threshold for regulating the false alarm depends on specific applications, especially on the cost of performing the subsequent verification procedure after our pre-screening step. We therefore formulate our classification problem as the following constrained optimization problem:

$$\begin{aligned} \max \quad & DP(\hat{C}), \\ s.t. \quad & \\ & FA(\hat{C}) \le \alpha, \\ & \hat{C} \in \Omega, \end{aligned} \tag{5}$$

where α is the threshold to regulate the false alarm rate. The decision variable is a classifier $\hat{C}$ in the set of classification rules Ω. This formulation will be materialized specifically in Section 2.3.

Solving this optimization problem analytically is not possible because the current classification theory can only evaluate the detection powers and false alarm rates for a given classifier. We narrow down the search space Ω of classifiers to the class of ensemble classifiers, parameterize the classifier and solve the optimization problem using a *data-driven approach*.

Before getting to the procedure, we need to clarify some notation we will use in the remainder of this article. We use S to denote the complete data set available to us, which is partitioned into two parts—the test data set S^l and the complementary set $S^{(l)}$ $(= S - S^l)$. The complementary data set is used to find the optimal classifier $\hat{C}$. The test data set will be used in Section 5 for validating our proposed method. Further details of this partitioning of S will be given later in Section 3.1.

2.2. *Ensemble classifiers*

In classifier ensembling methodology, people generate multiple classifiers from different training sets and synthesize individual predictions by the rule of majority voting; this synthesis forms an ensemble classifier. An ensemble classifier is defined by

$$C_A(\mathbf{x}) \equiv I(E_F[\hat{C}(\mathbf{x}, S^{(l)})] \ge 0.5). \tag{6}$$

Here $\hat{C}(\mathbf{x}, S^{(l)})$ is a base classifier, the output of which, in this binary response application, is either zero or one. Intuitively, $E_F[\hat{C}(\mathbf{x}, S^{(l)})]$ is the proportion of times when the prediction from $\hat{C}(\mathbf{x}, S^{(l)})$ at $\mathbf{x}$ is class 1, the minority class in our study, assuming that the base classifier is trained with infinitely many training samples. $I(\cdot)$ is the indicator function, meaning that if the class of $\hat{C}(\mathbf{x}, S^{(l)}) = 1$ is predicted by a simple majority of base classifiers (note that the 0.5 in the above equation implies a simple majority), the prediction from this ensemble classifier will be $C_A(\mathbf{x}) = 1$; otherwise, $C_A(\mathbf{x}) = 0$. In practice, the ensemble classifier can be estimated by

$$\hat{C}_A(\mathbf{x}) = I\left(\frac{1}{Q}\sum_{q=1}^{Q} \hat{C}(\mathbf{x}, T_q) \ge 0.5\right). \tag{7}$$

Here, the T_q, $q = 1, \ldots, Q$, are training sets created from the data set $S^{(l)}$, and Q is the ensemble size. Typically, the ensemble size Q should be large enough to get a stable result. It is also recommended to use an odd number for Q to avoid ties.

Tibshirani (1996) showed that a bagged classifier has smaller expected loss than the original base classifier for a broad array of loss functions including the 0/1 loss function. It has been shown that ensemble classifiers improve prediction accuracy over base classifiers in many applications (West *et al.*, 2005; Wezel and Potharst, 2007).

However, when the base classifiers have poor prediction power, the resulting ensemble classifier may not be able to improve the prediction accuracy sufficiently. We face this problem in the highly imbalanced classification problem when we use the conventional ensemble approach (bagging). In the conventional ensemble approach, a number of training sets are constructed by bootstrapping from

$S^{(l)}$, and each training set has the same size as $S^{(l)}$. Under this resampling scheme, the bootstrap training sets are still highly imbalanced. Thus, the base classifiers generally lead to trivial classifiers favoring the majority class when dealing with highly imbalanced data sets (Japkowicz *et al.*, 1995). Consequently, the resulting ensemble classifier also behaves similarly to its base classifiers and favors the majority class.

2.3. *A new ensemble approach*

Given a p-dimensional explanatory variable $\mathbf{x}$, the first step is to perform a dimension reduction to project $\mathbf{x}$ on to a g-dimensional ($g < p$) subspace. The reason is that prediction performance usually suffers in a high-dimensional space – a problem also commonly known as the "curse of dimensionality." Suppose that we employ a projection method $\mathcal{P}$ such that:

$$\mathcal{P} : \mathbb{R}^p \to \mathbb{R}^g; \quad \mathbf{z} = \mathcal{P}(\mathbf{x}), \tag{8}$$

where $\mathbf{z}$ is the g-dimensional explanatory variable after projection. When implementing this classifier, we will choose a specific projection method but leave the reduced dimension g as a decision variable to be decided by a optimization procedure. As such, the projection method $\mathcal{P}$ is parameterized by g, $g \in G$, where G is the set of permissible dimensions for z.

The second step is to create sensible training sets T_q that are used to create the classification rules for the base classifiers. This is basically a type of resampling operation $\mathcal{R}$. The resampling operation $\mathcal{R}$ is defined on the set $S^{(l)}$ as

$$\mathcal{R} : S^{(l)} \mapsto \{T_q\}_{q=1}^{Q}. \tag{9}$$

We fulfill this objective by applying *two resampling techniques*: down-sampling the majority class and up-sampling the minority class. Down-sampling (also known as under-sampling, or abatement) is to randomly select a subset of data records from the majority class; denote by n_0 the number of major class records after down-sampling. Up-sampling (also known as over-sampling, or augmentation) is to sample additional units *with replacement* from the minority class; denote by n_1 the number of minor class records after up-sampling. These n_0 majority class records and n_1 minority class records form a training set, $T_q, q = 1, \ldots, Q$. Note that n_1 could be larger than the size of the original minority class. We parameterize the resampling procedure $\mathcal{R}$ by using the resampling ratio $r = n_0/n_1, r \in R$, where R is the set of permissible resampling ratios.

After the training sets $\{T_q\}$'s are generated based on data reduction and resampling, a base classifier is established on each of these training sets. Any existing classification method such as Support Vector Machine (SVM), random forest, Multiple Additive Regression Tree (MART), neural network or logistic regression can be used to construct a base classifier. Again we will fix the type of base classifier in our implementation but will optimize certain parameters associated with the base classifier. We denote by $\theta \in \Theta$ the parameters associated with the base classifier.

Using the above parameterization of our ensemble classifier (using g, r, θ), we denote the base classifier as $\hat{C}(\mathbf{x}, T_q; g, r, \theta)$ and the ensemble classifier as $\hat{C}_{\mathrm{A}}(\mathbf{x}; g, r, \theta)$. Then, the optimization formulation in Equation (5) can be materialized as follows:

$$\begin{aligned} &\max_{g\in G, r\in R, \theta\in\Theta} DP(\hat{C}_{\mathrm{A}}(\mathbf{x}; g, r, \theta)), \\ &s.t. \\ &\quad FA(\hat{C}_{\mathrm{A}}(\mathbf{x}; g, r, \theta)) \le \alpha, \\ &\quad \hat{C}_{\mathrm{A}}(\mathbf{x}; g, r, \theta) = I\left(\frac{1}{Q}\sum_{q} \hat{C}(\mathbf{x}, T_q; g, r, \theta) \ge 0.5\right), \end{aligned} \tag{10}$$

where $\mathbf{x}$ is the explanatory variable of a future unit, and g, r, θ are the decision variables, also known as the *tuning parameters* in classification literature. Figure 1 illustrates the overall mechanism for the new ensembling approach.

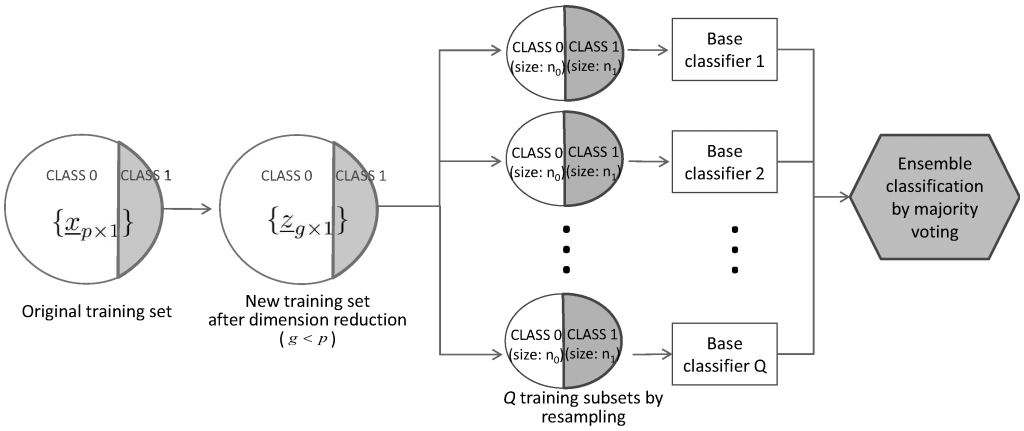

Fig. 1. Idea of the new ensembling approach combining data reduction and resampling techniques. Each ensemble classifier is constructed by assigning a set of values to the decision variables g, r, θ.

It is not possible to solve this optimization problem (10) analytically because *DP* and *FA* cannot be analytically evaluated. Therefore, we solve this problem empirically for a given data set $S^{(l)}$. Basically, we estimate the prediction error (including both *DP* and *FA*) using $S^{(l)}$ for a combination of the decision variable (g, r, θ) values and repeat the error estimation for all possible combinations of $g \in G$, $r \in R$, $\theta \in \Theta$. Usually G, R, Θ are finite sets of small size so the number of combinations of the decision variables is limited. For instance, in the cell phone warranty data example, we have that $|G| = 4$, $|R| = 7$ and $|\Theta| = 4$, so the total number of combinations is 112. The best set of decision variables is selected as the one producing the highest detection rate, while keeping the false alarm rate in control.

Thus, the key in solving the optimization problem is to use a data-driven method that can estimate the prediction errors. Towards that objective, we use the *out-of-bag* estimation proposed by Breiman (1996) in bagging. The specific procedure is as follows:

For a given $g \in G$, $r \in R$ and $\theta \in \Theta$, use the following procedure to estimate the prediction errors, and repeat the procedure until all possible combinations of choices in G, R and Θ are exhausted.

Step 1. Repeat for $q = 1, 2, \ldots, Q$:

1.1. Construct a training subset T_q from $S^{(l)}$. In doing so, for each T_q, $(n_0 + n_1)\frac{r}{1+r}$ majority class records are randomly drawn from the majority class in $S^{(l)}$ and $(n_0 + n_1)\frac{1}{1+r}$ minority class records are bootstrapped from the minority class in $S^{(l)}$. When sampling the minority class records, leave some unit(s) out for the subsequent out-of-bag estimation. The left-out minority unit(s), along with the left-out majority class units, form the out-of-bag estimation subset (or the validation subset).

1.2. Build the base classifier $\hat{C}(\mathbf{x}, T_q; g, r, \theta)$ using the training subset T_q.

Step 2. Construct the ensemble classifier $\hat{C}_{\mathrm{A}}(\mathbf{x}_i; g, r, \theta)$, for each observation $\mathbf{x}_i \in S^{(l)}$, as

$$\hat{C}_{\mathrm{A}}(\mathbf{x}_i; g, r, \theta) = I\left(\frac{1}{B_i}\sum_{q \in V_i} \hat{C}(\mathbf{x}_i, T_q; g, r, \theta) \geq 0.5\right), \quad (11)$$

where V_i is the set of indices of the training subsets that do not contain observation i and B_i is the number of such training subsets, that is, $B_i = |V_i|$.

Step 3. Compute the estimates of prediction error, detection power and false alarm rate for $\hat{C}_{\mathrm{A}}(\mathbf{x}; g, r, \theta)$:

$$\hat{PE}(\hat{C}_{\mathrm{A}}; g, r, \theta) = \frac{1}{N_0 + N_1}\sum_{i=1}^{N_0+N_1} L(y_i, \hat{C}_{\mathrm{A}}(\mathbf{x}_i; g, r, \theta)), \quad (12)$$

$$\hat{FA}(\hat{C}_{\mathrm{A}}; g, r, \theta) = \frac{1}{N_0}\sum_{i=1}^{N_0} L(y_i, \hat{C}_{\mathrm{A}}(\mathbf{x}_i; g, r, \theta)) \quad \text{for } i \in \{i : y_i = 0\}, \quad (13)$$

$$\hat{DP}(\hat{C}_{\mathrm{A}}; g, r, \theta) = \frac{1}{N_1}\sum_{i=1}^{N_1}(1 - L(y_i, \hat{C}_{\mathrm{A}}(\mathbf{x}_i; g, r, \theta))) \quad \text{for } i \in \{i : y_i = 1\}, \quad (14)$$

where N_0, N_1 are the number of records in the majority class and the minority class in $S^{(l)}$, respectively.

After getting the estimates of prediction errors, the solution to the optimization in Equation (10) is simply that among all the combinations of (g, r, θ) whose $\hat{FA}(\hat{C}_{\mathrm{A}}; g, r, \theta)$ is less than $100\alpha\%$, we choose the one with the highest detection power (that is, $\hat{DP}(\hat{C}_{\mathrm{A}}; g, r, \theta)$). When there are multiple solutions having the same degree of detection power but different false alarm rates (all smaller than $100\alpha\%$), we recommend choosing the one with the highest false alarm rate. This is because choosing the one with a higher false alarm rate will help maintain a high detection power in the testing data.

2.4. *Performance measure for ensemble classifier*

We can quantify the advantage of the ensemble classifier over its base classifiers by two measures (Tibshirani, 1996): aggregation effect (*AE*) and variance of base classifier (*Var*). AE is defined as

$$AE \equiv PE(Y, \hat{C}) - PE(Y, \hat{C}_{\mathrm{A}}). \quad (15)$$

From Equation (15) we note that *AE* is the reduced prediction error of the ensemble classifier $\hat{C}_{\mathrm{A}}$ over the base classifier $\hat{C}$. The aggregation effect can also be separated into the following two quantities: the aggregation effect in detection power and false alarm rate, respectively.

$$AE_{DP} \equiv DP(\hat{C}_{\mathrm{A}}) - DP(\hat{C}), \quad (16)$$

$$AE_{FA} \equiv FA(\hat{C}) - FA(\hat{C}_{\mathrm{A}}). \quad (17)$$

The variance of the base classifier is defined as

$$\mathrm{Var}(\hat{C}) \equiv PE(\hat{C}, \hat{C}_{\mathrm{A}}) = E_{\mathrm{F}}E_{0\mathrm{F}}\{L[\hat{C}(\mathbf{x}, S), \hat{C}_{\mathrm{A}}(\mathbf{x}, S)]\} \text{ by Equation (2).} \quad (18)$$

When using the 0/1 loss function, $\mathrm{Var}(\hat{C})$ is the expected rate at which the base classifier predicts the class differently from the ensemble classifier. High variance indicates an unstable prediction of the base classifier. Breiman (1996) showed that a bagged classifier can be considerably more stable than a single classifier. In Section 5.6, we will illustrate how much the prediction precision improves in our problem when using an ensemble classifier instead of individual base classifiers.

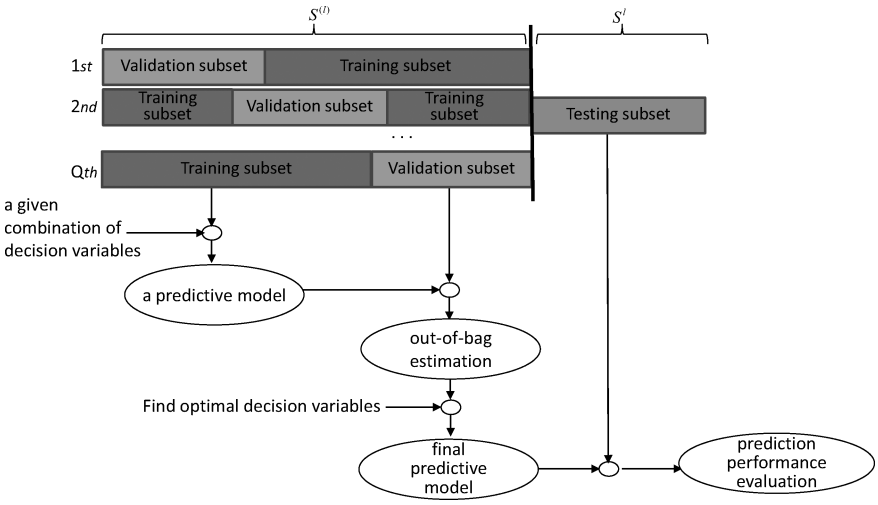

Fig. 2. The process of deciding the values of decision variables and model testing. For an ensemble classifier and its out-of-bag estimation, given a testing data set S^l, we construct multiple different training subsets and validation subsets (or out-of-bag estimation subsets) from $S^{(l)}$. A predictive model (or an ensemble classifier) is generated from the aggregation of classifiers on these Q training subsets.

3. Implementation details

In implementing ECHI, we need to specify a few details, including how the original data set is partitioned and utilized, the resampling ratio and the appropriate training subset size, and the specific methods used in this study for data reduction and as the base classifier. The overall procedure to implement ECHI is summarized at the end of this section.

3.1. *Data partition*

Recall that the historical data set S is partitioned into S^l and $S^{(l)}$ $(= S - S^l)$. In the process of constructing ECHI in Section 2.3, $S^{(l)}$ is actually further partitioned into two parts: the first part for establishing a base classifier and the second part for obtaining out-of-bag estimates. In other words, the original data set is in fact divided into three subsets: the actual training subset (the first part of $S^{(l)}$), the out-of-bag estimation subset (the second part of $S^{(l)}$, also called *validation subset* in literature), and the testing subset S^l. Figure 2 shows how the whole data set S is divided. It also demonstrates the overall process for deciding the values of decision variables and model testing. Note that for a given testing set S^l, multiple training subsets and validation subsets are generated from the complementary set $S^{(l)}$ for the ensembling purpose.

3.2. *Data reduction*

One of the reasons causing the "curse of dimensionality" for classifiers is the existence of strong correlations among the high-dimensional explanatory variables. Figure 3 shows the scatterplots for several pairs of the 34 explanatory variables in the warranty data set used in this study. One can see that several variables are strongly correlated to each other, for example, a strong positive correlation exists between the fourth and the 28th variable in the first scatterplot.

Principal Component Analysis (PCA; Johnson and Wichern, 2002) is a useful statistical technique that can find a smaller set of uncorrelated variables that can explain a significant portion of the variance in the original data. It is a computationally robust procedure and easy to implement. Thus, in the study, we fix our projection method $\mathcal{P}$ as PCA.

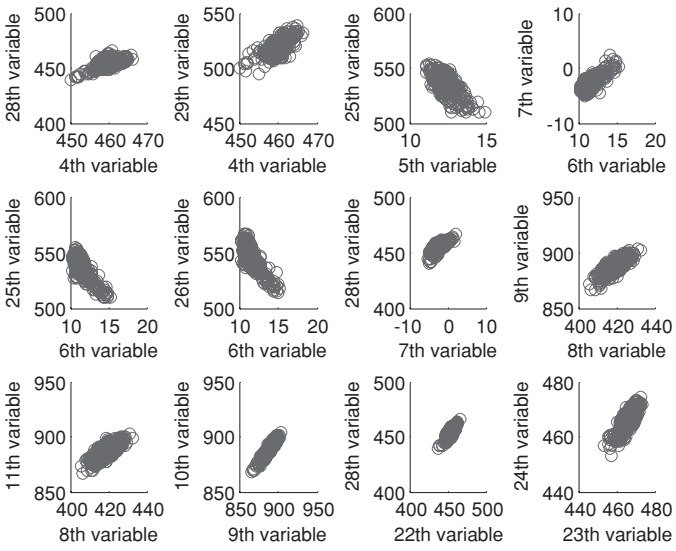

Fig. 3. Scatterplots of several pairs of the 34 explanatory variables in the warranty data set. Strong collinearity is observed.

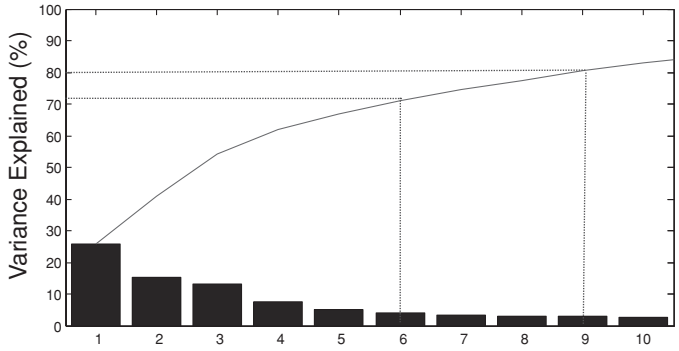

Fig. 4. Scree plot for warranty data after standardizing the original data. The first six to nine PCs can explain in a range from 70% to 80% of the variability in the data. The variation contribution for the tenth PC and onward levels off.

One area needing extra attention is that when the scales of the original variables are widely different, it is commonly recommended to standardize the original variables first, before applying PCA, in order to alleviate the adverse effect of different scales associated with the original variables.

After applying PCA to the original data matrix $\mathbf{X} = (x_{ij}), i = 1, \ldots, N, j = 1, \ldots, p$, $\mathbf{X}$ can be decomposed into the principal components and the residual noise as

$$\begin{aligned} \mathbf{X} &= \mathbf{Z} \times \mathbf{P}^{\mathrm{T}} \\ &= \mathbf{Z}_g \times \mathbf{P}_g^{\mathrm{T}} + \mathbf{X}_{g+1}, \end{aligned} \tag{19}$$

where $\mathbf{Z}$ and $\mathbf{P}$ are the loadings and scores matrices, respectively. Also, $\mathbf{Z}_g$ and $\mathbf{P}_g$ contain the g columns of $\mathbf{Z}$ and $\mathbf{P}$ corresponding to the largest g eigenvalues and form the Principal Components (PCs) portion; $\mathbf{X}_{g+1}$ is the residual noise. The number of PCs g represents the reduced dimension and is usually decided in such a way that the term $\mathbf{Z}_g \times \mathbf{P}_g^T$ explains a big portion of the total variance of $\mathbf{X}$, so that $\mathbf{X}_{g+1}$ is small and behaves like noise.

Johnson and Wichern (2002) suggested using a scree plot to decide the proper value of g. Figure 4 shows the scree plot after standardizing the original warranty data. One can see that the variance explained by the first six eigenvalues is more noticeable, whereas the variation contribution for the tenth PC and onward levels off. The first six to nine PCs can explain in a range from 70% to 80% of the variability in the data, which leads us to believe that the set of the first six to nine PCs would make a suitable set of predictors for the subsequent prediction.

It is not very easy to pinpoint the exact number of PCs to be included for prediction because the increase in the explained variance rises rather gradually from 70% to 80%. However, oftentimes, it is much easier to decide the set G that contains the suitable choices for g. For the example explained above, a good choice of G is that $G = \{6, 7, 8, 9\}$. As such, we choose to use the scree plot to decide G and let the optimization in Equation (10) solve for the best choice of $g \in G$.

3.3. *Resampling*

Recall that the resampling ratio, $r = n_0/n_1$, represents the weights assigned to each class. We need to specify the set R from which r is chosen.

If r is too big (that is, $n_0 \gg n_1$), it puts a lot more weight on the majority class than on the minority class and we are likely to get low detection power, just like with the original data set. On the contrary, too small an r (that is, $n_0 \ll n_1$) could result in a high false alarm rate, though we might end up with an improved detection power. We recommend that the range of ratios be as broad as possible to fully examine the solution space. For example, in the two data sets we studied, we let the set R contain a wide range of resampling ratios, from 20/80 to 80/20 with an increment of ten for n_0 (or, equivalently, a decrement of ten for n_1). Specifically, $R = \{20/80, 30/70, 40/60, 50/50, 60/40, 70/30, 80/20\}$. We exclude the extreme ratios such as 10/90 and 90/10 because they did not give any useful information in the preliminary experiments. If a preliminary experiment shows that the range does not have to be this wide, then we can narrow the range further for computational benefit. Also, the step size can be varied with different applications. If the out-of-bag estimates between neighboring ratios are quite different, we suggest decreasing the step size.

We fix the total number (that is, $n_0 + n_1$) of data records in each training set to be 100 in this study because through a preliminary study using a number of different data sizes, we found that the training set size did not make much difference in the eventual prediction (Section 5.9 summarizes the results from different sizes of training sets). Rather, the resampling ratio matters a lot. We also want to use a much smaller size than the size of the whole training set $S^{(l)}$ to expedite the process to build each base classifier. Certainly 100 is chosen out of convenience. This choice makes sense since we work with the data after dimension reduction, where predictors have fewer than ten dimensions. If the data reduction cannot reduce the data to be in a sufficiently smaller subspace, then this fixed data size should be increased or be considered as a decision variable in the optimization for better performance.

3.4. *Constructing a base classifier using SVM*

Conceptually, any classification method can be used to construct a base classifier. In this study, we use SVM because of its flexibility and strong performance on many learning problems. Taking the PCs as the predictors, the SVM classifier is formulated as follows (Hastie *et al.*, 2003):

$$f(\mathbf{z}) = \beta_0 + \sum_{i=1}^{n_0+n_1} \beta_i y_i K(\mathbf{z}_i, \mathbf{z}), \tag{20}$$

where $\mathbf{z}_i = [z_{i1}, z_{i2}, \ldots, z_{ig}]$ is the ith record in $\mathbf{Z}_g$ of Equation (19), $\mathbf{z} = [z_1, z_2, \ldots, z_g]$ is the future unit to

be classified, $K : \mathbb{R}^g \times \mathbb{R}^g \to \mathbb{R}$ is a kernel function, and $\beta_i, \beta_0 \in \mathbb{R}$ are parameters to be decided for the given training set. When a training data set is available, $\{(y_i, \mathbf{z}_i), i = 1, \ldots, (n_0 + n_1)\}$ would be known. Once a kernel function is chosen, the parameters β_i, β_0 can be estimated using $\{(y_i, \mathbf{z}_i), i = 1, \ldots, (n_0 + n_1)\}$.

A few software packages come to aid in establishing a SVM classifier. For example, one can use the `MATLAB` function *svmtrain( )* for building a SVM classifier (using the training data set $\{(y_i, \mathbf{z}_i), i = 1, \ldots, (n_0 + n_1)\}$, and function *svmclassify( )* for classifying a newly observed item. When the `MATLAB` function *svmclassify( )* yields one (equivalent to saying $f(\mathbf{z}) = 1$), the item is classified to the minority class; otherwise, it is classified to the majority class. Likewise, in the software `R`, *svm( )* and *predict.svm( )* functions in package *e1071* can be used to build a SVM classifier and make predictions, respectively (Dimitriadou *et al.*, 2008; R Development Core Team, 2008).

The `R` routine *svm( )* is more flexible than the MATLAB counterpart; for example, it can yield the probability of a future item belonging to a certain class instead of simply outputting a class label. When a SVM classifier is included as a base classifier in our classification method, we simply let it output a class label (namely zero or one). However, when a SVM is used as one of the off-the-shelf methods for performance comparison (in Section 5), we let it output the probability of belonging to a class. This is because outputting 0-1 class labels assumes a default setting of equal cost for the two types of misclassification errors (misdetection and false alarm); and as such, the SVM, functioning as a stand-alone classification method, will perform poorly for class-imbalanced data. Choosing the option of letting the SVM produce a probability will allow us to adjust the cut-off value in classification (equivalent to assigning different weights to different classes) so that the SVM's detection capability can be enhanced. Regarding weight assignment to the data, more discussions are available in Section 5.2.

In using a SVM, we need to carefully choose the kernel function. The consensus in the statistics community is that no specific kernel function can always outperform other kernel functions. Thus, we treat the type of kernel function as the classifier parameter θ, that is, one of the decision variables in the optimization (10). We examine four different types of kernel functions that are commonly used, which are linear, quadratic, polynomial and radial basis functions. That is to say, the set $\Theta = \{\text{linear, quadratic, polynomial, radial}\}$.

3.5. *Threshold to regulate the FA rate*

In this study, we set the threshold to be 10% for the warranty data, after consultation with our industrial partners, which aims at narrowing down the size of the screened data set to be roughly one-tenth (or 10%) of the original size. We choose the same threshold for the abalone data. Of course, the threshold value can be different in other applications depending on the relative cost of false alarms.

3.6. *Summary of implementation procedure*

The proposed procedure can be summarized as follows.

Step 1. (Data standardization) Standardize the data set when the scales of explanatory variables are widely different.

Step 2. (Data reduction) Apply PCA to the original training data S, and find the range of PCs (G) to be investigated.

Step 3. (Partitioning of data set) Form the test data set S^t and the complementary set $S^{(l)} = S - S^t$.

Step 4. (Constructing ensemble classifiers) Repeat the following for each combination of decision variables $\{g \in G, r \in R, \theta \in \Theta\}$:

- 4.1. For $q = 1, \ldots, Q$, build the qth training set T_q from $S^{(l)}$ and fit a SVM to create a base classifier $\hat{C}(x, T_q; g, r, \theta)$.
- 4.2. Construct the ensemble classifier $\hat{C}_{\mathrm{A}}(x, T_q; g, r, \theta)$ by taking the majority voting of base classifiers.
- 4.3. Calculate the out-of-bag estimates.

Step 5. (Selection of decision variables) Find the best combination of decision variables $\{g^*, r^*, \theta^*\}$ which achieves the highest estimated detection power while yielding a false alarm rate estimate less than the given threshold α.

4. Data sets

To evaluate the performance of the proposed procedure, we use two data sets from two different applications previously mentioned in Section 1. The first data set is a warranty data set from a major cellphone manufacturer. The second data set is an abalone data set that was obtained from the UCI data repository (Blake and Merz, 2008). Table 1 summarizes the characteristics of these data sets.

4.1. *Warranty data*

In the warranty data set, there are a total of $N = 11\,899$ records with $p = 35$ explanatory variables and one field return variable specifying the faulty phones. The first five explanatory variables provide information regarding the lot name, the wafer ID, the position on the wafer (Diex, Diey) and the test site. The physical meanings of the other 30 variables are encrypted due to confidentiality concerns. The response variable y_i, $i = 1, \ldots, N$, is the field return indicator. If y_i is zero, it means no field return and this phone belongs to the *normal class* (or *majority class*). If y_i is one, it means that this phone has been returned by

Table 1. Summary of data sets used in the study

Data set	*Number of records*	*Number in majority class*	*Number in minority class*	*Class distribution (%: %)*	*Number of covariates*
Warranty[a]	11 899	11 889	10	99.92:0.08	34
Lot A	4470	4466	4	99.91:0.09	
Training $S^{(l)}$	4369	4366	3	99.93:0.07	
Testing S^{l}	101	100	1		
Lot B	7429	7423	6	99.92:0.08	
Training $S^{(l)}$	7328	7323	5	99.93:0.07	
Testing S^{l}	101	100	1		
Abalone	4177	4141	36	99.14:0.86	8
Training $S^{(l)}$	4061	4041	20	99.51:0.49	
Testing S^{l}	116	100	16		

[a]The warranty data set consists of two data sets from two lots. The records from the two lots are analyzed separately.

a customer and represents a faulty unit due to customer dissatisfaction. Thus, this phone belongs to the *faulty class* (or *minority class*). In the entire data set, only ten units (0.08%) belong to the *faulty class*. The production units in the study are manufactured in two lots, Lot A and Lot B. In the semiconductor industry, a lot is a group of wafers treated in a batch. Usually, the lot information plays an important role in yield analysis. The records from the two lots are thus analyzed separately as each lot may have distinct characteristics. As such, the *lot name* is no longer an explanatory variable so that p reduces to 34, as reported in Table 1.

In order to assess the predictive performance of the proposed method (including searching for the best decision variables), we generate multiple testing sets as follows. Each testing set, S^{l}, contains 100 normal unit records and one faulty unit record. The reason that we keep only one faulty unit for testing is due to the lack of faulty class data in the original data set. Since we intend to establish a prediction model each for Lot A and Lot B, the testing sets are also constructed separately for both lots. As shown in Table 1, Lot A has four faulty unit records among a total of 4470 records and Lot B has six faulty units among a total of 7429 records. For Lot A, each testing set is generated by including one faulty unit, and 100 normal units that are randomly sampled from the $(4470 - 4) = 4466$ normal units in that lot. This gives four testing data sets. Likewise, we can form six testing data sets for Lot B. Totally we make 24 $(= 4 \times 6)$ testing sets, which leaves 24 complementary data sets $S^{(l)}$ $(= S - S^{l})$, $l = 1, 2, \ldots, 24$, for model fitting as well as out-of-bag estimation.

4.2. *Abalone data*

We can know precisely the age of an abalone by cutting the shell through the cone, staining it and counting the number of rings through a microscope. However, this task requires lots of time and effort. Thus, we want to predict the age of an abalone from physical measurements, which are easier to obtain. The measurements include sex (male, female or infant) and seven other appearance measurements such as length, diameter and height.

The original response variable of this data set is the age of abalone, ranging from one to 29, which is used for further dividing the whole data set into two classes. The young-age class (majority class) consists of abalone samples whose ages are less than or equal to 20, while the old-age class (minority class) contains the samples whose ages are older than 20. As a result, there are 36 (0.86%) samples out of the total 4177 samples in the minority class. Among these 36 minority samples, we use 20 samples for training while the left-out 16 samples are used for testing. In a way similar to what we did in the warranty data, we generate four different testing sets. Each testing set, S^{l}, $l = 1, \ldots, 4$, consists of 100 randomly chosen majority samples and 16 minority samples. These 16 minority samples are carefully chosen so that the duplicates of minority samples among the testing sets can be minimized. For example, one testing set has the first 16 minority samples among the total 36 minority samples. The second testing set has the last 16 samples. The third one has the odd-numbered samples: that is, 1st, 3rd, 5th, ..., 31st. The last testing set has the even-numbered samples likewise.

5. Results

5.1. *Deciding the values of decision variables*

Because G is decided by using a scree plot, they are different for the two data sets. We choose the set G so that the range of PCs can explain 70% to 80% variability in the original data. For these two data sets, the scree plots help choose $G = \{6, 7, 8, 9\}$ for the warranty data and $G = \{4, 5\}$ for the abalone data. Resampling ratio set R and kernel function set Θ are the same for both data sets, as described in Section 3.3 and Section 3.4, respectively.

Table 2. Example: out-of-bag estimates (unit: percentage)[a,b]

Function	*g\r*	*20/80*	*30/70*	*40/60*	*50/50*	*60/40*	*70/30*	*80/20*
Linear kernel function	6	80, 17.9	80, 16.1	80, 15.1	80, 13.6	60, 12.1	60, 10.3	60, 7.4
	7	80, 18.1	80, 15.9	80, 14.8	80, 13.6	80, 12.1	60, 10.3	40, 7.6
	8	80, 16.8	80, 15.3	80, 13.8	80, 12.7	80, 11.4	60, 9.5	40, 7.1
	9	80, 16.9	80, 14.8	80, 13.5	80, 12.4	60, 11.0	60, 9.5	40, 6.9
Polynomial kernel function with order = 3	6	80, 5.3	80, 4.0	80, 3.4	60, 3.0	60, 2.8	60, 2.7	60, 2.4
	7	**80, 8.8**[c]	80, 4.4	80, 3.5	60, 3.3	60, 3.0	60, 2.8	60, 2.6
	8	80, 5.9	80, 2.7	80, 2.2	80, 2.0	80, 1.9	80, 1.9	60, 1.8
	9	80, 7.9	80, 3.2	80, 2.4	80, 2.1	80, 2.0	80, 1.9	60, 1.8

[a]Values in each cell are the detection power and the false alarm rate, respectively.
[b]The cases with quadratic and radial basis kernel functions are omitted to save space.
[c]The out-of-bag estimates corresponding to the prediction model chosen by the optimization procedure.

To illustrate the optimization procedure, Table 2 shows the out-of-bag estimates for detection power and false alarm rate using one test set of Lot B in the warranty data set. Each cell in the table contains the detection power and false alarm values (both in percentage) for the specific combination of decision variables corresponding to that cell. Only the data corresponding to two kernel function types (that is, θ = "Linear" or "Polynomial") are displayed in order to save space; using the other two kernel function types does not generate any additional understanding. Recall that our optimization policy is to look for the cases which have the highest detection power among those whose false alarm rate is less than 10%. It turns out that there are many variable combinations satisfying this criterion. For instance, in the lower part of Table 2, the combinations of $g = 6$, 7, 8, or 9, $r = 20/80$, 30/70 or 40/60; and $g = 8$ or 9 and $r = 50/50$, 60/40 or 70/30. Among all the combinations of the decision variables that attains the highest *DP*, we choose the one with the highest *FA*, which is the combination of θ = polynomial kernel function (order = 3), $g = 7$ and $r = n_0/n_1 = 20/80$.

One may also notice that the detection power and false alarm rate get monotonically increased when the resampling ratio r is decreasing (note that the 20/80 is considered the lowest resampling ratio and 80/20 the highest). This is expected because low resampling ratio gives more weight to the minority class data and doing so makes it more likely to predict a minority unit. However, the effect of either the number of PCs, g, or the type of SVM kernel function θ is not so obvious.

5.2. *Off-the-shelf classification methods*

To compare the prediction accuracy of ECHI with the off-the-shelf classification methods, we investigate several existing methods including SVM, MART, neural network, random forest and logistic regression by applying them to the same training/test sets. We next elaborate some of the details.

In the case of SVM, we use all four kernel functions that were used in our method. For the radial basis kernel function, a fine tuning of the relevant parameters such as kernel width is determined by a five-fold cross-validation (Hastie *et al.*, 2003). The five-fold cross-validation is also used in selecting parameters for MART. For random forest, 1000 trees are built in the forest, and its model parameter, which is the number of variables randomly chosen at each split, is chosen in a way such that the out-of-bag estimate of prediction error is minimized. In the case of the neural network, a single hidden layer neural network with 100 units is fitted, and a decay parameter (that is, a regularization parameter) of 0.01 is used to avoid overfitting.

All of these off-the-shelf methods give the probability of a unit belonging to a certain class. One would need to decide a cut-off value (denoted by π_0) to classify each unit as either majority or minority. In other words, a unit will be classified as a minority class unit if the probability of belonging to this class is over π_0, otherwise the unit belongs to the majority class. $\pi_0 = 0.5$ is the default choice as the cut-off value for the aforementioned methods, meaning that a classifier has an equal tendency to place a unit in either class. When using this default choice, all these existing methods yield a zero detection for the warranty data. Most of them also turn out a zero detection for the abalone data, with two exceptions (random forest: $DP = 15.6\%$, and logistic regression: $DP = 4.7\%$).

In order to level the playground for comparison, the false alarm rate can be elevated for these off-the-shelf methods in order to enhance their detection power. One way is to vary the cut-off value π_0 from 0.5 with a decrement of 0.01. Another way to enhance the detection power is to assign a much greater weight to the minority class. For this, we assign the weight 1/class size, which is inversely proportionally to the data amount in the respective classes, to each data record so that the minority class receives a large weight.

Tables 3 and 4 show the classification results when random forest is applied to one of the test sets in the abalone data set; the understanding generated here also holds when

Table 3. Detection powers of random forest when different cut-off values and weights are applied (unit: percentage)

Cut-off	*0.01*	*0.02*	*0.04*	*0.06*	*0.1*	*0.3*	*0.5*
No weight	56.3	50	50	37.5	12.5	0	0
Weight	87.5	75	50	31.3	18.8	0	0

other test sets are used. Not surprisingly, when a weight is assigned or the cut-off value is lowered, the random forest yields a higher detection power; in the meanwhile, the false alarm rate also increases. Even in weighted cases, the default setting of the cut-off value, $\pi_0 = 0.5$, cannot detect any minority units. In order to get *DP* comparable to our method, together with using the weight, we also have to decrease the cut-off value down to a substantially low value, specifically, 0.01 and 0.02. However, the false alarm rate is well above the 10% threshold. Thus, we choose the combination of $\pi_0 = 0.01$ and no weight, following the same optimization policy used in our method, i.e., the one with the highest detection power and false alarm rate less than 10%.

5.3. *One-class classification algorithms*

Another method we compare our procedure with is the one-class classification algorithms. One-class classification is also known as *novelty detection* (Markou and Singh, 2003). Novelty detection is to identify new or unknown class data when a classifier is trained only with one-class data. Many novelty detection algorithms have been introduced mainly to address the cases where test data (or future data) contains information about samples that are not available during training. However, they can be also used to address class imbalance problems (Raskutti and Kowalczyk, 2004). Several different algorithms have been introduced in to the novelty detection approach. An insightful review of these algorithms is provided by Markou and Singh (2003).

Among many algorithms, we implement the novelty detection approach based on SVM. This is because we use the SVM classifier as our base classifier in implementing ECHI. Also, SVM-based novelty detection is known to be flexible and have good prediction power in many applications (Hayton *et al.*, 2001; Ratle *et al.*, 2007). We first use the majority class data for training a classifier, and then use test sets from both minority and majority classes to evaluate the performance of the constructed classifier. We also apply PCA to improve the prediction accuracy. Furthermore, to investigate whether ensembling can benefit the performance of the novelty detection method, we generate 99 new training sets by bootstrapping from the majority class data in the original training set. The size of each new training set is equal to the original training set. Then, the majority vote from the 99 base one-class classifiers is used to classify the units in the testing data sets.

The results of two one-class classifier cases—novelty detection with and without ensembling—are summarized in the next section with the results from ECHI and other off-the-shelf two-class classification methods.

Table 4. False alarm rates of random forest when different cut-off values and weights are applied (unit: percentage)

Cut-off	*0.01*	*0.02*	*0.04*	*0.06*	*0.1*	*0.3*	*0.5*
No weight	8	5	3	1	0	0	0
Weight	48	36	4	4	1	0	0

5.4. *Test results*

Tables 5 and 6 summarize the comparison results of our proposed method ECHI with the off-the-shelf methods, on the same sets of test data. Please note that the performance results of ECHI shown in Tables 5 and 6 are the testing results, so they are different from the out-of-bag estimates shown in Table 2.

For the warranty data, the 24 testing sets are used to test the selected prediction models. The averages of their performances are reported in the table. The proposed method gives quite impressive results. The detection power is 75%, while the false alarm rate is regulated at 9.0%. The standard deviation of detection power of ECHI is relatively high (25.5%) but so is that for the other methods having a non-zero detection power except novelty detection. We believe that this relatively large variability in *DP* has resulted because the very limited number of faulty units in the original data set allowed us to have only two faulty units in each test set. On the other hand, low (or even zero) variability in *DP* achieved by novelty detection would be explained by the facts that most of the majority units are common to the different training sets, resulting in similarly behaving classifiers, that to test detection power of novelty detection

Table 5. Testing results on detection power (standard deviation in parenthesis) (unit: percentage)

Method	*Warranty*	*Abalone*
Proposed method, ECHI	75.0 (25.5)	73.4 (2.7)
SVM (linear kernel)	12.5 (22.1)	46.9 (8.1)
SVM (quadratic kernel)	0 (0)	0 (0)
SVM (polynomial kernel)	0 (0)	45.3 (7.9)
SVM (radial basis kernel)	0 (0)	0 (0)
Random forest	25.4 (25.8)	59.4 (14.9)
MART	0 (0)	29.7 (18.9)
Neural network	37.5 (22.1)	35.9 (35.5)
Logistic regression	0 (0)	0 (0)
Novelty detection without ensembling	20.0 (0)	11.1 (4.0)
Novelty detection with ensembling	26.0 (5.8)	9.7 (2.8)

Table 6. Testing results on false alarm rate (standard deviation in parenthesis) (unit: percentage)

Method	*Warranty*	*Abalone*
Proposed method, ECHI	9.0 (1.6)	9.8 (2.2)
SVM (quadratic kernel)	0.3 (0.4)	8 (2.9)
SVM (quadratic kernel)	0 (0)	0 (0)
SVM (polynomial kernel)	0 (0)	9.8 (1.5)
SVM (radial basis kernel)	0 (0)	0 (0)
Random forest	0.5 (0.9)	3.3 (1.7)
MART	0.3 (0.6)	1.5 (1.0)
Neural network	1.1 (0.8)	3.0 (2.9)
Logistic regression	0 (0)	0 (0.3)
Novelty detection without ensembling	3.7 (1.9)	6.5 (1.3)
Novelty detection with ensembling	6.5 (2.0)	2.5 (1.3)

we always use the same set of minority units (as only the majority units are used in the training stage). Nonetheless, the average detection power of ECHI is far greater than the alternatives. Half of the existing methods cannot detect even a single faulty unit.

For the abalone data, ECHI also shows strong performance with 73.4% detection power and 9.8% false alarm rate. Most of the off-the-shelf methods achieved a much lower detection power (less than 50%) than ours. The only case with a decent detection power is the random forest ($DP = 59.4\%$) when we tune its cut-off value very low. However, the standard deviation is high (14.9%) compared to that of ECHI (2.7%), which means that the result from random forest is much less stable in this class-imbalance classification problem. Moreover, as shown in Tables 3 and 4, the price that would have been paid for further improving a random forest's detection power is quite stiff—when its *DP* becomes 75%, comparable to our method, its *FA* will elevate to 36%, much higher than the 10% tolerance line.

It is interesting to see that the novelty detection methods do not perform well in either of the two data sets. Even when we add the ensembling component, prediction accuracy is not significantly improved, both in terms of *DP* and *FA*. This low prediction power could be because novelty detection only uses a single class of data (in our case, majority class data) in the training stage while ignoring potentially valuable information embedded in the other class data (in our case, minority class data). For this reason, if the performance of two-class classification methods can be considerably improved, as in our method, then the two-class classification methods, having the advantage of utilizing all the information in the historical data sets, can outperform the novelty detection approaches.

From the examples, one may observe that the more imbalanced the class distribution is the clearer is the advantage of using our method. In the abalone data set with 0.5% minority class units in the training data set, some methods achieve moderate detection. Particularly, the random forest method is a valid alternative in that application. However, none of these methods works well for the warranty data where the class imbalance is much more severe. By contrast, ECHI shows similar performances in these two data sets.

The practical implication of the results of ECHI is that when applied to the original warranty data, it will keep roughly 1100 data records (that is, 11 889 $\times$ 9.0% plus the faulty units), among which there are about eight faulty units. This subset of cellphone units will be handled more diligently with a more thorough testing procedure (with human operators involved from time to time) to identify the faulty units correctly. Because this subset is much smaller than the original set, doing so will not drastically slow down the production process. A similar story goes with the abalone data. After the screening, the subset has roughly 440 abalone samples with 26 old-age ones. Then, the cutting–staining–counting manual procedure can be used on these 440 samples to determine precisely the old-age samples. Again, it will lead to a great reduction in time and effort for such a biological investigation.

5.5. *Out-of-bag estimation*

Figure 5 compares the out-of-bag estimates of the detection power and false alarm rate of ECHI with the results from using the test sets. One can observe that the out-of-bag estimates did a good job in reflecting the actual test performances in both data sets. This testifies the merit of our recommended optimization procedure.

5.6. *Ensembling effect*

In this subsection, we would like to elaborate the benefit of having an ensemble classifier by quantifying the improvements in prediction performance. The aggregation effect and the variance of a base classifier, as defined in Equations (15) to (18), indicate the advantage of ensembling over simply using a base classifier.

The aggregation effect $AE(C)$ can be obtained by taking the difference in prediction error between the base classifier and an ensemble classifier. The prediction error of the base classifier is given by

$$PE(\hat{C}) = \frac{1}{N_0^{S^l} + N_1^{S^l}} \sum_{i=1}^{N_0^{S^l}+N_1^{S^l}} \left(\sum_{q=1}^{Q} L(y_i, \hat{C}(\mathbf{x}_i, T_q)) \right) \Big/ Q, \tag{21}$$

where $N_0^{S^l}$, $N_1^{S^l}$ are the number of majority class units and the minority class units in a test set S^l, respectively. When we use multiple test sets, we average the prediction errors from individual test sets. Also, the prediction error for the

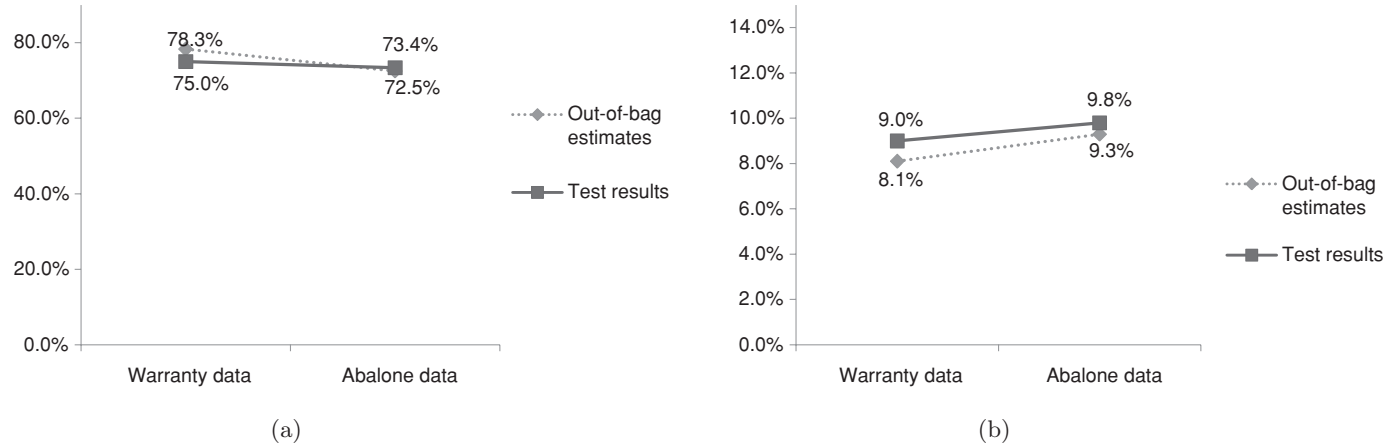

Fig. 5. Comparison of out-of-bag estimates with test set results: (a) detection power; (b) false alarm rate.

ensemble classifier is given by

$$PE(\hat{C}_{\mathrm{A}}) = \frac{1}{N_0^{S^l} + N_1^{S^l}} \sum_{i=1}^{N_0^{S^l}+N_1^{S^l}} L(y_i, \hat{C}_{\mathrm{A}}(\mathbf{x}_i)). \quad (22)$$

Then, the aggregation effect is

$$AE = PE(\hat{C}) - PE(\hat{C}_{\mathrm{A}}). \quad (23)$$

AE_{DP} and AE_{FA} can be likewise obtained. Please note that when we define the aggregation effect of detection power in Equation (16), the order of difference between the ensemble classifier and the base classifier is changed from the original definition of aggregation effect. This is done to ensure that a positive value of AE consistently indicates the improved prediction accuracy of the ensemble classifier over the base classifier.

The first three columns of Table 7 summarize the ensembling effects for the two data sets. The results show a noticeable increase in prediction power when an ensemble classifier is used. For example, in the warranty data, the ensemble classifier classifies the faulty units 11.2%, and the normal units 3.4%, more correctly than does a base classifier. It is interesting to note that the improvement of an ensemble classifier over the base classifiers is more significant in its detection power than in its false alarm rate.

Variance of the base classifier, defined in Equation (18), is the rate at which the base classifier classifies differently from an ensemble classifier, when using the 0/1 loss function. If the resulting variance is high, it indicates that the base classifier is unstable. The variance of the base classifier can be obtained by

$$\mathrm{Var}(\hat{C}) = \frac{1}{N_0^{S^l}+N_1^{S^l}} \sum_{i=1}^{N_0^{S^l}+N_1^{S^l}} \left(\sum_{q=1}^{Q} L(\hat{C}_{\mathrm{A}}(\mathbf{x}_i), \hat{C}(\mathbf{x}_i, T_q)) \right) / Q. \quad (24)$$

The variance for the detection power and false alarm rate (Var_{DP}, Var_{FA}) can be obtained in a similar way. The last three columns of Table 7 show the variance of the base classifier. One can observe that between 14% and 16% predictions of the base classifier are different from the aggregated results from the ensemble classifiers in detecting the minority units. The base classifier also shows considerable variance in false alarm rate. It is fair to conclude that a base classifier alone is not highly suitable for generating a stable enough prediction for these class-imbalance data sets.

Table 7. Ensembling effect (unit: percentage)

	Aggregation effect			*Variance of base classifier*	
	Warranty	*Abalone*		*Warranty*	*Abalone*
$DP\,(AE_{DP})$	11.2	6.1	$DP\,(\mathrm{Var}_{DP})$	14.0	15.6
$FA\,(AE_{FA})$	3.4	2.6	$FA\,(\mathrm{Var}_{FA})$	8.0	6.3

5.7. *Ensemble size*

The misclassification rates become stabilized as the ensemble size increases. Therefore, a big enough ensemble size

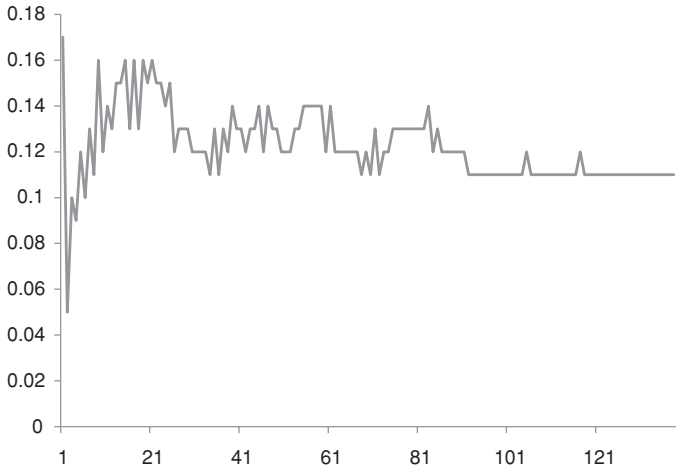

Fig. 6. Example of misclassification rates.

Table 8. PCA effect (unit: percentage)

	Detection power		*False alarm rate*	
	Warranty	*Abalone*	*Warranty*	*Abalone*
ECHI	75.0	73.4	9.0	9.8
ECHI without PCA	54.2	51.6	7.0	6.3

should be used to reduce variability in the prediction results. Figure 6 shows an example of how the misclassification rate varies with different ensemble sizes using one of the test data sets. One can see that the misclassification rate becomes stabilized with increasing ensemble size; in this particular case, roughly after $Q \geq 90$. Similar behavior is observed in other test sets as well. In our study, $Q = 99$ is used as the ensemble size, as this size is large enough to get stable results in both data sets.

5.8. *PCA effect*

In order to measure the effect of PCA, we apply the same procedure to both data sets without implementing PCA. Table 8 summarizes the results. PCA improves *DP* by 20.8%(= 75.0% − 54.2%) and 21.8%(= 73.4% − 51.6%) in the two data sets. It is interesting to see that the improvement in *DP* due to PCA in the moderate-dimensional problem, namely abalone data set, is similar to the high-dimensional data set. On the other hand, we have slightly lower *FA* when PCA is not applied. Overall, the results support the inclusion of PCA in the classificiation procedure.

5.9. *Sensitivity analysis*

In order to investigate whether the results are robust when the size of training sets varies, we implement the suggested procedure with three different training set sizes, $n_0 + n_1 =$ 50, 100, 200. Figure 7 summarizes the results. From the results, we believe when using reduced data with dimension less than ten, training data size of 100 is large enough. Choosing different training data sizes (especially, the larger ones) does not appear to affect the prediction accuracy significantly.

6. Summary

This paper presents a new ensemble classifier ECHI to improve the prediction power in high class-imbalance problems. The major challenge in analyzing these highly imbalanced data is that usually an extremely limited number of minority class units are mixed with an overwhelmingly large number of majority class units. We tackled several aspects we believe are critical to making a good prediction for a class-imbalance data set, including a data reduction component and both up- and down-sampling techniques when generating a training set. We formulate the classification problem as a constrained optimization problem and solve for the decision variables using a data-driven approach.

As is evident from the case studies using the two real-world problems with various features, degrees of imbalance and sizes, the prediction results show remarkable improvements over a number of popular off-the-shelf classification methods. ECHI is able to achieve about 75% detection power. False alarm rates are kept at bay as well. What these results imply is that with a pre-screening tool like this, the original data set will be shrunk into a data subset that is one-tenth the size of the original data while keeping three-quarters of the minority units. This result will help the subsequent investigation a great deal. When we applied the existing classification methods, including SVM, random forest, MART, neural network and logistic regression, to the same data set, most of them turned out zero detection, while a few others turned out low detection rates.

It is not hard to imagine that using a cost-sensitive classifier is another way of addressing the class-imbalance problems (Pazzani *et al.*, 1994; Domingos, 1999). A cost-sensitive classifier assigns a higher cost to a missed detection of the minority class than a false alarm of the majority class so as to enforce the preference of predicting the minority class. In Section 5.2, adding the weight assignment

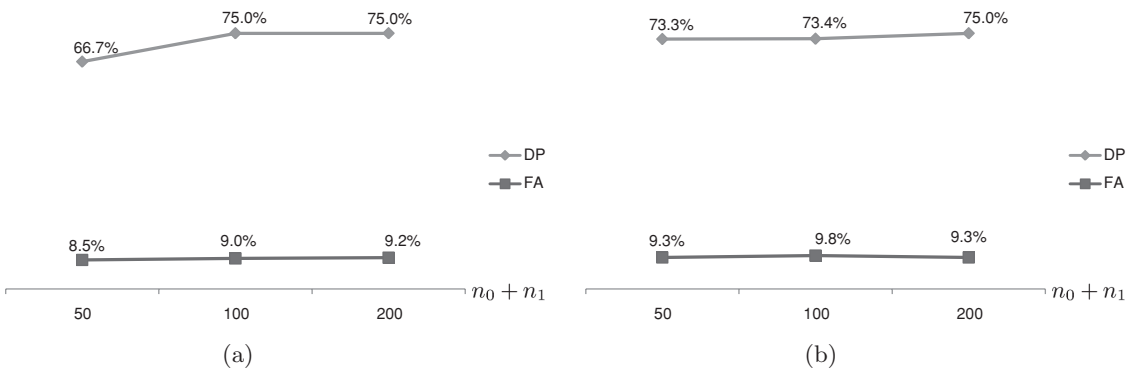

Fig. 7. Sensitivity analysis on the sizes of training sets: (a) warranty data and (b) abalone data.

to the data when applying the off-the-shelf methods is in fact constructing a cost-sensitive classifier. However, we observed that with merely the cost reassignment, the resulting classifier is not yet as capable as our proposed method. In ECHI, the resampling component is, to some extent, comparable to the idea of cost differentiation; the difference is that the resampling chooses to adjust the class sizes so that a cost differentiation directly assigned to the data is not needed. However, resampling is just one component in our procedure. For a procedure to work well for highly imbalanced data, all the components recommended by our research are critical and necessary.

If there are special underlying patterns such as temporal or spatial patterns in data sets, there may be a potential risk of losing information when we down-sample the majority class data and use a small number of majority units in each training set. For this type of data sets, one needs to make sure that the resampling should be frequent enough (or the resampling data should be large enough) for the underlying patterns to be preserved. The proposed method is better applicable to data sets with independent units, some of which are considered faulty or anomalous due to random events oftentimes observed as in discrete-part manufacturing, fraud detection, disease diagnosis and so on. Furthermore, we combine the resampling technique with ensembling approach, which is based on a set of resampled data sets. When the number of the base classifiers is rather large, ensembling would reduce the risk of down-sampling in the type of applications we referred to above.

We believe that the proposed method forms a sophisticated computational package which can also be applied to other class-imbalance problems that often appear in disease diagnosis (Chen *et al.*, 2005), fraud detection (Fawcett and Provost, 1997), image classification (Lee *et al.*, 2007) and security surveillance problems. A commonality of these problems is that it is more important to correctly detect the minor-class units (representing malignant cases or anomaly) than to classify the major-class units (representing the normality). Therefore, when a fully automatic classification procedure is not yet feasible, a pre-screening classification tool as we advocate in this paper should help; and the same treatment developed in this study should be applicable in improving the power of detecting the minority class. Despite the 75% detection power attained by the proposed method, there is still much room (and a great need as well) to further improve the detection capability via innovative future research.

References

Anderson, E.W. (1998) Customer satisfaction and word of mouth. *Journal of Service Research*, **1**(1), 5–17.

Blake, C.L. and Merz, C.J. (2008) Repository of machine learning databases. Available at http://archive.ics.uci.edu/ml/. Accessed 16 May 2008.

Breiman, L. (1996) Bias, variance and arcing classifiers. Technical report 460, Statistics Department, University of California, Berkeley, CA.

Chan, P.K. and Stolfo, S.J. (1998) Toward scalable learning with non-uniform class and cost distributions: A case study in credit card fraud detection, in *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining*, New York, NY, pp. 164–168.

Chen, J.J., Tsai, C.A., Young, J.F. and Kodell, R.L. (2005) Classification ensembles for unbalanced class sizes in predictive toxicology. *SAR and QSAR in Environmental Research*, **16**(6), 517–529.

Dimitriadou, E., Hornik, K., Leisch, F., Meyer, D. and Weingessel, A. (2008) e1071: Misc Functions of the Department of Statistics (e1071), TU Wien. R Foundation for Statistical Computing. R package version 1.5-18.

Domingos, P. (1999) Metacost: A general method for making classifiers cost-sensitive, in *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 155–164.

Fawcett, T. and Provost, F.J. (1997) Adaptive fraud detection. *Data Mining and Knowledge Discovery*, **1**(3), 291–316.

Hastie, T., Tibshirani, R. and Friedman, J. (2003) *The Elements of Statistical Learning*, third edition, Springer, New York, NY.

Hayton, P., Schölkopf, B., Tarassenko, L. and Anuzis, P. (2001) Support vector novelty detection applied to jet engine vibration spectra, in *Porceeding of the Conference on Advances in Neural Information Processing Systems 13*, MIT Press, Cambridge, MA, pp. 946–952.

Japkowicz, N., Myers, C. and Gluck, M.A. (1995) A novelty detection approach to classification, in *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, Montreal, Quebec, Canada, pp. 518–523.

Johnson, R.A. and Wichern, D.W. (2002) *Applied Multivariate Statistical Analysis*, fifth edition, Prentice-Hall, Englewood Cliffs, NJ.

Kubat, M., Holte, R.C. and Matwin, S. (1998) Machine learning for the detection of oil spills in satellite radar images. *Machine Learning*, **30**, 195–215.

Lee, K.-M., Li, Q. and Daley, W. (2007) Effects of classification methods on color-based feature detection with food processing applications. *IEEE Transactions on Automation Science and Engineering*, **4**(1), 423–439.

Linderman, K., Schroeder, R.G., Zaheer, S. and Choo, A.S. (2003) Six sigma: a goal-theoretic perspective. *Journal of Operations Management*, **21**, 193–203.

Mannar, K., Ceglarek, D., Niu, F. and Abifaraj, B. (2006) Fault region localization: product and process improvement based on field performance and manufacturing measurements. *IEEE Transactions on Automation Science and Engineering*, **3**(4), 423–439.

Markou, M. and Singh, S. (2003) Novelty detection: a review - part 1: statistical approaches. *Signal Processing*, **83**(12), 2481–2497.

Pazzani, M., Merz, C. Murphy, P. Ali, K., Hume, T. and Brunk, C. (1994) Reducing misclassification costs, in *Proceedings of the Eleventh International Conference on Machine Learning*, New Brunswick, NJ, pp. 518–523.

R Development Core Team (2008) *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria.

Raskutti, B. and Kowalczyk, A. (2004) Extreme rebalancing for SVMs: a case study. *SIGKDD Explorations*, **6**(1), 60–69.

Ratle, F., Terrettaz-Zufferey, A., Kanevski, M., Esseiva, P. and Ribaux, O. (2007) A comparison of one-class classifiers for novelty detection in forensic case data, in *Proceedings of the Eighth International Conference on Intelligent Data Engineering and Automated Learning*, Birmingham, UK, pp. 67–76.

Tibshirani, R. (1996) Bias, variance and prediction error for classification rules. Technical report, Statistics Department, University of Toronto, Toronto, Ontario, Canada.

West, D., Mangiameli, P., Rampal, R. and West, V. (2005) Ensemble strategies for a medical diagnostic decision support system: a breast cancer diagnostic application. *European Journal of Operations Research*, **162**, 532–551.

Westbrook, R.A. (1987) Product/consumption-based affective responses and postpurchase process. *Journal of Marketing Research*, **24**, 258–270.

Wezel, M.V. and Potharst, R. (2007) Improved customer choice predictions using ensemble methods. *European Journal of Operations Research*, **181**, 436–452.

Biographies

Eunshin Byon is a Ph.D. candidate in the Department of Industrial and Systems Engineering at Texas A&M University, College Station. She received her M.S. and B.S. (Honors) in Industrial and Systems Engineering from Korea Advanced Institute of Science and Technology (KAIST), South Korea. Her research interests include rare event detection, statistical modeling for complex systems, decision making under uncertainty and operations and maintenance of wind power systems. She is a member of IIE, INFORMS and IEEE.

Abhishek K. Shrivastava is an Assistant Professor in the Department of Manufacturing Engineering and Engineering Management at City University of Hong Kong, Hong Kong. His research interests are in design of experiments, data mining for rare events, spatio-temporal statistics, and the intersection of optimization and statistics. He received his B. Tech. in Industrial Engineering from the Indian Institute of Technology, Kharagpur, India in 2003 and his Ph.D. in Industrial Engineering from Texas A&M University, College Station, TX, USA, in 2009. He is a member of INFORMS, IIE, ASA and IMS.

Yu Ding received a B.S degree in Precision Engineering from the University of Science and Technology of China in 1993, M.S. in Precision Instruments from Tsinghua University, China in 1996, M.S. in Mechanical Engineering from the Pennsylvania State University in 1998 and Ph.D. in Mechanical Engineering from the University of Michigan in 2001. He is currently an Associate Professor in the Department of Industrial and Systems Engineering at Texas A&M University. His research interests are in the area of system informatics and quality and reliability engineering. He has received a number of awards for his work, including an *IIE Transactions* Best Paper Award in 2006, CAREER Award from the National Science Foundation in 2004 and a Best Paper Award from the ASME Manufacturing Engineering Division in 2000. He currently serves as a Department Editor of *IIE Transactions* and an Associate Editor of *IEEE Transactions on Automation Science and Engineering*. He is a member of IIE, INFORMS, IEEE and ASME.