ORIGINAL RESEARCH

Augmenting cost-SVM with gaussian mixture models for imbalanced classification

Miao He, Teresa Wu*, Alvin Silva, Dianna-Yue Zhao, Wei Qian

School of Computing, Informatics, Decision Systems Engineering, Arizona State University, United States

Received: May 10, 2015 **Accepted:** June 22, 2015 **Online Published:** July 3, 2015
DOI: 10.5430/air.v4n2p93 **URL:** http://dx.doi.org/10.5430/air.v4n2p93

Abstract

The Support Vector Machine (SVM), a known discriminative classifier is ineffective in dealing with imbalanced classification problems where the training examples of target class are outnumbered by non-target class examples. Though cost-SVM (cSVM) has been proposed to tackle the imbalanced datasets by assigning different cost functions to different classes, the performance is less than satisfactory due to its limited ability to enforce cost-sensitivity. In this research, a generative classifier, Gaussian Mixture Model (GMM) is studied which can learn the distribution of the imbalanced data to improve the discriminative power between imbalanced classes. By fusing this knowledge into cSVM, a model fusion approach, termed CSG (cSVM+GMM), is proposed to tackle the imbalanced classification problem. Experimental results on eleven benchmark datasets and one medical imaging dataset show the effectiveness of CSG in dealing with imbalanced classification problems.

Key Words: Imbalanced classification, Support vector machines, Gaussian mixture model, Supervised learning

1 Introduction

Classification is a supervised learning problem which identifies the labels of new observations given a training dataset. Based on the number of classes studied, there exists multiclass classification and binary classification. Multiclass classification is usually treated under the one-versus-one or one-versus-all framework[1] both of which use binary classifier as the base classifier. One of the most commonly used binary classifiers is support vector machine (SVM) developed by Vapnik *et al.* in 1995.[2] Extensive research has explored the performance of SVM and concludes that SVM outperforms many other conventional methods in classification. For example, Bazzani1 *et al.*[3] apply a SVM classifier to separate false signals from micro calcifications in digital mammograms. The result shows that the SVM achieves better/comparable performance than multi-layer perceptron (MLP)[4] and linear discriminant analysis (LDA).[5] Shon *et al.*[6] propose a SVM based classification method to tackle the internet anomaly detection and conclude that SVM outperforms the real-world employed Network Intrusion Detection Systems (NIDS),[7] just to name a few.

While promising, SVM is known to be ineffective in dealing with imbalanced datasets[8–10] where the minority class (named positive class in this paper) is greatly outnumbered by the majority class (negative class). Indeed, in many applications, minority class possesses higher misclassification cost than majority class. For example, in the field of medical diagnosis (diseased patients), fraud detection (true frauds), identifying the minority examples is more of interest. Unfortunately, the performance of the standard SVM on minority class labeling is less than satisfactory. This is because the SVM algorithm assumes balanced class distribution and as-

*Correspondence: Teresa Wu; Email: Teresa.Wu@asu.edu; Address: School of Computing, Informatics, Decision Systems Engineering, Arizona State University, United States.

signs same penalty considerations to both majority and minority classes in the training process. As a result, the class boundary of SVM skews towards the minority class leading to high false-negative rate.[11]

Due to the significance and the prevalence of imbalanced datasets, many researchers explore ways to extend SVM for imbalanced classification. In general, the extensions can be divided into two categories: data preprocessing approach and algorithmic approach. The data preprocessing approaches use different sampling techniques to alter the input data distribution to reduce the degree of class imbalance. The representative methods are: undersampling (US), oversampling (OS) and synthetic data generation method such as SMOTE.[12] The preprocessing approaches are usually combined with different classifiers to achieve classification. For instance, Akbani *et al.*[13] compare the performance of SMOTE-SVM and SMOTE-cSVM (cost SVM[8]) on imbalanced datasets. Instead of modifying the distribution of the input data, the algorithmic approaches modify SVM algorithm directly to make it less sensitive to class imbalance. Some examples of algorithmic methods are: boundary movement (BM-SVM)[14] which shifts the decision boundary by adjusting the threshold parameter of the standard SVM; kernel modification method[11, 14] which modifies the associated kernel matrix K; and cost sensitive SVM (cSVM)[8] which applies cost-sensitive learning in SVM training by assigning different costs to different classes. It is noted from the literatures[15–17] that cSVM method is promising in dealing with imbalanced classification problems. This is because in Bayes decision theory, the costs associated with false positives and false negatives are generally unequal. Taking cancer diagnosis as an example, if a cancer patient is diagnosed as non-cancer, the associated cost would be missing the best timing for treatment which can be life threatening. On the other hand, the associate cost is much less if a non-cancer patient is diagnosed as having cancer, in which case only follow-up tests are needed for confirmation. The unequalness of this false positive/false negative costs can be further aggravated by the class imbalance due to the limited number of target-class examples to learn. Therefore, classifier designed using cost sensitive algorithms (*e.g.* cSVM) may be a good choice in dealing with an imbalanced dataset.[16] However, many empirical studies[11, 16, 18] show that cSVM does not work well as expected. As explained by Wu *et al.*,[11] this is due to the fact that cSVM has limited ability to enforce cost-sensitivity. Specifically, cSVM assigns higher cost to the positive class in order to increase the influences of the positive support vectors. The impact of a support vector is directly reflected by the value of its coefficient. However, the cost function serves as the upper bound, rather than lower bound, of support vector coefficients according to the Karush Kuhn Tucker (KKT) conditions. Thus, increasing of the cost does not necessarily affect the coefficients. In addition, the overall influences from positive and negative support vectors are forced to be equal according to the KKT condition (see validation in Section 4.2). As a result, the increase of positive support vector coefficients will inevitably increase some negative support vector coefficients which may lead to the unsatisfactory classification performance.

To address these issues, many researchers propose ways to improve cSVM's. Masnadi-Shirazi *et al.*[16] replace the hinge lose function of cSVM with cost-sensitive hinge lose function to enforce cost-sensitivity. Akbani *et al.*[13] combine cSVM with SMOTE method to make the boundary well-defined. Brefeld *et al.*[19] use example dependent cost instead of class dependent cost to further enforce cost-sensitivity of cSVM. Note these extensions focus on the discriminative models only which are designed to classify positive and negative class examples directly based on the provided input data.[20] While being directed to classify the data, the potential contributions from the underlying knowledge of the input data (*e.g.*, distributions, clusters) may be ignored. Alternatively, generative models[20] study the probability distribution of the training data, and apply Bayes rules to obtain the posterior probability for classification. In addition, generative models can incorporate the domain knowledge of the training data, *i.e.* the prior knowledge about the interaction among the variables, the data clustering and the parameter's range of values into the classification process. The complementary nature of discriminative and generative models motivates us to take a model fusion approach, termed CSG, by integrating cSVM with one type of generative models, Gaussian mixture model (GMM) to tackle the imbalanced classification problem. GMM is chosen here because it is computationally inexpensive and has fewer subjective parameters to adjust.[21] In addition, probability outputs from cSVM and GMM enable us to develop a unified formulation for integration. To test the performance of CSG, we conduct the experiments on eleven KEEL benchmark datasets and one medical imaging dataset collected from Mayo Clinic, Arizona. Experimental results show that CSG is effective in dealing with imbalanced classification problems.

The rest of the paper is organized as follows: in Section 2 we discuss the related work. In Section 3 we describe the CSG algorithm in detail followed by the comparison experiments in Section 4. We conclude with the findings and future work in Section 5.

2 Related work

2.1 Data preprocessing approaches

The data preprocessing approaches use different sampling techniques to alter the size and distribution of the training data in order to reduce class imbalance. Some common data preprocessing methods used in imbalanced classification are: undersampling,[22] oversampling[22] and the synthetic minority oversampling technique (SMOTE).[12]

Undersampling and oversampling are designed to rebalance the training data in different ways: undersampling decreases the size of majority class, while oversampling increases the size of minority class. The problematic consequences thus are different.[23–25] Undersampling reduces the imbalanced ratio by randomly removing the majority examples and thus may lead to the loss of information about the majority class. Oversampling increases the size of the minority class by randomly duplicating the minority examples which may cause over fitting.[10] The synthetic data generation method SMOTE[12] increases the size of the minority class by generating artificial data which are convex combinations of the existing ones with its nearest neighbors, thus improves learning.

2.2 Algorithmic approaches

The algorithmic approaches augment the SVM formulation to make it more tolerate to the class imbalance. Based on the parameters to be adjusted, the algorithmic approaches are in general classified into three subcategories: boundary movement (BM-SVM)[14,26] kernel modification[11,14] and cost-SVM (cSVM).[8]

Let the decision function of SVM be:

$$sgn(f(x) = \sum_{i=1}^{n} y_i \alpha_i K(x, x_i) + b) \tag{1}$$

As seen in (1), there are three parameters which impact the formation of the classification boundary: b, K and α. BM-SVM method shifts the class boundary by adjusting b, the threshold of the standard SVM. In the cases the data is non-separable, where the expected modifications should be on both the separating hyperplane w and threshold b, BM-SVM may not be performed.[16] The kernel modification method, Kernel-boundary alignment (KBA) on the other hand, tackles the imbalanced learning problem by modifying the associated kernel matrix K. This method adjusts the class boundary by using the adaptive conformal transformation (ACT) method based on the consideration of the feature-space distance and class-imbalanced ratio, and reduces the imbalanced support-vector ratio by reducing the number of support vectors from the majority class. However, removing existing negative support vectors may lead to the loss of information of the majority class and thus may introduce new bias. The cSVM, proposed by Veropoulos,[8] assigns different cost functions which are used as upper bounds to constrain α (formulations are presented in Section 3.2). Since it assigns higher cost to the minority class than majority class, the skewed class boundary can be pushed away from the minority class thus the accuracy of minority classification is improved. Based on the Bayes decision theory, cSVM is supposed to be a promising method in dealing with imbalanced classification problems. Yet, a number of empirical studies[11,16,18] show cSVM does not always have expected performance. The reason, as discussed by Wu *et al.*,[11] is that cSVM has issues for enforcing cost-sensitivity. Though research proposes cost-sensitive hinge loss function into cSVM,[16] integrating SMOTE with cSVM[13] and employing example dependent cost in cSVM training process,[19] the focus has only been on discriminative models. In this research, we integrate cSVM with a generative model, GMM, which incorporates the data distribution information into the training process to tackle the imbalanced classification problem. The detail of our proposed CSG is explained in the following section.

3 Proposed algorithm: CSG

3.1 SVM basics

SVM finds the decision boundary by constructing the separation hyperplane with maximum margin between different classes. The data points closest to the hyperplane are called support vectors in the soft-margin formulation.[2]

$$\begin{aligned} &\min \frac{1}{2} w \cdot w + C \sum_{i=1}^{n} \xi_i \\ &\text{s.t. } y_i(w \cdot \Phi(x_i) + b) \geq 1 - \xi_i \\ &\xi_i \geq 0,\ i = 1, \cdots, n \end{aligned} \tag{2}$$

Finding the support vectors is the key issue for the SVM classifier. This is because the decision function (in (1)) of a new testing data x is calculated based on the similarity measurement (kernel function K) between x and all the existing support vectors. The coefficients for non-support vector data points are zero (α_i=0) in (1). This indicates that the non-support vector data points have no impact on classification of the new testing data x once the support vectors has been determined.

The performance of the SVM classifier mainly relies on the choice of kernel function and the tuning of various parameters in the kernel function. The kernel function K (x_i, x_j) is a similarity measure between the pair of data points x_i and x_j. The kernel method works by mapping the two data points from original input space (x_i and x_j) onto the high-dimensional feature space ($\varphi(x_i)$ and $\varphi(x_j)$). The kernel function is calculated by taking the inner product of the transformed data vector:

$$K(x_i, x_j) = \langle \Phi(x_i), \Phi(x_j) \rangle = e^{(-\gamma \|x_i - x_j\|^2)},\ \gamma > 0 \tag{3}$$

In this paper, we choose the most commonly used radial basis function (RBF) kernel (in (3)) for its good performance on various domain applications.[27]

The SVM algorithm predicts the label of a test example x by computing the sign function in (1). Instead of predicting the label, much research requires the posterior class probability $P(y|x)$. Platt[28] proposes a method to approximate the posterior probability by using

$$P_{A,B}(x) = P(Y = 1|X = x) = \frac{1}{1 + e^{(Af(x)+B)}} \quad (4)$$

where A and B are estimated by minimizing the negative log likelihood of training dataset (x_i, y_i):

$$(A^*, B^*) = \arg\max_{A,B} \sum_{i=1}^{n_y} \left(\frac{1+y_i}{2} \log(P_{A,B}(x_i')) + \frac{1-y_i}{2} \log(1 - P_{A,B}(x_i')) \right) \quad (5)$$

In our proposed method, we also use the probability outputs of cSVM to fuse with the GMM probabilities in order to benefit from both methods.

3.2 cSVM

In cSVM, the formulation is given as:

$$\begin{aligned} &\min \frac{1}{2} w \cdot w + C \left[C^+ \sum_{i|y_i=+1}^{n_+} \xi_i + C^- \sum_{i|y_i=-1}^{n_-} \xi_i \right]; \\ &\text{s.t. } y_i(w \cdot \Phi(x_i) + b) \geq 1 - \xi_i; \\ &\xi_i \geq 0, i = 1, \cdots, n \end{aligned} \quad (6)$$

The Lagrangian for the cSVM formulation is:

$$\begin{aligned} L_p = \frac{w^2}{2} + C \left[C^+ \sum_{i|y_i=+1}^{n_+} \xi_i + C^- \sum_{i|y_i=-1}^{n_-} \xi_i \right] \\ - \sum_{i=1}^{n} \alpha_1 [y_i(w \cdot x_i + b) - 1 + \xi_i] - \sum_{i=1}^{n} \mu_i \xi_i \end{aligned} \quad (7)$$

With the constraints on α_i as follows:

$$\begin{cases} 0 \leq \alpha_i \leq C^+ & if\, y_i = +1 \\ 0 \leq \alpha_i \leq C^- & if\, y_i = -1 \end{cases} \text{and} \sum_{i=1}^{n} \alpha_i y_i = 0 \quad (8)$$

cSVM assigns different cost functions C^+ and C^- to the positive and negative classes respectively. The unequal setting of cost functions will allow the class boundary to be skewed towards the class with higher costs. In cSVM, one can assign higher costs to the minority class examples to push the class boundary toward the majority class. Yet, cSVM suffers from two drawbacks: first, cSVM changes the upper bound (C^+, C^-) of the support vector coefficients α_i, instead of working on α_i directly. Thus, increasing of C^+ does not always guarantee a change of α_i. Second, the KKT condition $\sum_{i=1}^{n} \alpha_i y_i = 0$ (in (8)) imposes equal influences from positive/negative support vectors. As a result, the increase of some positive support vector coefficients will inevitably increase some coefficients of negative support vectors which may weaken the discriminative power in identifying the minority examples.

3.3 GMM basics

GMM is a generative model applied in many applications such as object classification and speech recognition.[29–32] Based on the training data, GMM models the probability density function of the feature vector x by using a mixture of weighted Gaussians.

$$P_{GMM}(x|y_i) = \sum_{m=1}^{M} c_{im} N(x, \mu_{im}, \sigma_{im}^2) \quad (9)$$

Where:

$$N(x, \mu_{im}, \sigma_{im}^2) = \frac{1}{(2\pi\sigma_{im}^2)^{\frac{d}{2}}} e^{\left(-\frac{1}{2}\frac{\|x-\mu_{im}\|^2}{\sigma_{im}^2}\right)} \quad (10)$$

c_{im}, μ_{im}, and σ_{im}^2 are the weight, mean and covariance of the m^{th} mixture for class i. M is the number of mixtures which should be defined by the user. The GMM method is an unsupervised method that only reflects the intra-class information. Given a training dataset with binary class labels $\{(x_1, y_1), \cdots, (x_n, y_n)\}, y \in \{-1, 1\}$, the data are separated into two groups according to their class label. Then the coefficients c_{im}, μ_{im}, and σ_{im}^2 for each mixture are computed using an Expectation Maximization (EM) algorithm.[33] The EM algorithm is an iterative method for finding the maximum likelihood function of the parameters. Starting from some initial estimate of parameters, the iteration alternates between E step and M step where in the E step, the algorithm evaluates the expectation of the log-likelihood using the current parameters; in the M step, it computes the new parameters to maximize the log-likelihood function found in the E step. The stopping criterion for the iterations could be either convergence to a local maxima, or the difference between two consecutive iterations is smaller than a small value.

Once the coefficients are obtained, Bayesian rules can be used to calculate the posterior class probability:

$$P(y_i|x) = P(y_i) \sum_{m} P(m|y_i) N(x|\mu_{im}, \sigma_{im}^2) \quad (11)$$

3.4 Proposed algorithm: CSG

In this research, we propose a model fusion based approach to integrate cSVM discriminative algorithm with GMM generative algorithm which is explained as follows.

Table 1: Notations used in CSG algorithm

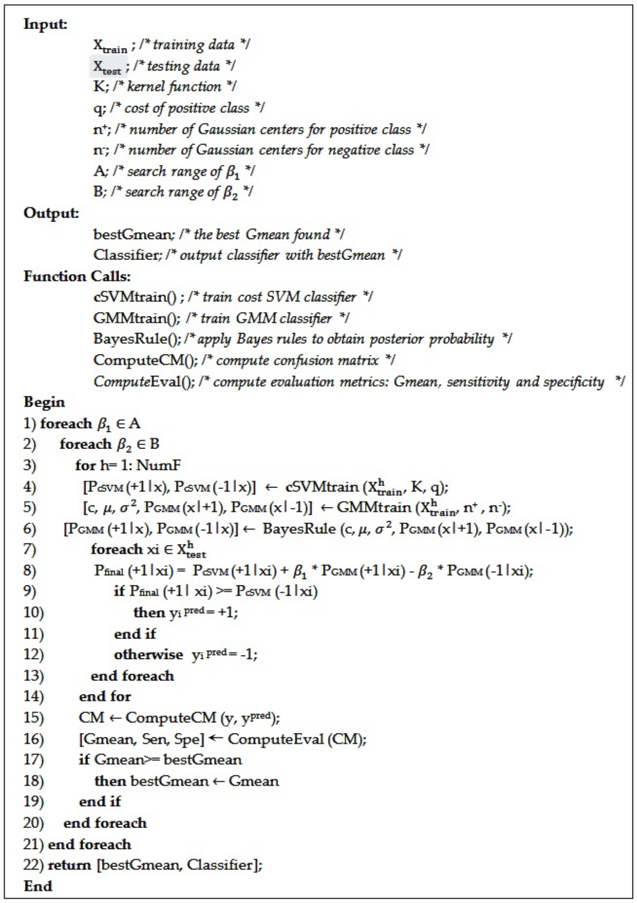
Symbol	Meaning
X_{train}	training dataset
X_{test}	testing dataset
y	True label
y^{pred}	Predicted label
NumF	Number of folds in cross validation
n^+ , n^-	Number of Gaussian centers for positive/negative class
c, μ, σ^2	GMM parameters
q	Cost for positive class in cSVM
P_{cSVM} (+1|x), P_{cSVM} (-1|x)	Probability outputs of cSVM
P_{GMM} (x|+1), P_{GMM} (x|-1)	Probability distribution of GMM
P_{GMM} (+1|x), P_{GMM} (-1|x)	Posterior probabilities of GMM
P_{final} (+1|x)	Modified posterior probability for positive class
β_1, β_2	Combining coefficients
A	Search range of β_1
B	Search range of β_2
C-matrix	Confusion matrix
Sen	Sensitivity
Spe	Specificity

Note that the RBF parameters: kernel parameters γ, c, combining coefficients β_1 and β_2, cost ratio q, are obtained by the grid search method. The search ranges of parameters are defined according to the empirical experience. The detailed parameter settings are discussed in Section 4.

In the CSG algorithm, we combine posterior probabilities of cSVM and GMM for the final classification. The Gaussian mixtures from both positive and negative classes are used to modify the class boundary by adjusting the positive class posterior probability (in (12)). The prediction is made by comparing the posterior probability for each class.

$$\begin{aligned}P_{final}(y=+1|x_i) &= P_{cSVM}(y=+1|x_i)+ \\ &\beta_1 \cdot P_{GMM}(y=+1|x_i) - \beta_2 \cdot P_{GMM}(y=-1|x_i)\end{aligned} \tag{12}$$

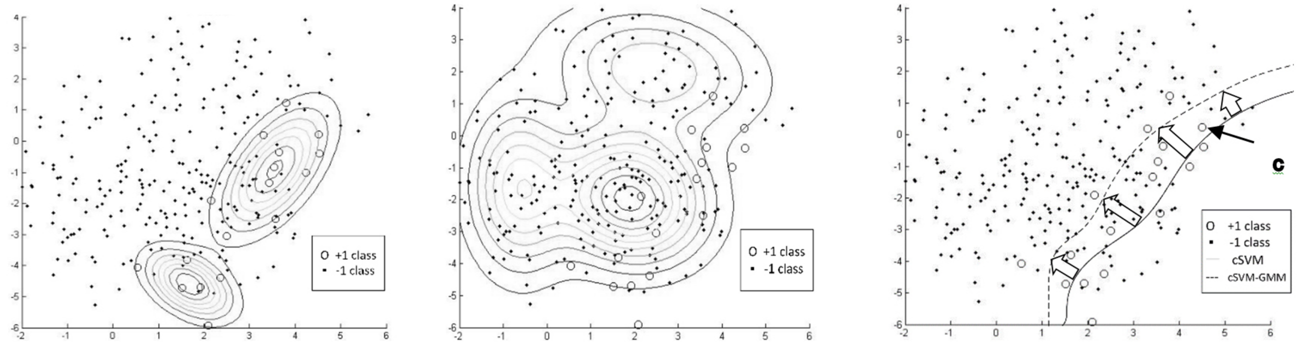
The assumption of integrating the cSVM and GMM posterior probabilities as in (12) is: a positive testing example x_i should generally be closer to the positive Gaussian mixture centers than negative Gaussian mixture centers. Therefore, $P_{GMM}(+1|x_i)$ should be greater than $P_{GMM}(-1|x_i)$. On the other hand, a negative testing example should have $P_{GMM}(+1|x_i)$ less than its $P_{GMM}(-1|x_i)$ in general. By carefully tuning the coefficients β_1 and β_2, the positive test examples may have a better chance of being predicted as positive, while the negative test examples remain negative in prediction.



```
Input:
        $X_{train}$ ; /* training data */
        $X_{test}$ ; /* testing data */
        K; /* kernel function */
        q; /* cost of positive class */
        $n^+$; /* number of Gaussian centers for positive class */
        $n^-$; /* number of Gaussian centers for negative class */
        A; /* search range of $\beta_1$ */
        B; /* search range of $\beta_2$ */
Output:
        bestGmean; /* the best Gmean found */
        Classifier; /* output classifier with bestGmean */
Function Calls:
        cSVMtrain() ; /* train cost SVM classifier */
        GMMtrain(); /* train GMM classifier */
        BayesRule(); /* apply Bayes rules to obtain posterior probability */
        ComputeCM(); /* compute confusion matrix */
        ComputeEval(); /* compute evaluation metrics: Gmean, sensitivity and specificity */
Begin
foreach $\beta_1 \in A$
    foreach $\beta_2 \in B$
      for h= 1: NumF
        [$P_{cSVM}$(+1|x), $P_{cSVM}$(-1|x)] ← cSVMtrain ($X^h_{train}$, K, q);
        [c, $\mu$, $\sigma^2$, $P_{GMM}$(x|+1), $P_{GMM}$(x|-1)] ← GMMtrain ($X^h_{train}$, $n^+$ , $n^-$);
    [$P_{GMM}$(+1|x), $P_{GMM}$(-1|x)] ← BayesRule (c, $\mu$, $\sigma^2$, $P_{GMM}$(x|+1), $P_{GMM}$(x|-1));
        foreach xi ∈ $X^h_{test}$
        $P_{final}$(+1|xi) = $P_{cSVM}$(+1|xi) + $\beta_1$ * $P_{GMM}$(+1|xi) - $\beta_2$ * $P_{GMM}$(-1|xi);
          if $P_{final}$(+1| xi) >= $P_{cSVM}$(-1|xi)
            then $y_i^{pred}$= +1;
          end if
          otherwise $y_i^{pred}$= -1;
        end foreach
      end for
      CM ← ComputeCM (y, $y^{pred}$);
      [Gmean, Sen, Spe] ← ComputeEval (CM);
      if Gmean>= bestGmean
        then bestGmean ← Gmean
      end if
    end foreach
end foreach
return [bestGmean, Classifier];
End
```

Figure 1: The CSG Algorithm

As seen in Figure 2, circles are positive class examples and dots are negative class examples. In Figure 2(a) and 2(b), CSG finds the mixture of Gaussians for positive and negative class respectively. Figure 2(c) shows that CSG pushes the class boundary of cSVM towards the negative class. This is achieved by modifying the cSVM probability output with the GMM probabilities using (12). For illustration, let C be a positive class example, assume cSVM predicts C as negative class with $P_{cSVM}(+1|C)$ = 0.45 and $P_{cSVM}(-1|C)$ = 0.55. By using GMM method, we find $P_{GMM}(+1|C)$ = 0.3 and $P_{GMM}(-1|C)$ = 0.7. If we choose $\beta_1 = \beta_2$=1, according to (12), we have $P_{final}(+1|C)$ = 0.45 +1*0.3-1*0.7 = 0.05. Then, C will be predicted as positive since $P_{final}(+1|C) > P_{cSVM}(-1|C)$. This example shows CSG can push the class boundary of cSVM towards the negative class to improve the discriminative power in identifying the positive examples.



(a)Mixture of Gaussians for positive class (b) Mixture of Gaussians for negative class (c) Estimated boundaries of cSVM and CSG

Figure 2: Illustration example of CSG algorithm

4 Experiments and results

In this section, we first test the performance of CSG using eleven KEEL benchmark datasets.[34] Next, we use a medical imaging dataset to test the applicability of CSG on a real world application. To evaluate the performance of the classifiers, we use the Gmean[35] metric which has been widely used for evaluating classifiers on imbalanced datasets.[13,36,37] Gmean is defined as $\sqrt{acc^{+} \star acc^{-}}$, where acc^{+}(also called sensitivity) and acc^{-} (also called specificity) are positive and negative class prediction accuracy, respectively. Other than Gmean, sensitivity is of great interest in many imbalanced learning domains,[13,38,39] because improving the prediction accuracy on the minority class is the focus of many domain applications. In this section, we focus the discussion on Gmean and sensitivity to show the outperformance of CSG. Specificity measure is also provided. In addition to sensitivity, specificity and Gmean, Area Under the Curve (AUC) metric which has been widely used in imbalanced classification problems[10,22] is provided.

4.1 KEEL benchmark datasets

The eleven benchmark datasets we used in the experiments are collected from KEEL-dataset repository.[34] The details of the datasets are listed in Table 2. The imbalance ratio (IR) varies from 2 to 130 among these datasets. The original multiclass datasets are preprocessed as binary class problems, and the number in name of dataset indicates a positive class. For example, in vehicle2, class 2 is used as positive class and all the other classes in the original data have been joined to represent the negative class.

In the experiments, we first compare CSG with the standard SVM and cSVM algorithms to show fusing GMM knowledge into cSVM can improve the classification on imbalanced datasets. Then we compare the performance of CSG with SMOTE based algorithms such as SMOTE-SVM and SMOTE-cSVM which has been compared in many literatures.[13,18,39] Lastly, we further explore the effect of sampling on CSG by combining SMOTE with the CSG algorithm.

We use LIBSVM[40] MATLAB codes to build the SVM and cSVM models. SMOTE method is applied to preprocess the datasets using KEEL data mining software.[34] The datasets are oversampled until both the classes are equal in number. We apply 10-fold stratified cross validation on each dataset so that the GMM method would have equal number of positive examples to train in each fold. In each fold, we use the SMOTE data to train the model and original data to test the model performance. The results of the 10-folds are aggregated to form the final result. Due to the random nature of the GMM algorithm, each experiment of CSG algorithm has been run 20 times and the mean and standard deviation has been listed. The parameters: RBF kernel parameters γ, c, combining coefficients β_1, β_2, cost ratio q are obtained by the grid search method. The searching ranges of the parameters are defined according to the empirical experience. γ is searched from 0 to 512, c from 0 to 2048, β_1, β_2 from 0 to 10^{10}. q is related to the class IR. The search range for q is from 1 to $IR^{1.4}$.

Table 2: The KEEL dataset used in the experiments

Dataset	#Examples	#Attributes	#Positive	#Negative	Imbalance Ratio
pima	768	8	268	500	1.9
haberman	306	3	81	225	2.8
contraceptive2	1473	9	333	1140	3.4
hepatitis	80	18	13	67	5.2
yeast3	1484	8	163	1321	8.1
glass2	214	9	17	197	11.6
cleveland_0_vs_4	173	13	13	160	12.3
pageblocks 2	548	10	33	515	15.6
flareF	1066	11	43	1023	23.8
winequality_red_4	1599	11	53	1546	29.2
abalone19	4174	9	32	4142	129.4

Table 3: Results of sensitivity, specificity and Gmean

Dataset		**Algorithmic approach**			**Preprocessing approach**		
		SVM	**cSVM**	**CSG**	**SMOTE-SVM**	**SMOTE-cSVM**	**SMOTE-CSG**
pima	Sen	0.519	0.705	**0.721 ± 0.016**	0.728	0.746	**0.791 ± 0.001**
	Spe	0.876	0.708	0.709 ± 0.015	0.742	0.738	0.706 ± 0.002
	Gmean	0.674	0.707	**0.715 ± 0.003**	0.735	0.742	**0.747 ± 0.001**
haberman	Sen	0.198	0.333	**0.614 ± 0.023**	0.593	0.654	**0.704 ± 0.000**
	Spe	0.951	0.907	0.673 ± 0.013	0.742	0.680	0.661 ± 0.005
	Gmean	0.433	0.550	**0.643 ± 0.014**	0.663	0.667	**0.682 ± 0.002**
contracepti ve2	Sen	0.159	0.270	**0.541 ± 0.000**	0.423	0.471	**0.598 ± 0.000**
	Spe	0.969	0.932	0.725 ± 0.000	0.807	0.768	0.699 ± 0.001
	Gmean	0.393	0.502	**0.626 ± 0.000**	0.585	0.602	**0.647 ± 0.000**
hepatitis	Sen	0.231	0.385	**0.746 ± 0.043**	0.769	0.846	**0.923 ± 0.000**
	Spe	0.985	0.955	0.841 ± 0.039	0.866	0.866	0.821 ± 0.000
	Gmean	0.477	0.606	**0.791 ± 0.013**	0.816	0.856	**0.870 ± 0.000**
yeast3	Sen	0.791	0.840	**0.853 ± 0.000**	**0.963**	**0.963**	**0.963 ± 0.000**
	Spe	0.976	0.953	0.943 ± 0.000	0.907	0.907	0.907 ± 0.000
	Gmean	0.879	0.895	**0.897 ± 0.000**	**0.935**	**0.935**	**0.935 ± 0.000**
glass2	Sen	0.000	0.118	**0.694 ± 0.055**	0.706	0.882	**0.897 ± 0.025**
	Spe	0.990	0.995	0.724 ± 0.047	0.858	0.711	0.721 ± 0.013
	Gmean	0.000	0.342	**0.707 ± 0.017**	0.778	0.792	**0.804 ± 0.005**
cleveland_0 _vs_4	Sen	0.077	0.077	**0.665 ± 0.056**	0.615	0.538	**0.615 ± 0.000**
	Spe	1.000	1.000	0.565 ± 0.039	0.688	0.800	0.800 ± 0.000
	Gmean	0.277	0.277	**0.611 ± 0.019**	0.650	0.656	**0.702 ± 0.000**
pageblocks 2	Sen	0.485	0.515	**0.577 ± 0.000**	0.606	0.636	**0.879 ± 0.000**
	Spe	0.996	0.996	0.994 ± 0.000	0.963	0.922	0.784 ± 0.000
	Gmean	0.695	0.716	**0.757 ± 0.000**	0.764	0.766	**0.830 ± 0.000**
flareF	Sen	0.023	0.116	**0.653 ± 0.036**	**0.907**	**0.907**	**0.907 ± 0.000**
	Spe	0.999	0.994	0.790 ± 0.032	0.833	0.833	0.833 ± 0.000
	Gmean	0.152	0.340	**0.718 ± 0.019**	**0.869**	**0.869**	**0.869 ± 0.000**
winequality _red_4	Sen	0.000	0.000	**0.585 ± 0.000**	0.585	0.585	**0.623 ± 0.000**
	Spe	1.000	1.000	0.498 ± 0.002	0.735	0.735	0.704 ± 0.000
	Gmean	0.000	0.000	**0.540 ± 0.001**	0.656	0.656	**0.662 ± 0.000**
abalone19	Sen	0.000	0.031	**0.613± 0.033**	**0.813**	**0.813**	**0.813 ± 0.000**
	Spe	1.000	0.990	0.687 ± 0.024	0.733	0.772	0.772 ± 0.000
	Gmean	0.000	0.176	**0.648 ± 0.015**	0.772	**0.792**	**0.792 ± 0.000**

Table 3 presents the sensitivity, specificity and Gmean measures of each method. For algorithmic approaches, SVM shows good specificity but poor sensitivity in general for all eleven experiments since it trends to predict all examples as majority (negative) class. Both cSVM and CSG show improvements on the sensitivity with sacrifice on specificity to some extent. CSG achieves highest sensitivity for all eleven datasets, and for five datasets (glass2, cleveland_0_vs_4, flareF, winequality_red_4, abalone19) on which SVM and cSVM fails completely, CSG works reasonably well. This is because CSG exploits the underlying knowledge of the imbalanced data distribution in the model building and thus further improves the discriminative power of positive examples. For SMOTE-based methods, SMOTE-CSG shows best sensitivity on seven out of eleven datasets, and equal sensitivity on the remaining four datasets (yeast3, cleveland_0_vs_4, flareF and abalone19). In conclusion, CSG method is effective in dealing with imbalanced classification problems.

In all eleven datasets, CSG achieves best Gmean among all three algorithmic approaches, while SMOTE-CSG achieves best Gmean among all three preprocessing approaches. Compared with SVM, cSVM shows better Gmean measures in nine out of eleven datasets, while CSG further improves cSVM in all eleven datasets by fusing the underlying knowledge of the data distributions to the model training process. As a result, CSG is able to further enhance the Gmean measure on datasets, such as abalone19 and winequality_red_4, where cSVM shows little or even no improvement over SVM. Compared with SVM and cSVM, SMOTE based methods, SMOTE-SVM and SMOTE-cSVM show improved Gmean on all eleven datasets. This indicates that SMOTE is effective in enhancing the classifiers (SVM and cSVM) on imbalanced datasets. Similarly, the SMOTE-CSG method also achieves better Gmean than the CSG method. Among all three SMOTE based methods, SMOTE-CSG outperforms others in nine out of eleven datasets, and in the other two datasets it has equal Gmean with the second best method SMOTE-cSVM. These results show that CSG is effective in dealing with imbalanced datasets.

SMOTE-CSG shows significant improved performance than the CSG on all eleven datasets. SMOTE oversamples the data by adding synthetic data instances which are generated using convex combinations of the existing data. In SMOTE-CSG method, SMOTE provides more training data to CSG algorithm which can aid the training process of cSVM and GMM, and thus lead to better class separation. In all, the experimental results indicate that the preprocessing method SMOTE is necessary in dealing with imbalanced datasets.

Table 4: Results of AUC

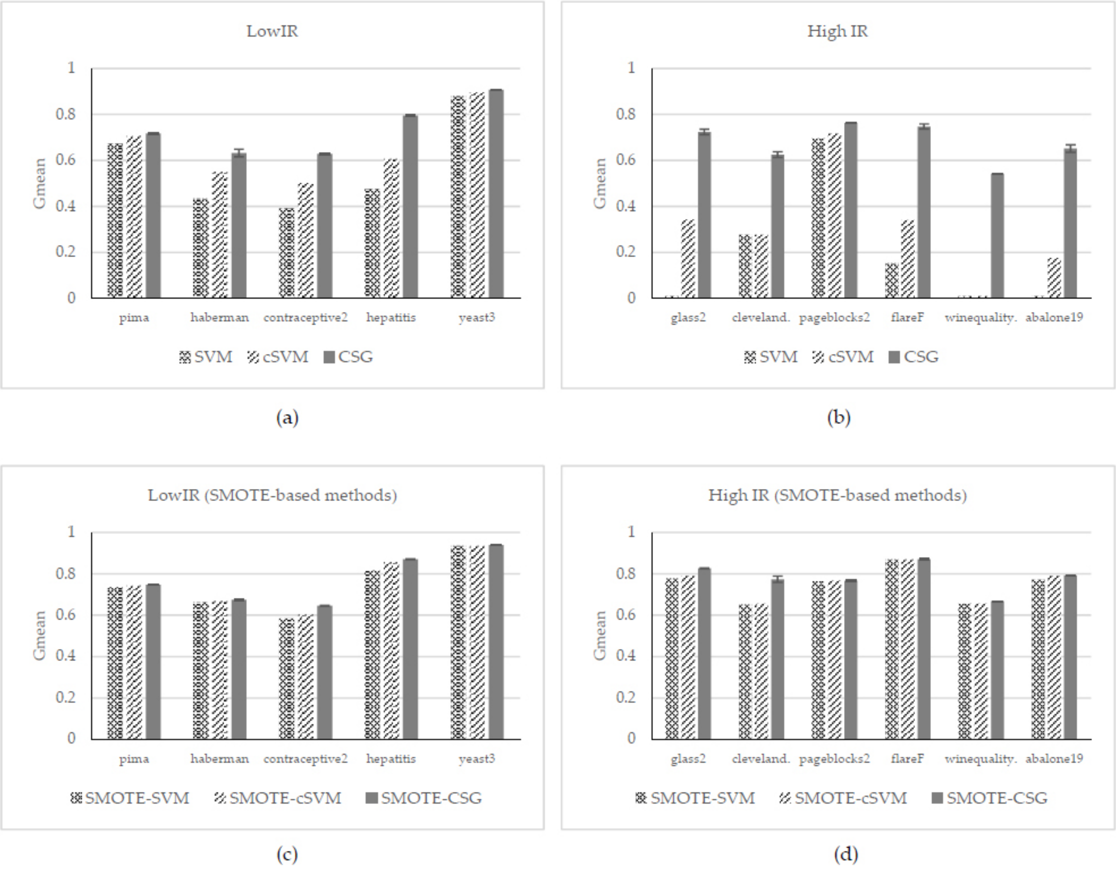
Dataset	Algorithmic approach			Preprocessing approach		
	SVM	cSVM	CSG	SMOTE-SVM	SMOTE-cSVM	SMOTE-CSG
pima	0.719	0.728	0.736 ± 0.007	0.740	0.748	0.750 ± 0.001
haberman	0.627	0.658	0.670 ± 0.011	0.657	0.658	0.680 ± 0.002
contraceptive2	0.604	0.631	0.661 ± 0.000	0.610	0.611	0.636 ± 0.000
hepatitis	0.705	0.773	0.864 ± 0.016	0.760	0.816	0.844 ± 0.000
yeast3	0.894	0.913	0.916 ± 0.000	0.938	0.938	0.938 ± 0.000
glass2	0.563	0.603	0.745 ± 0.023	0.834	0.807	0.821 ± 0.013
cleveland_0_vs_4	0.608	0.608	0.666 ± 0.019	0.650	0.642	0.693 ± 0.000
pageblocks2	0.788	0.815	0.842 ± 0.000	0.797	0.778	0.829 ± 0.000
flareF	0.574	0.599	0.752 ± 0.014	0.878	0.878	0.878 ± 0.000
winequality_red_4	0.564	0.564	0.572 ± 0.001	0.657	0.657	0.658 ± 0.000
abalone19	0.569	0.599	0.682 ± 0.016	0.786	0.797	0.797 ± 0.000

The AUC metric in Table 4 shows similar results as Gmean in Table 3. For algorithmic approaches, CSG shows better AUC than both SVM and cSVM for all eleven datasets. For preprocessing approaches, SMOTE-CSG shows better AUC in eight out of eleven datasets, and equal AUC for the rest three datasets.

Table 5: Pair t-test on Sensitivity, Specificity, G-Mean and AUC (p<.05 indicating significant differences)

Metrics	Algorithmic approach		Preprocessing approach	
	CSG vs. SVM	CSG vs. cSVM	SMOTE-CSG vs. SMOTE-SVM	SMOTE-CSG vs. SMOTE-cSVM
Sensitivity	0.00008	0.00063	0.00996	0.01863
Specificity	0.00042	0.00226	0.11836	0.04760
Gmean	0.00124	0.00269	0.00344	0.01879
AUC	0.00214	0.00669	0.03539	0.01401

In addition, a pair t-test is conducted to draw statistical conclusions in comparing the performance of our proposed CSG with the other two algorithmic approaches: SVM and CSG, and the performance of our proposed SMOTE-CSG with the other two preprocessing approaches: SMOTE-SVM and SMOTE-cSVM. As shown in Table 5, CSG statistically outperforms SVM and cSVM on all four metrics, that is, sensitivity, specificity, Gmean and AUC (p<.05). SMOTE-CSG statistically outperforms SMOTE-SVM and SMOTE-cSVM on three metrics: sensitivity, Gmean and AUC. As for specificity, SMOTE-CSG outperforms SMOTE-cSVM yet underperforms SMOTE-SVM. This may be explained that SMOTE-SVM (and even SMOTE-cSVM) is designed to perform well on specificity. As indicated earlier, this research focuses on sensitivity which is more important for imbalanced data while Gmean and AUC are overall measures considering the tradeoffs between the sensitivity and specificty. Therefore we conclude CSG and SMOTE-CSG are satisfactory in handling imbalanced data.



(a)

(b)

(c)

(d)

Figure 3: Gmeans for Low IR datasets and High IR datasets

To evaluate the effect of IR on each method, we divide the datasets into low-IR group (IR<10) and high-IR group (IR$\geq$10). Figure 3 shows the Gmean measures of each datasets in each group. Figure 3(a) and 3(b) are the comparison of SVM, cSVM and CSG, and Figure 3(c) and 3(d) are the comparison of SMOTE-SVM, SMOTE-cSVM and SMOTE-CSG. Figure 3(a) and 3(b) show that CSG has a significant improvements of Gmean compared to SVM and cSVM on high-IR datasets relative to low-IR datasets which indicates CSG is very effective in dealing with highly imbalanced datasets on which SVM and cSVM performs poorly. This is because in highly imbalanced datasets, the majority class dominates the training of the SVM and thus the class boundary is highly skewed. cSVM shows improved performance by assigning higher cost to the minority class, but its performance is still less than satisfactory due to the limited ability to enforce cost-sensitivity as we discussed in Section 3.2. CSG tackles the high imbalance issue by fusing the underlying knowledge of the data distribution (GMM) into the training process of cSVM, and thus the skewed class boundary can be adjusted towards the majority class. In all, the performance of CSG is much better on the high-IR group than on low-IR group.

For SMOTE-based methods (see Figure 3(c) and (d)), SMOTE-CSG marginally improved Gmean over both SMOTE-SVM and SMOTE-cSVM methods. This is because the SMOTE method oversamples the minority class until the whole dataset is balanced and SVM generally performs well on balanced datasets since the class boundary of SVM is not skewed. As a result, methods such as cSVM and CSG which aim to adjust the skewed class boundary would have marginal performance improvements over SVM on balanced datasets.

To further test the performance of CSG, a real world renal stone medical image dataset is collected from Mayo Clinic, Arizona. The comparison experiment is discussed in the next section.

4.2 Renal stone dataset

Renal stones, also called kidney calculi, are the solid crystal aggregations formed in the kidneys from dietary minerals in the urine. Renal stone disease can cause nausea and vomiting with sharp pain in the back or lower abdomen and sometimes blood in urine (*e.g.*, hematuria).[41] It affects approximately one in eleven people in the United States.[42] Each year, more than one million visits to health care providers are related to the renal stone disease.[41] Based on the chemical composition, clinically relevant renal stones can be categorized into four types: unic acid, calcium oxalate, struvite and cystine. The determination of the chemical composition of renal stone is a key factor in preoperative patient evaluation, treatment planning and recurrence prevention.[43] The commonly used stone analysis techniques include in vitro x-ray diffraction, infrared spectroscopy and polarization microscopy.[44] These tests, unfortunately, are performed only after the stones are extracted from the patients. In renal stone preoperative evaluation, minimally invasive intervention is preferred for the benefits of the patients. Utilizing noninvasive tests such as radiology imaging studies to identify the renal stone composition is of great interest.[45,46]

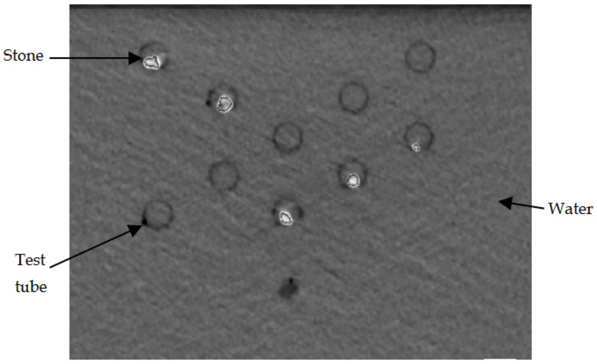


Figure 4: The DECT image of renal stones (phantom study)

Dual Energy CT (DECT) is a recently developed technique used for the purpose of diagnostic imaging. Instead of acquiring a single data set as per conventional CT, it acquires two simultaneous or near simultaneous data sets, one low and one high energy, during a single acquisition. This setting enables DECT to differentiate materials with similar electron densities but varying photon absorption abilities,[47] improving noninvasive renal stone characterization.[48]

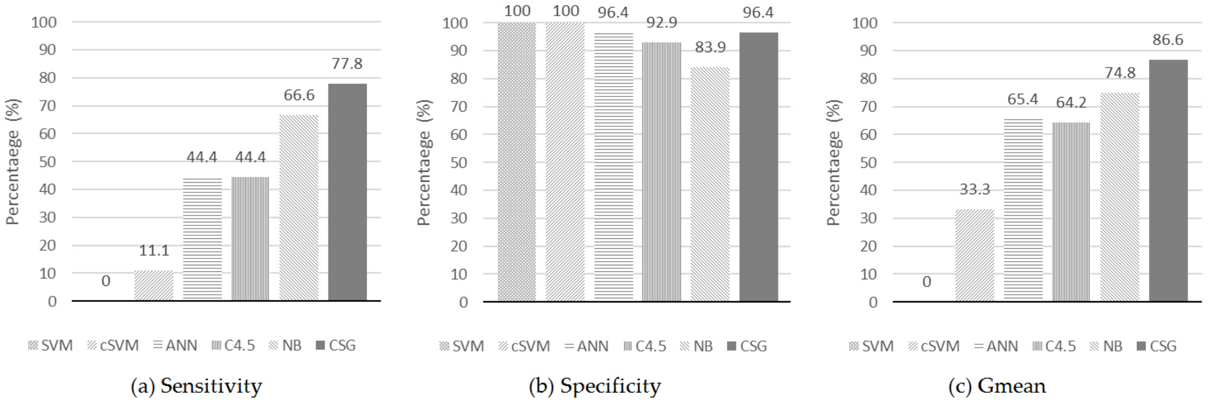
In this study, we collect 65 stones from the stone analysis laboratory at Mayo Clinic Arizona. All stones are extracted from previous patients through surgical and endoscopic intervention. The chemical composition has been determined with stereo microscopy and infrared spectrophotometry. According to the chemical composition, the 65 stones are divided into four groups: uric acid (n = 34), calcium oxalate (n = 18), cystine (n = 9) and struvite (n = 4). The diameter of the stones varies from 2.6 mm to 6.2 mm (mean size 3.5 mm). Among all four types of renal stones, cystine stone is of great interest for the following reasons: first, cystine stone is usually too dense to be broken up by applying extracorporeal shock wave lithotripsy as can be done for some other types of stones. Instead, techniques designed for removing dense stones, such as percutaneous nephrolithotripsy (PNL), may be applied. Second, cysteine stone is the result of cystinuria, which is a genetic autosomal recessive metabolic disorder.[49] Patients with cysteine stones may also need to take additional genetic screening tests other than medical treatment.[50] In this experiment, cystine stone has been selected as the target class, and the other stone types are combined as the non-target class. Thus, the imbalance ratio is 6.2 (n=56 for non-cystine stones and n=9 for cystine stones). The details of the DECT renal stone dataset are shown in Table 6.

In this comparison experiment, we are interested in showing the outperformance of CSG over cSVM. In addition, some commonly used machine learning algorithms in medical data classification problems such as SVM,[51] artificial neural network (ANN),[52] C4.5[53] and NaiveBayes (NB)[54] are also implemented for comparison. The SVM, cSVM and CSG methods are performed using the same settings as in section 4.1. The ANN, C4.5 and NB methods are performed using data mining software Weka 3.6.9.[55] 5-fold stratified cross validation is applied. In addition to sensitivity, specificity and Gmean, we also use two other important evaluation metrics for the medical diagnosis field: Positive Predictive Value (PPV) and Negative Predictive Value (NPV). PPV indicates the probability that patients with positive screening tests truly have the disease, while NPV shows the probability that patients with negative screening tests truly don't have the disease. The results are shown in Figure 5 and Figure 6.

Figure 5 shows that the standard SVM method performs poorly on this imbalanced dataset. The zero sensitivity shows that SVM has no recognition ability of the cystine stones. cSVM improves the sensitivity very little (11.1%), and still far less than satisfactory. CSG method has much better sensitivity than SVM or cSVM (77.8% *vs.* 0% and 11.1%). ANN has equal sensitivity with C4.5 (44.4%) but higher specificity (96.4% *vs.* 92.9%). Compared with ANN, NB has better sensitivity (66.6%), but lower specificity (83.9%). The CSG method achieves highest sensitivity (77.8%) and Gmean (86.6%) among all six methods while maintaining high specificity (96.4%). The CSG method also achieves the second highest values in PPV (77.8%) and the highest value in NPV (96.4%) according to Figure 6. In conclusion, CSG outperforms the other five methods in classification of cystine stones.

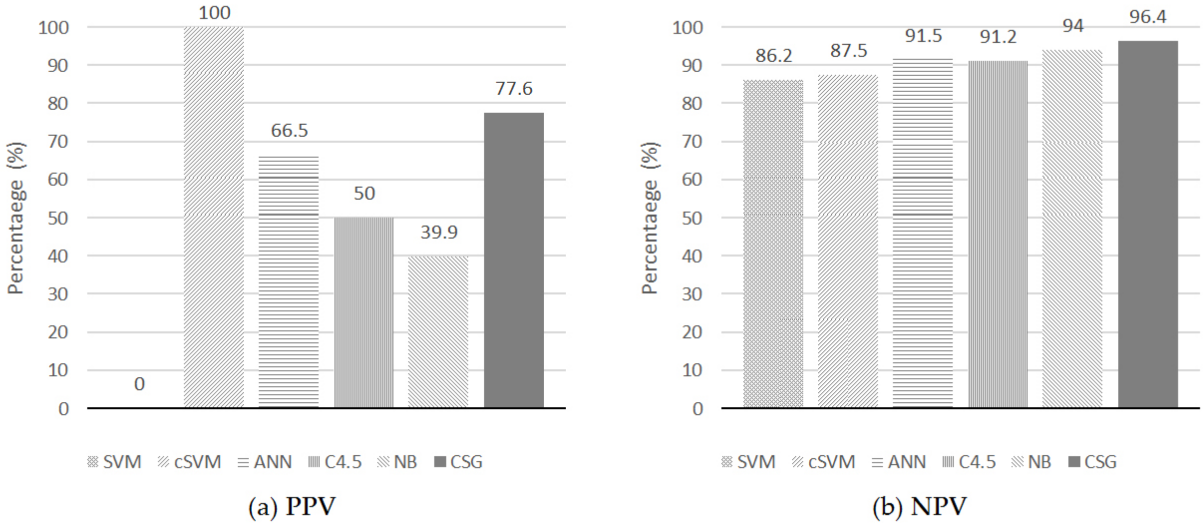
Table 6: The RenalStone_cys dataset

Dataset	#Examples	#Features	#Positive	#Negative	IR	Feature Description
RenalStone_cys	65	18	9	56	6.2	11 energy level measures 1 effective atomic number 6 material density measures



(a) Sensitivity (b) Specificity (c) Gmean

Figure 5: Sensitivity, Specificity and Gmean on RenalStone_cys dataset



(a) PPV (b) NPV

Figure 6: PPV and NPV on RenalStone_cys dataset

5 Conclusion and discussion

In this research, we propose a model fusion based approach integrating cSVM with GMM for the imbalanced classification problem. The CSG method augments cSVM by incorporating the GMM modeling of imbalanced data distribution into the training process and thus leads to better identification of the minority class examples. Experimental results on KEEL benchmark datasets and the medical imaging dataset show the CSG method to be effective in dealing with imbalanced classification problems.

We also find that CSG performs even better when the dataset is preprocessed by SMOTE method. This is because the synthetic data instances generated by SMOTE creates larger and less specific decision regions for the cSVM and GMM models to learn from, thus the decision boundary can be further adjusted towards the majority class and thus lead to better class separation. In all, the SMOTE method can further improve the CSG method in dealing with imbalance classification problems.

References

[1] Duan KB, Keerthi SS. Which is the best Multiclass SVM method? An Empirical Study. Springer Berlin Heidelberg. 2005: 278-285.

[2] Cortes C, Vapnik V. Support-vector Networks. Machine Learning. 1995; 20(3): 273-297. http://dx.doi.org/10.1023/A:1022627411411

[3] Bazzani A, Bevilacqua A, Bollini D, *et al*. An SVM Classifier to Separate False Signals from Microcalcifications in Digital Mammograms. Physics in Medicine and Biology. 2001; 46(5): 1651. PMid:11419625

[4] Collobert R, Bengio S. Links Between Perceptrons, MLPs and SVMs. In Proceedings of the twenty-first international conference on Machine learning, ACM, 2004.

[5] McLachlan G. Discriminant Analysis and Statistical Pattern Recognition. 2004; 544.

[6] Shon T, Kim Y, Lee C, *et al*. A Machine Learning Framework for Network Anomaly Detection using SVM and GA. In Information Assurance Workshop, 2005. IAW'05. Proceedings from the Sixth Annual IEEE SMC, 2005.

[7] Scarfone K, Mell P. Guide to Intrusion Detection and Prevention Systems (IDPS). NIST Special Publication. 2007; 800: 94.

[8] Veropoulos K, Campbell C, Cristianini N. Controlling the Sensitivity of Support Vector Machines. Proceedings of the International Joint Conference on Artificial Intelligence, 1999.

[9] Wu G, Chang EY. Adaptive Feature-space Conformal Transformation for Imbalanced-data Learning. MACHINE LEARNING-INTERNATIONAL WORKSHOP THEN CONFERENCE, 2002.

[10] He H, Garcia EA. Learning from Imbalanced Data. Knowledge and Data Engineering. IEEE Transactions. 2009; 21(9): 1263-84.

[11] Wu G, Chang EY. Aligning Boundary in Kernel Space for Learning Imbalanced Dataset. Data Mining, ICDM'04. Fourth IEEE International Conference on. IEEE, 2004.

[12] Chawla NV, Bowyer KW, Hall LO, *et al*. SMOTE: Synthetic Minority Over-sampling Technique. Journal of Artificial Intelligence Research. 2002; 16: 321-357.

[13] Akbani R, Kwek S, Japkowicz N. Applying Support Vector Machines to Imbalanced Datasets. Machine Learning: ECML 2004, Berlin Heidelberg, 2004.

[14] Wu G, Chang EY. Class-boundary Alignment for Imbalanced Dataset Learning, in ICML 2003 Workshop on Learning from Imbalanced Data Sets II, Washington, DC, 2003.

[15] Chawla NV, Japkowicz N, Kotcz A. Editorial: Special Issue on Learning from Imbalanced Data Sets. ACM SIGKDD Explorations Newsletter. 2004; 6(1): 1-6.

[16] Masnadi-Shirazi H, Vasconcelos N, Iranmehr A. Cost-Sensitive Support Vector Machines. arXiv preprint arXiv:1212.0975, 2012.

[17] Maloof MA. Learning when Data Sets are Imbalanced and when Costs are Unequal and Unknown. In ICML-2003 Workshop on Learning from Imbalanced Data Sets II, 2003.

[18] Cao P, Zhao D, Zaiane O. An Optimized Cost-Sensitive SVM for Imbalanced Data Learning. In Advances in Knowledge Discovery and Data Mining, 2013. p.280-292.

[19] Brefeld U, Geibel P, Wysotzki F. Support Vector Machines with Example Dependent Costs. In Machine Learning: ECML 2003, 2003.

[20] Jordan A. On Discriminative vs. Generative Classifiers: A Comparison of Logistic Regression and Naive Bayes. Advances in Neural Information Processing Systems. 2002; 14: 841.

[21] Bishop CM, Nasrabadi NM. Pattern Recognition and Machine Learning, New York: springer, 2006.

[22] Chawla NV. Data Mining for Imbalanced Datasets: An Overview. In Data Mining and Knowledge Discovery Handbook, Springer, 2005. p.853-867.

[23] Batista GE, Prati RC, Monard MC. A Study of the Behavior of Several Methods for Balancing Machine Learning Training Data. ACM SIGKDD Explorations Newsletter. 2004; 6(1): 20-29. http://dx.doi.org/10.1145/1007730.1007735

[24] Holte RC, Acker L, Porter BW. Concept Learning and the Problem of Small Disjuncts. Proceedings of the Eleventh International Joint Conference on Artificial Intelligence, 1989.

[25] Estabrooks A, Jo T, Japkowicz N. A Multiple Resampling Method for Learning from Imbalanced Data Sets. Computational Intelligence. 2004; 20(1): 18-36. http://dx.doi.org/10.1111/j.0824-7935.2004.t01-1-00228.x

[26] Karakoulas G, Shawe-Taylor J. Optimizing Classifiers for Imbalanced Training Sets. In Proceedings of the 1998 Conference on Advances in Neural Information Processing Systems II. 1999: 253-259.

[27] Bishop CM. Neural Networks for Pattern Recognition, Oxford university press, 1995.

[28] Platt J. Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods. Advances in Large Margin Classifiers, the MIT Press. 2000: 61-74.

[29] Kim D, Lee SC. Pairwise Threshold for Gaussian Mixture Classification and its Application on Human Tracking Enhancemen. Advanced Video and Signal-Based Surveillance (AVSS) 2012 IEEE Ninth International Conference on, 2012.

[30] Wang K, Ren Z. Enhanced Gaussian Mixture Models for Object Recognition using Salient Image Features. Mechatronics and Automation, 2007. ICMA 2007. International Conference on, 2007.

[31] Reynolds DA, Rose RC. Robust Text-independent Speaker Identification using Gaussian Mixture Speaker Models. Speech and Audio Processing, IEEE Transactions. 1995; 3(1): 72-83.

[32] Fauve BG, Evans NM, Pearson N, *et al*. Influence of Task Duration in Text-independent Speaker Verification. In Proc. Interspeech, 2007.

[33] Dempster AP, Laird NM, Rubin DB. Maximum Likelihood from Incomplete Data via the EM Algorithm. Journal of the Royal Statistical Society. Series B (Methodological). 1977: 1-38.

[34] Alcalá J, Fernández A, Luengo J, *et al*. KEEL Data-Mining Software Tool: Data Set Repository, Integration of Algorithms and Experimental Analysis Framework. Journal of Multiple-Valued Logic and Soft Computing. 2011; 17(2-3): 255-287.

[35] Kubat M, Holte R, Matwin S. Learning when Negative Examples Abound. Machine Learning: ECML-97, 1997.

[36] Wang HY. Combination Approach of SMOTE and Biased-SVM for Imbalanced Datasets. In Neural Networks, 2008. IJCNN 2008.(IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on IEEE, 2008.

[37] Imam T, Ting KM, Kamruzzaman J. z-SVM: An SVM for Improved Classification of Imbalanced Data. In AI 2006: Advances in Artificial Intelligence, Berlin Heidelberg, 2006.

[38] Maciejewski T, Stefanowski J. Local Neighbourhood Extension of SMOTE for Mining Imbalanced Data. In Computational Intelligence and Data Mining (CIDM), 2011 IEEE Symposium on, 2011.

[39] Hui H, Wang WY, Mao BH. Borderline-SMOTE: A new Over-sampling Method in Imbalanced Data Sets Learning. In Advances in Intelligent Computing, Berlin Heidelberg, 2005.

[40] Chang CC, Lin CJ. LIBSVM: A Library for Support Vector Machines. ACM Transac-tions on Intelligent Systems and Technology. 2011; 2(27): 1-27.

[41] NKUDIC. Kidney Stones in Adults, 2013. Available from: http://kidney.niddk.nih.gov/kudiseases/pubs/stonesadults/?control=Pubs. [Accessed 31 10 2013].

[42] Scales CD, Smith AC, Hanley JM, *et al*. Prevalence of kidney stones in the United States. European urology. 2012; 62(1): 160-165. PMid:22498635.

[43] Eliahou R, Hidas G, Duvdevani M, *et al*. Determination of renal stone composition with dual-energy computed tomography: an emerging application. Seminars in Ultrasound, CT, and MRI. 2010; 31(4): 315-320.

[44] Hidas G, Eliahou R, Duvdevani M, *et al*. Determination of renal stone composition with dual-energy CT: in vivo analysis and comparison with x-ray diffraction. Radiology. 2010; 257(2): 394-401. PMid:20807846. http://dx.doi.org/10.1148/radiol.10100249

[45] Abdel-Halim RE, Abdel-Halim MR. A review of urinary stone analysis techniques. Saudi medical journal. 2006; 27(10): 1462. PMid:17013464.

[46] GOEL R, WASSERSTEIN AG. Kidney Stones: Diagnostic and Treatment Strategies. Consultant. 2012; 52: 121-130.

[47] Riedel M. An Introduction to Dual Energy Computed Tomography [Internet]. Available from: http://ric.uthscsa.edu/personalpages/lancaster/DI2_Projects_2010/dual-energy_CT.pdf. [Accessed 1 11 2013].

[48] Graser A, Johnson TR, Bader M, *et al*. Dual energy CT characterization of urinary calculi: initial in vitro and clinical experience. Investigative radiology. 2008; 43(2): 112-119. PMid:18197063.

[49] Wu J. Chapter 58 – Urolithiasis. Integrative Medicine, 3rd ed, WB Saunders Company, 2012.

[50] Breuning MH, Hamdy NA. From gene to disease; SLC3A1, SLC7A9 and cystinuria. Nederlands tijdschrift voor geneeskunde. 2003; 147(6): 245. PMid:12621979.

[51] Dal Moro F, Abate A, Lanckriet GRG, *et al*. A Novel Approach for Accurate Prediction of Spontaneous Passage of Ureteral Stones: Support Vector Machines. Kidney International. 2006; 69(1): 157-160. PMid:16374437. http://dx.doi.org/10.1038/sj.ki.5000010

[52] Chiang D, Chiang H, Chen W, *et al*. Prediction of Stone Disease by Discriminant Analysis and Artificial Neural Networks in Genetic Polymorphisms: a New Method. BJU International. 2003; 7: 661-666.

[53] Kaladhar D, Krishna AR, Varahalarao V. Statistical and Data Mining Aspects on Kidney Stones: A Systematic Review and Meta-analysis. 2012; 1: 543. http://dx.doi.org/10.4172/scientificreports

[54] Lavanya D, Rani K. Performance Evaluation of Decision Tree Classifiers on Medical Datasets. International Journal of Computer Applications. 2011; 26(4): 1-4.

[55] Hall M, Frank E, Holmes G, *et al*. The WEKA Data Mining Software: An Update. SIGKDD Explorations. 2009; 11(1).