A Model Fusion Based Framework

For Imbalanced Classification Problem with Noisy Dataset

by

Miao He

A Dissertation Presented in Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

Approved October 2014 by the
Graduate Supervisory Committee:

Teresa Wu, Chair
Jing Li
Alvin Silva
Connie Borror

ARIZONA STATE UNIVERSITY

December 2014

ABSTRACT

Data imbalance and data noise often coexist in real world datasets. Data imbalance affects the learning classifier by degrading the recognition power of the classifier on the minority class, while data noise affects the learning classifier by providing inaccurate information and thus misleads the classifier. Because of these differences, data imbalance and data noise have been treated separately in the data mining field. Yet, such approach ignores the mutual effects and as a result may lead to new problems. A desirable solution is to tackle these two issues jointly. Noting the complementary nature of generative and discriminative models, this research proposes a unified model fusion based framework to handle the imbalanced classification with noisy dataset.

The phase I study focuses on the imbalanced classification problem. A generative classifier, Gaussian Mixture Model (GMM) is studied which can learn the distribution of the imbalance data to improve the discrimination power on imbalanced classes. By fusing this knowledge into cost SVM (cSVM), a CSG method is proposed. Experimental results show the effectiveness of CSG in dealing with imbalanced classification problems.

The phase II study expands the research scope to include the noisy dataset into the imbalanced classification problem. A model fusion based framework, K Nearest Gaussian (KNG) is proposed. KNG employs a generative modeling method, GMM, to model the training data as Gaussian mixtures and form adjustable confidence regions which are less sensitive to data imbalance and noise. Motivated by the K-nearest neighbor algorithm, the neighboring Gaussians are used to classify the testing instances.

Experimental results show KNG method greatly outperforms traditional classification methods in dealing with imbalanced classification problems with noisy dataset.

The phase III study addresses the issues of feature selection and parameter tuning of KNG algorithm. To further improve the performance of KNG algorithm, a Particle Swarm Optimization based method (PSO-KNG) is proposed. PSO-KNG formulates model parameters and data features into the same particle vector and thus can search the best feature and parameter combination jointly. The experimental results show that PSO can greatly improve the performance of KNG with better accuracy and much lower computational cost.

ACKNOWLEDGMENTS

The first thanks are to my dear Lord Jesus. Along the whole journey of my Ph.D study, His word is my strength and His dear presence is my comfort. I thank him for all the training and perfecting of this process for His good purpose.

I would like to thank my dear brothers and sisters in Christ of the Church in Phoenix, who has been nourishing and cherishing me in every aspect of my school life, family life and church life during the past six years.

I would like to thank my family, my dear wife Sarah, my baby son David, my parents and parents-in-law. Thank them all for their supports and encouragements at all times.

This thesis would not have been possible without the help, support and patience of my advisor Prof. Teresa Wu. Thank her for her advice, insights and guidance through this process. The breadth of knowledge and experience that she has imparted to me will serve me throughout my life.

My thesis committee helped and guided me in many ways. My sincere thanks to Prof. Jing Li for her insights and valuable suggestions which inspired me in determining my research topic. Thanks to Dr. Alvin Silva for collecting and sharing data with me which has been invaluable to my research. He is very friendly and easy to get along with. It was such an enjoyable time working with him in my summer interns at Mayo Clinic, Arizona. Thanks to Prof. Connie Borror for her thoughtful comments on my research.

I am also grateful to all my lab mates for their help, encouragement and the enormous valuable discussions: Mengqi Hu, Min Zhang, Can Cui, Debanjan Bhattacharya, Gaurav Bansal, Balaji Solai Rameshbabu, Xianghua Chu, Fei Gao, Yinlin Fu and Congzhe Su.

TABLE OF CONTENTS

Page

CHAPTER Page

LIST OF TABLES

Table Page

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Background and Rationale

In real world application, classification problems always suffer from the data quality issues such as imbalance and noise. These issues not only increase the complexity of learning, but also hinder the performance of most classification algorithms.

Data imbalance occurs when one class (minority class) is greatly outnumbered by another class (majority class). Indeed, many applications call special attention on labeling the minority class. For example, in the field of medical diagnosis (diseased patients), fraud detection (true fraud), identifying the minority examples is the interest (if not the only interest) of the problem. The standard classifiers generally have poor recognition power on the minority class when dealing with imbalance data due to the fact that majority class dominates the whole dataset. As a result, the performance of most standard classifiers is less than satisfactory in dealing with imbalanced dataset.

Data noise occurs when the data has been corrupted by various errors such as systematic uncertainty, measurement error, human error, etc (Sáez et al., 2013),(Zhu & Wu, 2004). Based on its information sources, data noise can be characterized as (1) attribute noise, which refers to the corruption in the attributes, and (2) class noise, which occurs when the instances are incorrectly labeled. Noise may hinder the knowledge extraction from the data and thus makes the classifier less effective, particularly if the classifier is noise-sensitive.

Although various approaches to tackle the imbalance and noise classification problems have been proposed (He & Garcia, 2009),(Chawla, 2005),(Xiong et al., 2006),(Lee et al., 2000),(Mingers, 1989a),(Long & Servedio, 2008), most of the existing approaches deal with

imbalance and noise issues separately. This is because the causes and problematic consequences of imbalance and noise are different, as aforementioned. However, doing so ignores the mutual effects of data imbalance and data noise and thus may lead to new problems. Besides, this two-step procedure is more likely to be computational costly. Thus, a framework that can handle imbalance and noisy data jointly is required. To the best of our knowledge, existing literature focuses on discriminative models (Jordan, 2002) to handle either imbalanced or noisy dataset but not both in the classification problems. This is mainly due to the fact that discriminative model tends to be more effective in forming the class boundary. However, since it works on the raw data directly, discriminative model may be more error-prone to the data imbalance and noise. Alternatively, generative models (Jordan, 2002) focus on extracting the characteristics from the raw data which are expected to be less sensitive to data imbalance and noise. Due to the complementary nature of the generative and discriminative classifiers, in this research, we propose a generative/discriminative model fusion based framework to tackle the problem of imbalanced classification with noisy dataset.

1.2 Research Scope

In this research, we are interested in three specific research questions as following:

Research Question 1: How to handle imbalanced classification problem?

Proposed Approach: CSG: Augmenting cost SVM with Gaussian Mixture Model (GMM) for imbalanced classification.

We first focus on the data imbalance issue only. Based on Bayes decision theory, the misclassification costs of false positive and false negative are generally unequal. Thus, classifier designed using cost sensitive framework is expected to be optimal in dealing with imbalanced dataset (Masnadi-Shirazi et al., 2012). However, the well-known cost sensitive SVM (cSVM)

method does not work well in many empirical studies (Wu & Chang, 2004),(Masnadi-Shirazi et al., 2012),(Cao et al., 2013) because its ability to enforce cost-sensitivity is limited by the KKT condition (detailed discussion can be found in Section 2.3.2). In this study, we propose a model fusion based framework, CSG, which augments cSVM with a generative model, GMM, to improve the performance of cost SVM on imbalanced datasets. By fusing the GMM with cSVM, the skewed class boundary can be pushed back towards the majority class and more minority instances can be correctly recognized. Experimental results on seven UCI benchmark datasets and one real world medical imaging dataset show the effectiveness of CSG in dealing with imbalanced classification problem.

Research Question 2: How to handle imbalanced classification problem with noisy dataset?

Proposed Approach: K Nearest Gaussian (KNG) - a Model Fusion based Framework for Imbalanced Classification with Noisy dataset.

In Phase II study, we further explore the imbalance issue and noise issue jointly. In Phase I study, we show a case where a generative classifier (GMM) can be used as supplementation to a discriminative classifier (cSVM) in dealing with imbalanced classification problem. We also find from literatures that most discriminative classifiers are criticized to be ineffective on imbalanced and noisy data (Sáez et al., 2013),(Akbani et al., 2004). On the contrary, the data characteristics extracted by generative classifiers are expected to be less sensitive to imbalance and noise. This leads us to a research question: instead of using generative classifiers as supplement method, can we turn our focus to generative classifier and use discriminative classifier as supplement to handle the imbalanced and noisy data? Our proposed approach is KNG method. KNG employs GMM to model the training data as Gaussian mixtures and form adjustable confidence regions of each Gaussian. The classification of a testing instance is achieved by majority voting of its

neighboring Gaussians. The experimental study show that KNG method outperforms other commonly used classifiers in both Gmean and robustness measures.

Research Question 3: How to jointly perform feature selection and parameter tuning on KNG method?

Proposed Approach: PSO-KNG: A Particle Swarm Optimization (PSO) based KNG algorithm.

In phase II study, we propose the KNG algorithm to handle imbalanced classification with noisy dataset. Although the experiment results show the effectiveness of KNG, we do find two issues that may hinder the performance of KNG. First, KNG may suffer from the redundancy among the features which may highly impact the effectiveness of GMM. As a result, the Gaussian mixtures modeled by GMM may not be robust. Secondly, we observe through empirical experiments that the success of KNG is mainly based on the proper tuning of the parameters. However, the parameter tuning technique employed in phase II study is grid search, which has been criticized to be inefficient.

To further improve the performance of KNG, we explore the feature selection and parameter tuning issues in phase III study. Traditionally, feature selection and parameter tuning are generally treated as separate process. However, doing so simply ignores the mutual influence among model parameters and data features which may not achieve optimal model performance. In this study, we propose a PSO based method, PSO-KNG, to tackle these two issues jointly. PSO is a stochastic optimization technique. We use PSO to formulate model parameters and data features into the same particle vector so that it can search the best combination of parameters and features which jointly achieve best model performance. The experimental results show that PSO-KNG greatly outperforms KNG in terms of both Gmean and running time measures.

1.3 Dissertation Organization

The rest of this dissertation is organized into three interrelated chapters that address the problem of imbalanced classification with noisy dataset. The reader may encounter some level of redundancy in the writing of this dissertation, this is because each Chapter is written as a standalone paper for scholarly journal publication.

Chapter 2 provides a generative/ discriminative model fusion based approach CSG to tackle the imbalanced classification problem. CSG is built mainly based on the discriminative classifier cSVM, and use the posterior probability provided by GMM as supplement information to aid the classification process. Comparison experiments between the proposed approach and the existing methods are conducted using KEEL benchmark datasets.

Furthermore, Chapter 3 provides a generative/ discriminative model fusion based approach KNG to tackle the imbalance and noise issues jointly. KNG is built mainly based on the generative classifier GMM, and apply the idea of k nearest neighbor on the extracted data characteristics to achieve classification. Comparison experiments between the proposed approach and four widely used classification methods are conducted using UCI benchmark datasets.

Lastly, Chapter 4 provides PSO based method to further improve KNG algorithm by tackling feature selection and parameter tuning issues. Comparison experiments between the proposed approach and the original KNG algorithm are conducted using UCI benchmark datasets.

The conclusions and future work are discussed in Chapter 5.

CHAPTER 2

IMBALANCED CLASSIFICATION

2.1 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 (Duan & Keerthi, 2005) both of which use binary classifier as the base classifier. One of the most commonly used binary classifier is support vector machine (SVM) developed by Cortes and Vapnik (1995). Extensive research has explored the performance of SVM and concludes that SVM outperforms many other conventional methods in classification. For example, Bazzani1 et al. (2001) 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) (Collobert & Bengio, 2004) and linear discriminant analysis (LDA) (McLachlan, 2004). Shon et al. (2005) 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) (Scarfone & Mell, 2007), just to name a few.

While promising, SVM is known to be ineffective in dealing with imbalanced dataset (Veropoulos et al., 1999),(Wu & Chang, 2002),(He & Garcia, 2009) 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 fraud), 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 assigns 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 (Wu & Chang, 2004).

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 approach uses different sampling techniques to alter the input data distribution to reduce the degree of class imbalance. The representative methods are: undersampling(US)(Chawla, 2005), oversampling(OS)(Chawla, 2005) and synthetic minority oversampling technique (SMOTE)(Chawla et al., 2002). The preprocessing approach is usually combined with different classifiers to achieve classification. For instance, Akbani et al (2004) compare the performance of SMOTE-SVM and SMOTE-cSVM on imbalanced datasets. Instead of modifying the distribution of the input data, the algorithmic approach modifies SVM algorithm directly to make it less sensitive to class imbalance. Some examples of algorithmic approaches are: boundary movement (BM-SVM) (Wu & Chang, 2003) which shifts the decision boundary by adjusting the threshold parameter of the standard SVM; kernel modification method (Wu & Chang, 2004),(Wu & Chang, 2003) which modifies the associated kernel matrix K; and cost sensitive SVM (cSVM) (Veropoulos et al., 1999) which applies cost sensitive learning in SVM training by assigning different costs to different classes. It has been noted from the literature (Chawla et al., 2004),(Masnadi-Shirazi et al., 2012),(Maloof, 2003) that cSVM method is promising in dealing with imbalanced classification problems. This is because in Bayes decision theory, the costs of

false positive and false negative 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 threaten. On the other hand, the associate cost is much less if a non-cancer patient is diagnosed as 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) should be optimal in dealing with imbalanced dataset (Masnadi-Shirazi et al., 2012). However, many empirical studies (Wu & Chang, 2004),(Masnadi-Shirazi et al., 2012),(Cao et al., 2013) show that cSVM does not work as well as expected. As explained by Wu et al. (2004), 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 2.3.2). 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. (2012) replace the hinge lose function of cSVM with cost sensitive hinge lose function to enforce cost sensitivity. Akbani et al. (2004) combine cSVM with SMOTE method to make the

boundary well-defined. Brefeld et al. (2003) 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 discriminate positive and negative class examples directly based on the provided input data (Jordan, 2002). While being directive 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 (Jordan, 2002) 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 a generative model, Gaussian mixture model (GMM), to tackle imbalanced classification problem. GMM is chosen here because it is computationally inexpensive and has less subjective parameters to adjust (Bishop & Nasrabadi, 2006). 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 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 problem.

The rest of the paper is organized as follows: in Section 2.2 we discuss the related works. In Section 2.3 we describe the CSG algorithm in detail followed by the comparison experiments in Section 2.4. We conclude the findings and future work in Section 2.5.

2.2 Related works

2.2.1 Data preprocessing approach

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, oversampling and SMOTE.

Undersampling and oversampling are designed to rebalance the training data in different ways: undersampling reduces the size of majority class, while oversampling increases the size of minority class. The problematic consequences thus are different (Batista et al., 2004),(Holte et al., 1989),(Estabrooks et al., 2004). 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 lead to over fitting (He & Garcia, 2009). Instead of using simple duplication, SMOTE increases the size of the minority class by generating artificial data which are convex combinations of the existing ones with its nearest neighbor, thus improves learning.

2.2.2 Algorithmic approach

The algorithmic approach augments the SVM formulation to make it more tolerating to the class imbalance. Based on the parameters to be adjusted, the algorithmic approach is in general classified into three subcategories: boundary movement (BM-SVM), kernel modification and cSVM.

Let the decision function of SVM be:

$$sgn\left(f(x) = \sum_{i=1}^{n} y_i \alpha_i K(x, x_i) + b \right) \tag{2.1}$$

As seen in Equation 2.1, there are three parameters which impact the formation of the class 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 (Masnadi-Shirazi et al., 2012). The kernel modification method, Kernel-boundary alignment on the other hand, tackles the imbalanced learning problem by modifying the associated kernel matrix K. This method adjusts the class boundary by using 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 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 assigns different cost functions which are used as upper bounds to constrain α (formulations are presented in Section 2.3.2). 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 class classification is improved. Based on the Bayes decision theory, cSVM is supposed to be optimal in dealing with imbalanced classification problems. Yet, a number of empirical studies (Wu & Chang, 2004),(Masnadi-Shirazi et al., 2012),(Cao et al., 2013) show cSVM does not always have expected performance. The reason, as discussed by Wu et al. (2004), is that cSVM has issues for enforcing cost sensitivity. Though research proposes cost sensitive hinge lose function into cSVM (Masnadi-Shirazi et al., 2012), integrating SMOTE with cSVM (Akbani et al., 2004) and employing example dependent cost in cSVM training process (Brefeld et al., 2003), only discriminative models have been of the focus. 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.

2.3 Proposed algorithm: Cost SVM fusing with Gaussian Mixture Model (CSG)

2.3.1 SVM Basics

SVM finds the decision boundary by constructing the separation hyperplane with maximum margin between two classes. The data points closest to the hyperplane are called support vectors in the soft-margin formulation (Cortes & Vapnik, 1995).

$$\begin{aligned} \min \quad & \frac{1}{2} w \cdot w + C \sum_{i=1}^{n} \xi_i \\ \text{s.t.} \quad & y_i \left(w \cdot \Phi\left(x_i\right) + b \right) \geq 1 - \xi_i \\ & \xi_i \geq 0, \quad \text{i=1,...,n} \end{aligned} \tag{2.2}$$

Finding the support vectors is the key issue for the SVM classifier. This is because the decision function (in Equation 2.1) 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 Equation 2.1. 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 tuning of 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. Kernel method works by mapping the two data points from original input space (x_i and x_j) onto the high-dimensional feature space ($\phi(x_i)$ and $\phi(x_j)$). The kernel function is calculated by taking the inner product of transformed data vector:

$$K(x_i, x_j) = \left\langle \Phi(x_i), \Phi(x_j) \right\rangle = e^{\left(-\gamma \left\| x_i - x_j \right\|^2\right)}, \gamma > 0 \tag{2.3}$$

In this paper, we choose the most commonly used radial basis function (RBF) kernel (in Equation 2.3) for its good performance on various domain applications (Bishop, 1995).

The SVM algorithm predicts the label of a testing example x by computing the sign function in Equation 2.1. Instead of predicting the label, many research requires the posterior class probability P (y|x). Platt (2000) proposes a method to approximate the posterior probability by using

$$P_{A,B}(x) = P(Y = 1|X = x) = \frac{1}{1 + e^{(Af(x)+B)}} \tag{2.4}$$

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

$$(A^*, B^*) = \arg\max_{A,B} \sum_{i=1}^{n_v} \left(\frac{1+y_i}{2} \log\left(P_{A,B}(x_i')\right) + \frac{1-y_i}{2} \log\left(1 - P_{A,B}(x_i')\right) \right) \tag{2.5}$$

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

2.3.2 cSVM

In cSVM, the formulation is given as:

$$\begin{aligned} \min \quad & \frac{1}{2} w \cdot w + C \left[C^+ \sum_{\{i|y_i=+1\}}^{n_+} \xi_i + C^- \sum_{\{i|y_i=-1\}}^{n_-} \xi_i \right] \\ \text{s.t.} \quad & y_i \left(w \cdot \Phi(x_i) + b \right) \geq 1 - \xi_i \\ & \xi_i \geq 0, \quad \text{i=1,...,n} \end{aligned} \tag{2.6}$$

The Lagrangian for the cSVM formulation is:

$$L_p = \frac{w^2}{2} + C \left[C^+ \sum_{\{i|y_i=+1\}}^{n_+} \xi_i + C^- \sum_{\{i|y_i=-1\}}^{n_-} \xi_i \right] - \sum_{i=1}^{n} \alpha_i \left[y_i \left(w \cdot x_i + b \right) - 1 + \xi_i \right] - \sum_{i=1}^{n} \mu_i \xi_i \tag{2.7}$$

With the constraints on α_i as follows:

$$\begin{cases} 0 \le \alpha_i \le C^+, \text{ if } y_i = +1 \\ 0 \le \alpha_i \le C^-, \text{ if } y_i = -1 \end{cases} and \quad \sum_{i=1}^{n} \alpha_i y_i = 0 \tag{2.8}$$

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 Equation 2.8) 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 discrimination power in identifying the minority examples.

2.3.3 GMM Basics

GMM is a generative model applied in many applications such as object classification (Kim & Lee, 2012),(Wang & Ren, 2007) and speech recognition (Reynolds & Rose, 1995),(Fauve et al., 2007). Based on the training data, GMM models the probability density function of the feature vector x by using a mixture of weighted Gaussians.

$$P_{GMM}\left(x \mid y_i\right) = \sum_{m=1}^{M} c_{im} N\left(x \mid \mu_{im}, \sigma_{im}^2\right) \tag{2.9}$$

Where :

$$N\left(x \mid \mu_{im}, \sigma_{im}^2\right) = \frac{1}{\left(2\pi\sigma_{im}^2\right)^{\frac{d}{2}}} e^{\left(-\frac{1}{2}\frac{\|x-\mu_{im}\|^2}{\sigma_{im}^2}\right)} \tag{2.10}$$

c_{im}, μ_{im}, and σ^2_{im} 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 user. GMM method is an unsupervised method only reflects the intra-class information. Given a training dataset with binary class labels $\{(x_1, y_1),\ldots, (x_n, y_n)\}$, $y \in \{-1,1\}$, the data are separated into two groups according to their class label. Then the coefficients c_{im}, μ_{im}, and σ^2_{im} for each mixture are computed using an Expectation Maximization (Allouani et al., 2012) algorithm (Dempster et al., 1977). 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 were obtained, Bayesian rules can be used to calculate the posterior class probability:

$$P_{GMM}\left(y_i \mid x\right) \propto P\left(y_i\right) \sum_{m=1}^{M} w_{im} N\left(x \mid \mu_{im}, \sigma_{im}^2\right) \tag{2.11}$$

2.3.4 Proposed Algorithm: CSG

In this research, we propose a model fusion based approach to integrate discriminative algorithm (cSVM) with generative algorithm (GMM) which is explained in Figure 2-1.

Table 2-1 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

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* β_1 **/*

B*; /* search range of* β_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 */*

*Compute*Eval(); */* compute evaluation metrics: Gmean, sensitivity and specificity */*

Begin

1) **foreach** $\beta_1 \in$ A

```
2)    foreach β2 ∈ B
3)      for h= 1: NumF
4)        [PcSVM (+1|x), PcSVM (-1|x)] ← cSVMtrain (x^h_train, K, q);
5)        [c, μ, σ², PGMM (x|+1), PGMM (x|-1)] ← GMMtrain (x^h_train, n+ , n-);
6)     [PGMM (+1|x), PGMM (-1|x)] ← BayesRule (c, μ, σ², PGMM (x|+1), PGMM (x|-
1));
7)          foreach xi ∈ X^h_test
8)          Pfinal (+1|xi) = PcSVM (+1|xi) + β1 * PGMM (+1|xi) - β2 * PGMM (-1|xi);
9)              if Pfinal (+1| xi) >= PcSVM (-1|xi)
10)                 then yi pred = +1;
11)             end if
12)             otherwise yi pred = -1;
13)         end foreach
14)       end for
15)       CM ← ComputeCM (y, ypred);
16)       [Gmean, Sen, Spe] ← ComputeEval (CM);
17)       if Gmean>= bestGmean
18)          then bestGmean ← Gmean
19)       end if
20)    end foreach
21) end foreach
22) return [bestGmean, Classifier];
End
```

Figure 2-1 CSG Algorithm

Note that the parameters: RBF 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 setting is discussed in Section 2.4.

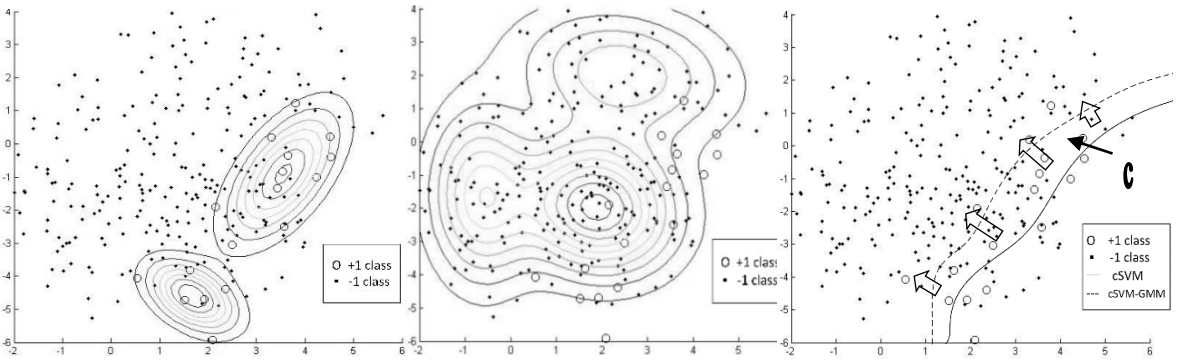
In the CSG algorithm, we combine posterior probabilities of cSVM and GMM for the final classification. The Gaussian mixtures from both positive and negative classes are used to modify the class boundary by adjusting the positive class posterior probability (in Equation 2.12). The prediction is made by comparing the posterior probability for each class.

$$P_{final}(+1|x_i) = P_{cSVM}(+1|x_i) + \beta_1 \cdot P_{GMM}(+1|x_i) - \beta_2 \cdot P_{GMM}(-1|x_i) \quad (2.12)$$

The assumption of integrating the cSVM and GMM posterior probabilities as in Equation 2.12 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 being predicted as positive, while the negative test examples remain negative in prediction.

Let us use Figure 2-2 to explain the ideas behind the CSG algorithm using simulated data.



(a) positive Gaussian mixtures (b) negative Gaussian mixtures (c) Estimated boundaries

Figure 2-2 Illustration example of CSG algorithm

As seen in Figure 2-2, circles are positive class examples and dots are negative class examples. In Figure 2-2(a) and Figure 2-2(b), CSG finds the mixture of Gaussians for positive/negative class respectively. Figure 2-2(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 Equation 2.12. 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 $P_{GMM}(-1|C) = 0.1$. If we choose $\beta_1=\beta_2=1$, according to (12), we have $P_{final}(+1|C) = 0.45 + 1*0.3 - 1*0.1 = 0.65$. 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 discrimination power in identifying the positive examples.

2.4 Experiments and results

In this section, we first test the performance of CSG using eleven KEEL benchmark datasets (Alcalá et al., 2011). Next, we use a medical imaging dataset to test the applicability of CSG on real world application. To evaluate the performance of the classifiers, we use Gmean (Kubat et al., 1997) metric which has been widely used for evaluating classifiers on imbalanced datasets (Akbani et al., 2004),(Wang, 2008),(Imam et al., 2006). Gmean is defined as $\sqrt{acc^+ * acc^-}$, 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 (Akbani et al., 2004),(Maciejewski & Stefanowski, 2011),(Hui et al., 2005), 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.

2.4.1 KEEL benchmark datasets

The eleven benchmark datasets we used in the experiments are collected from KEEL-dataset repository. The details of the datasets are listed in Table 2-2. 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 the name of the dataset indicates 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 (Akbani et al., 2004),(Cao et al.,

2013),(Hui et al., 2005). Lastly, we further explore the effect of sampling on CSG by combining SMOTE with CSG algorithm.

We use libSVM (Chang & Lin, 2011) MATLAB codes to build the SVM and cSVM models. SMOTE method is applied to preprocess the datasets using KEEL data mining software (Alcalá et al., 2011). 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 imbalance ratio (IR). The search range for q is from 1 to IR^1.4.

Table 2-2 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
pageblocks2	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 2-3 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	**0.746 ± 0.000**	0.728	0.746	**0.761 ± 0.000**
	Spe	0.876	0.708	0.688 ± 0.000	0.742	0.738	0.734 ± 0.000
	Gmean	0.674	0.707	**0.717 ± 0.000**	0.735	0.742	**0.747 ± 0.000**
haberman	Sen	0.198	0.333	**0.527 ± 0.033**	0.593	0.654	**0.679 ± 0.000**
	Spe	0.951	0.907	0.760 ± 0.026	0.742	0.680	0.671 ± 0.000
	Gmean	0.433	0.550	**0.633 ± 0.017**	0.663	0.667	**0.675 ± 0.000**
contraceptive2	Sen	0.159	0.270	**0.592 ± 0.000**	0.423	0.471	**0.588 ± 0.003**
	Spe	0.969	0.932	0.669 ± 0.000	0.807	0.768	0.710 ± 0.001
	Gmean	0.393	0.502	**0.629 ± 0.000**	0.585	0.602	**0.646 ± 0.002**
hepatitis	Sen	0.231	0.385	**0.769 ± 0.000**	0.769	0.846	**0.923 ± 0.000**
	Spe	0.985	0.955	0.821 ± 0.000	0.866	0.866	0.821 ± 0.002
	Gmean	0.477	0.606	**0.795 ± 0.000**	0.816	0.856	**0.870 ± 0.001**
yeast3	Sen	0.791	0.840	**0.945 ± 0.000**	**0.963**	**0.963**	**0.963 ± 0.000**
	Spe	0.976	0.953	0.871 ± 0.000	0.907	0.907	0.916 ± 0.000
	Gmean	0.879	0.895	**0.907 ± 0.000**	0.935	0.935	**0.939 ± 0.000**
glass2	Sen	0.000	0.118	**0.838 ± 0.025**	0.706	0.882	**0.941 ± 0.000**
	Spe	0.990	0.995	0.625 ± 0.013	0.858	0.711	0.727 ± 0.002
	Gmean	0.000	0.342	**0.724 ± 0.012**	0.778	0.792	**0.827 ± 0.001**

cleveland_0_vs_4	Sen	0.077	0.077	**0.673 ± 0.068**	0.615	0.538	**0.731 ± 0.052**
	Spe	1.000	1.000	0.585 ± 0.041	0.688	0.800	0.823 ± 0.042
	Gmean	0.277	0.277	**0.625 ± 0.012**	0.650	0.656	**0.774 ± 0.015**
pageblocks2	Sen	0.485	0.515	**0.636 ± 0.000**	0.606	**0.636**	**0.636 ± 0.000**
	Spe	0.996	0.996	0.917 ± 0.000	0.963	0.922	0.922 ± 0.000
	Gmean	0.695	0.716	**0.764 ± 0.000**	0.764	**0.766**	**0.766 ± 0.000**
flareF	Sen	0.023	0.116	**0.684 ± 0.020**	**0.907**	**0.907**	**0.907 ± 0.000**
	Spe	0.999	0.994	0.819 ± 0.011	0.833	0.833	0.836 ± 0.000
	Gmean	0.152	0.340	**0.748 ± 0.011**	0.869	0.869	**0.871 ± 0.000**
winequality_red_4	Sen	0.000	0.000	**0.509 ± 0.000**	0.585	0.585	**0.604 ± 0.000**
	Spe	1.000	1.000	0.577 ± 0.000	0.735	0.735	0.738 ± 0.000
	Gmean	0.000	0.000	**0.542 ± 0.000**	0.656	0.656	**0.668 ± 0.000**
abalone19	Sen	0.000	0.031	**0.700 ± 0.041**	**0.813**	**0.813**	**0.813 ± 0.000**
	Spe	1.000	0.990	0.608 ± 0.021	0.733	0.772	0.773 ± 0.000
	Gmean	0.000	0.176	**0.652 ± 0.016**	0.772	**0.792**	**0.792 ± 0.000**

Table 2-3 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 tends 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 discrimination 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, pageblocks2, flareF, 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. Comparing 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. Comparing 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 CSG method. Among all three SMOTE based methods, SMOTE-CSG outperforms others in nine out of eleven datasets, and in the rest 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 CSG on ten out of eleven datasets and marginal improvements on the remaining dataset (pageblocks2). 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 order to achieve better performance.

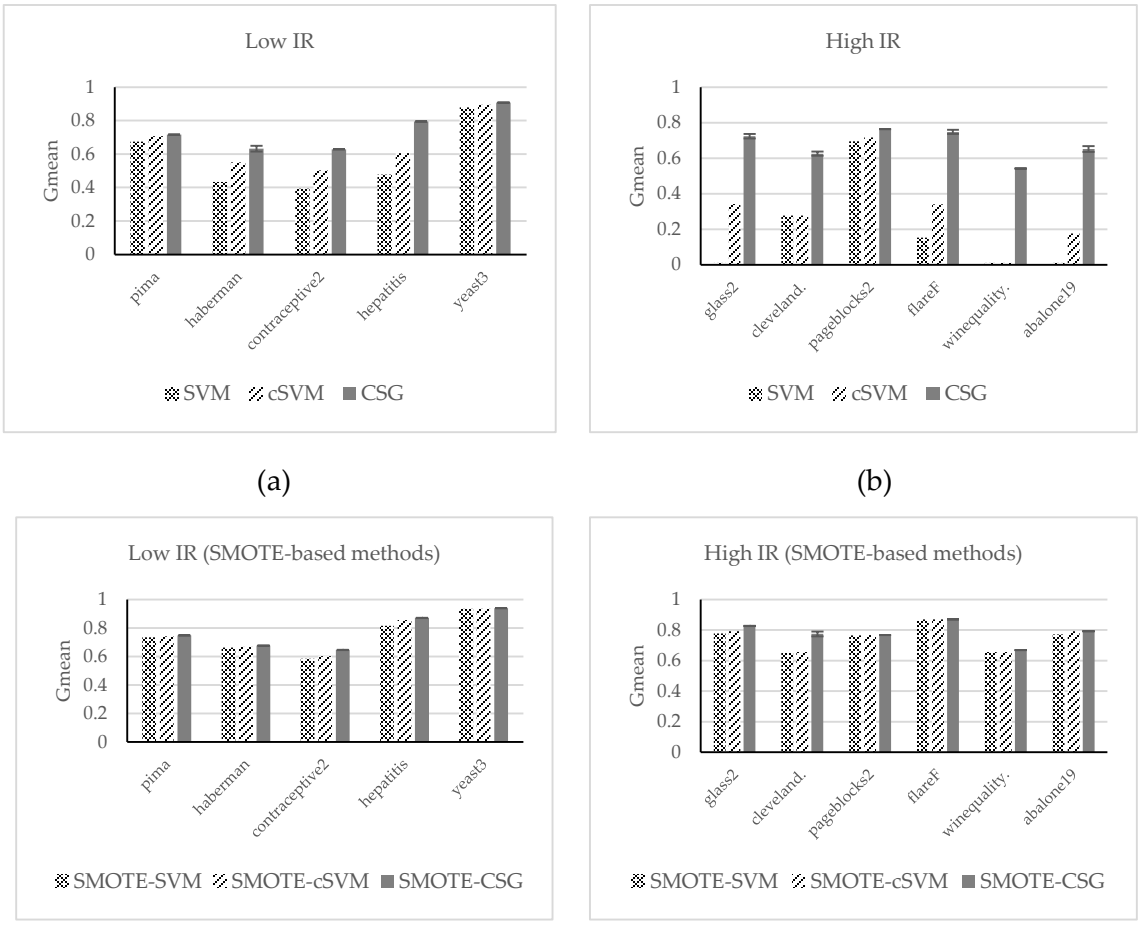


Figure 2-3 Gmeans for Low IR datasets and High IR datasets

To evaluate the effect of IR on each method, we divide the datasets into Low IR group (IR<10) and High IR group (IR>=10). Figure 2-3 shows the Gmean measures of each datasets in each group. Figure 2-3(a) and Figure 2-3(b) are the comparison of SVM, cSVM and CSG, and Figure 2-3(c) and Figure 2-3(d) are for SMOTE-SVM, SMOTE-cSVM and SMOTE-CSG. Figure 2-

3(a) and Figure 2-3(b) show that CSG greatly improves Gmean over SVM and cSVM on High IR datasets than 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 SVM and thus the class boundary is high 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 2.3.2. CSG tackles the highly 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 High IR group than on Low IR group.

For SMOTE-based methods (Figure 2-3(c) and Figure 2-3(d)), SMOTE-CSG marginally improves 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 aims to adjust the skewed class boundary would have marginal performance improvement 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 conducted and the results are shown in the next section.

2.4.2 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) (NKUDIC,

2013). It affects approximately one in eleven people in the United States (Scales et al., 2012). Each year, more than one million visits to health care providers are related to the renal stone disease (NKUDIC, 2013). Based on the chemical composition, clinically relevant renal stones can be categorized into four types: uric 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 (Eliahou et al., 2010). The commonly used stone analysis techniques include in vitro x-ray diffraction, infrared spectroscopy and polarization microscopy (Hidas et al., 2010). 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 draws many attentions (Abdel-Halim & Abdel-Halim, 2006),(Goel & Wasserstein, 2012).

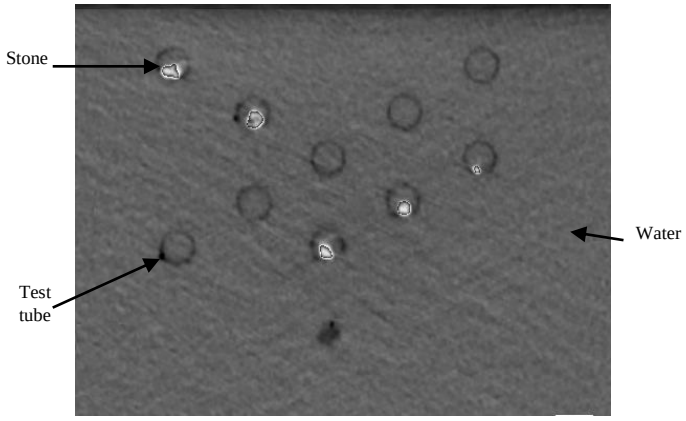


Figure 2-4 The DECT image of renal stones (phantom study)

Dual Energy CT (DECT) is a recently developed technique used for diagnostic imaging purpose. 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 (Riedel, 2010), improving noninvasive renal stone characterization (Graser et al., 2008). Figure 2-4 is an example of DECT image of renal stones from a phantom study where the stones are placed in test tubes and scanned by DECT scanner.

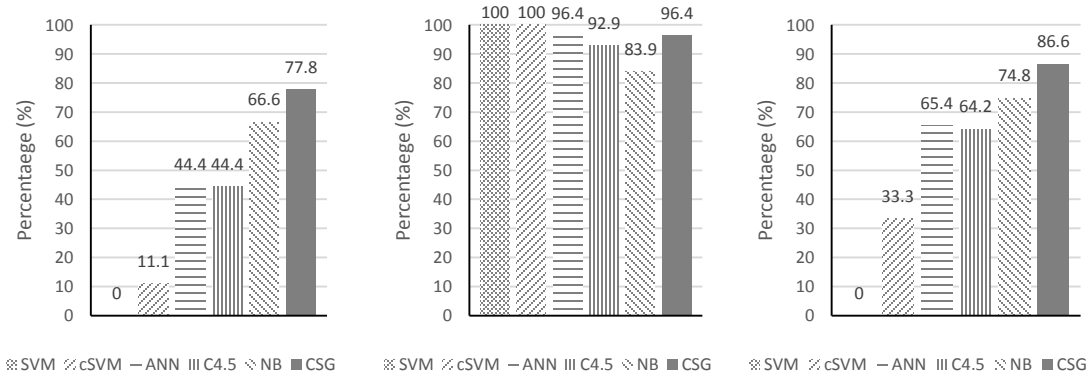
In this study, we collect 65 stones from stone analysis laboratory at Mayo Clinic Arizona. All stones are extracted from previous patients through surgical and endoscopic intervention. The chemical composition has been determined with stereo microscopy and infrared spectrophotometry. According to the 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 the 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 (Wu, 2012). Patients with cysteine stones may also need to take additional genetic screening tests other than medical treatment (Breuning & Hamdy, 2003). In this experiment, cystine stone has been selected as target class, and the rest stone types are combined as non-target class. Thus, the imbalance ratio is 6.2 (n=56 for non-cystine stones and n=9 for cystine stones). The detail of the DECT renal stone dataset is shown in Table 2-4.

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 (Dal Moro et al., 2006), artificial neural network

(ANN)(Chiang et al., 2003), C4.5 (Kaladhar et al., 2012) and NaiveBayes (NB) (Lavanya & Rani, 2011) are also implemented for comparison. The SVM, cSVM and CSG methods are performed using the same settings as in Section 2.4.1. The ANN, C4.5 and NB methods are performed using a data mining software Weka 3.6.9 (Hall et al., 2009). 5-fold stratified cross validation is applied. In addition to sensitivity, specificity and Gmean, we also use two other important evaluation metrics for medical diagnosis field: Positive Predictive Value (PPV) and Negative Predictive Value (NPV). PPV indicates the probability patients with positive screening tests truly have the disease, while NPV shows the probability patients with negative screening tests truly don't have the disease. The results are shown in Figure 2-5 and Figure 2-6.

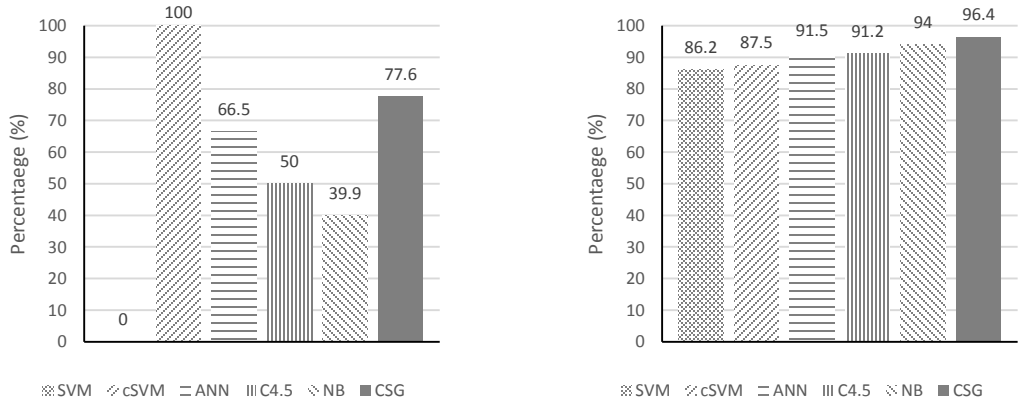
Table 2-4 The RenalStone_cys dataset

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 2-5 Sensitivity, Specificity and Gmean on RenalStone_cys dataset



(a) PPV (b) NPV

Figure 2-6 PPV and NPV on RenalStone_cys dataset

Figure 2-5 shows 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 and 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%). Compare with ANN, NB has better sensitivity (66.6%), but lower specificity (83.9%). CSG method achieves highest sensitivity (77.8%) and Gmean (86.6%) among all six methods while maintains high specificity (96.4%). CSG method also achieves second highest values in PPV (77.8%) and highest value in NPV (96.4%) according to Figure 2-6. In conclusion, CSG outperforms other five methods in classification of cystine stones.

2.5 Conclusion and discussion

In this research, we propose a model fusion based approach integrating cSVM with GMM for imbalanced classification problem. 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 CSG method to be effective in dealing with imbalanced classification problems.

We also find from the experiments that the preprocessing method SMOTE is effective in achieving better performance of CSG on imbalanced datasets. 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. Thus, the performance of CSG method can be further improved by SMOTE method.

CHAPTER 3

IMBLANCED CLASSIFICATION WITH NOISY DATASET

3.1 Introduction

Classification is a supervised learning problem which identifies the labels of new observations given a training dataset. Classification methods extract knowledge from the training dataset, and use the learned information to build models to predict the class of new observations. Therefore, the success of the classification methods highly depends on the quality of the training dataset. The real world datasets suffer from many quality issues (He & Garcia, 2009),(Seiffert et al., 2014),(Zhu & Wu, 2004). Among them, the presences of imbalance and noise are the key factors which draw great attentions (Chawla, 2005),(He & Garcia, 2009),(Sáez et al., 2013). Data imbalance occurs when one class (minority class) is greatly outnumbered by another class (majority class). Most classification methods generally tend to ignore the minority class due to the fact that majority class dominates the whole dataset. As a result, the performance of most classification methods degrades for imbalanced dataset. Data noise occurs when the data has been corrupted by various reasons such as systematic uncertainty, measurement error, human error, etc (Sáez et al., 2013),(Zhu & Wu, 2004). It can be characterized as (1) attribute noise, which refers to the corruption in the features, and (2) class noise, which occurs when the instances are incorrectly labeled. Noise may hinder the knowledge extraction from the data and thus makes the classifier less effective, particularly if the classifier is noise-sensitive.

Data imbalance and data noise often coexist in the real world datasets, that is, the dataset is imbalanced as well as noisy. Taking the CT imaging dataset as an example, the cancer patient often has a small portion of cancer tissues compared with normal tissues on the CT images which

makes the dataset imbalanced. And the reconstruction methods (Hsieh et al., 2013) used to generate the CT images comes with a systematic uncertainty making the images inherently noisy. Data imbalance affects the learning classifier by degrading the recognition power of the classifier on the minority class because the majority class dominates, while data noise affects the learning classifier by providing inaccurate information to the classifier and thus misleads the classifier. Because of these differences, data imbalance and data noise issues have been treated separately in the data mining field. Yet, such approaches ignore the mutual effects and as a result may lead to new problems. For example, data cleaning techniques (Galhardas et al., 2000) have been widely used in dealing with data noise which removes the noisy instances. If the removed instances happen to be the minority class, doing so may aggravate the level of imbalance. On the other hand, sampling method such as SMOTE (Chawla et al., 2002), which has been widely used for imbalanced datasets, may cause the data even noisier if the oversampled instances happen to be the noisy ones. One may argue that techniques may be carefully chosen to handle the data imbalance followed by data noises or vice versa, however, this two-step procedure may not be computational efficient. A desirable solution is to tackle these two issues jointly.

Most research on addressing the dataset imbalance and data noises employs discriminative models (Jordan, 2002) which are effective in finding the class boundaries (Jordan, 2002),(Lasserre, 2008) but also sensitive to data imbalance and noise since they work on the raw training data directly. Alternatively, generative models (Jordan, 2002) study the probability distribution of the training data and extract data characteristics from the training data which can be used to achieve classification, yet, may be less effective in identifying the class boundaries than discriminative models. Noticing the complementary nature of the generative and discriminative classifiers, in this research, we propose a novel generative-discriminative model

fusion based framework, termed K Nearest Gaussian (KNG). A generative classifier, Gaussian Mixture Model (GMM) is used to model the training data as Gaussian mixtures and form adjustable confidence regions of each Gaussian. GMM is chosen here due to its capability in modeling arbitrary shaped densities (Lindsay, 1995). Motivated by the idea of K-nearest neighbor (KNN), KNG finds nearest Gaussians modeled by GMM to classify the testing data instances. To test the performance of KNG, we use 7 UCI benchmark dataset. We purposely modify the datasets with added imbalance and noise. Experimental study shows that KNG method is more effective and robust than other widely used classification methods, such as Support Vector Machine (SVM) (Cortes & Vapnik, 1995), Artificial Neural Network (ANN) (Kriesel, 2011), Decision Tree (C4.5) (Quinlan, 1993) and KNN (Tan et al., 2006).

3.2 Literature review

3.2.1 Review of Techniques on Handling Imbalanced Dataset

Presently, there are a number of studies attempting to overcome the classification problem with imbalance issue. They can be categorized into two approaches: data-level approach and algorithm-level approach.

The data-level approach uses different sampling techniques to increase/decrease the size of the training data in order to generate a balanced dataset. The representative methods are: undersampling (Chawla, 2005), oversampling (Chawla, 2005) and synthetic minority oversampling technique (SMOTE) (Chawla et al., 2002). Undersampling randomly removes the data instances of majority class and thus may lead to information loss. Oversampling increases the size of the data by duplicating the existing instances of minority class which may lead to over fitting (He & Garcia, 2009). SMOTE oversamples the minority class by generating artificial data which are the convex combination of the existing ones and thus improves learning. However,

SMOTE may not perform well when the data instances used to generate new instances happen to be outliers and noisy examples (He et al., 2008). Generally, the data-level approach alters the original training data distributions to make the dataset less imbalanced. However, the change of original data may compromise the underlying knowledge of the training data and thus is expected to be avoided.

The algorithm-level approach augments the existing methods to make them less sensitive to data imbalance. Many of the existing studies tackle the imbalance data by developing extensions of existing algorithms such as SVM. For example, boundary movement (BM-SVM) (Wu & Chang, 2003) method changes the threshold value in SVM decision function to push the class boundary towards the majority class, Kernel-boundary alignment (Akbani et al., 2004) (Wu & Edward, 2004) modifies the kernel matrix used in SVM training, and cSVM applies different penalty to different classes. There are also a number of studies works on extensions of ANN to tackle the imbalance issue. For example, two-step ANN (Adam, 2012) optimizes the weights and decision threshold values by using particle swarm optimization (PSO) to recognize the minority class, HIPPO method (Japkowicz et al., 1995) trains the ANN in a novelty detection approach, and cost sensitive ANN (Berardi & Zhang, 1999) integrates the misclassification cost to ANN. In summary, most of the algorithm-level approaches are extensions of the base classifiers such as SVM and ANN. Generally, these extensions are algorithm dependent and application dependent. Thus their effectiveness is limited by certain application context.

3.2.2 Review of Techniques on Handling Noisy Dataset

The existing noise handling techniques can also be categorized into two approaches: data-level approach and algorithm-level approach.

Data-level approach, also known as noise elimination techniques, handles the noise issue by removing the noise instances from the training data. For example, AJAX method (Galhardas et al., 2000) uses four types of data transformations—mapping, matching, clustering, and merging to detect and remove the noise data, Brodley and Friedl (2011) compare the single algorithm filter, majority vote filter and consensus filter to identify and eliminate mislabeled training instances, Miranda et al. (2009) combine the prediction of four different machine learning methods to guide the noise detection and removal. These data-level approach focuses on detecting and removing the noise instances. However, these methods generally cannot distinguish the noisy cases from rare cases. The removal of rare cases may lead bias to the training data. In addition, noise instances which contain error in some features may still contain correct (and useful) information in other features. Thus, the removal of noise under this circumstances may lead to loss of valuable information.

Algorithm-level approach tackles the noisy dataset by improving the mechanism of a learning algorithm to make it less sensitive to data noise. For example, Pechenizkiy et al. (2006) use feature extraction technique as a preprocessing step in the training to diminish the effect of class noise, Mingers (1989) compares different search heuristics and stopping criteria in decision tree construction in dealing with noise data, Quinlan (1986) applies a post-pruning decision tree building procedure to deal with noise data. Although most of the algorithm-level approach does not require data preprocessing, they are generally algorithm dependent or application dependent, thus are effective only when applied under certain context.

As a summary of both imbalance handling and noise handling techniques, data-level approach alters the original distribution of training dataset which may lead to loss of valuable information and thus is expected to be avoided. The algorithm-level approach are developed based on

existing classifiers (such as SVM, ANN, C4.5), all of which employ discriminative models which are sensitive to data imbalance and noise since they work on the raw training data directly.

3.3 Proposed approach: K Nearest Gaussian (KNG)

In this study, we propose a novel method, K Nearest Gaussian (KNG). Specifically, we employ a generative model, GMM, into the training process to extract the data characteristics from training data. GMM is shown promising in dealing with data imbalance issue in our previous study (He et al., 2014) since the extracted data characteristics are expected to be less sensitive to data imbalance and noise. The idea of KNN to draw the class boundary is adopted here to differentiate the classes based on the extracted Gaussian mixtures and their corresponding confidence regions. In the following, we review the basics of KNN in section 3.3.1 and the detail of our proposed KNG in section 3.3.2.

3.3.1 K Nearest Neighbor (KNN)

KNN is a discriminative model that classifies instance based on the majority voting of its k nearest neighbor (Cover & Hart, 1967). Figure 3-1 is the illustration example of KNN algorithm.

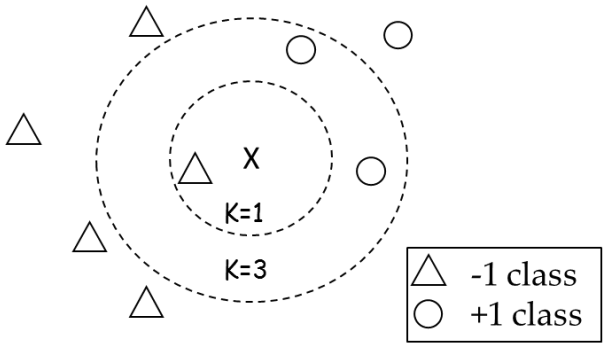


Figure 3-1 Illustration example of KNN algorithm

In Figure 3-1, X is a testing instance, circles and triangles are positive and negative class instances, respectively. KNN first calculates the distances from X to other training instances, and classify X according to the majority voting of its k nearest neighbors. K is predefined by the user. In Figure 3-1, when k=1, X is classified as negative class since the nearest neighbor is negative, while when k=3, X is classified as positive class since the majority of its three nearest neighbors is positive. Thus, X can be classified based on the neighboring instances.

3.3.2 K Nearest Gaussian (KNG)

Inspired by the KNN algorithm, which classifies an instance based on neighboring instances, we propose our KNG algorithm to tackle the imbalance and noise data issues. Instead of using the neighboring data instances, KNG uses the neighboring Gaussian mixtures to achieve classification. Specifically, KNG first applies GMM method to model the distributions of each class, and the data characteristics (such as centroid, variance) of each Gaussian can be then used to calculate the distances of the testing instance to the confidence region of each Gaussian. The smaller the distance, the higher probability that the testing instance belongs to the corresponding Gaussian distribution. Thus based on the distance to each Gaussian, the testing instance can be classified by majority voting. The data characteristics extracted by GMM method, comparing with raw training data, are expected to be less sensitive to imbalanced and noisy dataset. This makes KNG a promising method to deal with imbalanced dataset with noisy features. The notations and pseudo code of KNG algorithm can be found in Table 3-1 and Figure 3-2.

Table 3-1 Notations used in KNG 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 +1/-1 class
μ^+, σ^{2+}	Centers and variances for GMM (+1 class)
μ^-, σ^{2-}	Centers and variances for GMM (-1 class)
β_+	Confidence region adjusting coefficient (+1 class)
β_-	Confidence region adjusting coefficient (-1 class)
A	Search range of β_1
B	Search range of β_2
K	Number of nearest Gaussians
CM	Confusion matrix
EvalMetric	Evaluation metric

Input:
$\boldsymbol{X_{train}}$; / training data */*
$\boldsymbol{X_{test}}$; / testing data */*
K; / number of nearest Gaussians */*
n^+; / number of Gaussian centers for positive class */*
n^-; / number of Gaussian centers for negative class */*
A; / search range of $\boldsymbol{\beta_1}$ */*
B; / search range of $\boldsymbol{\beta_2}$ */*
Output:
bestEvalMetric; / the best Evaluation metric found */*
Classifier; / output classifier with EvalMetric*/*
Function Calls:
GMMtrain (); / train GMM classifier */*
ComputeDist_PR (); / compute point to region distance */*
Sort (); / sort the distances in ascending order */*
ComputeCM (); / compute confusion matrix */*
ComputeEval (); / compute evaluation metrics */*
Begin
1) foreach $\boldsymbol{\beta_+} \in A$
2) foreach $\boldsymbol{\beta_-} \in B$
3) for h= 1: NumF
4) $[\boldsymbol{\mu^+}, \boldsymbol{\sigma^{2+}}, \boldsymbol{\mu^-}, \boldsymbol{\sigma^{2-}}] \leftarrow$ GMMtrain ($\boldsymbol{X^h_{train}}$, n^+ , n^-);
5) foreach xi $\in \boldsymbol{X^h_{test}}$
6) foreach j $\in n^+$
7) Dist_PR (xi, j) $\leftarrow$ ComputeDist_PR (xi, $\boldsymbol{\mu_j^+}$, $\boldsymbol{\sigma_j^{2+}}$, $\boldsymbol{\beta_+}$);
8) end foreach
9) foreach q $\in n^-$

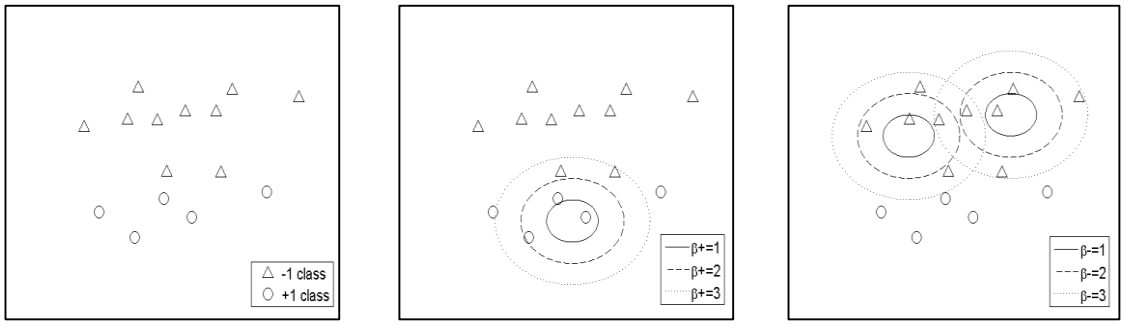
```
10)            Dist_PR (xi, q + n+) ← ComputeDist_PR (xi, μq−, σq2−, β−);
11)          end foreach
12)           [order] ← Sort (Dist_PR(xi,:));
13)           yi pred = sum(y(order(1:K)));
14)          end foreach
15)       end for
16)       CM ← ComputeCM (y, y pred);
17)       EvalMetric ← ComputeEval (CM);
18)       if EvalMetric >= bestEvalMetric
19)          then bestEvalMetric ← EvalMetric
20)       end if
21)    end foreach
22) end foreach
23) return [bestEvalMetric, Classifier];
End
```

Figure 3-2 Pseudo code for KNG Algorithm

In KNG algorithm, the ***ComputeDist_PR*** function is used to compute point to region distance, which is defined as following:

$$Dist\_PR(x_i, \mu_i, \sigma_i^2, \beta) = EuclideanDist(x_i, \mu_i) - \beta\sigma_i \quad (3.1)$$

β_+ and β_- are used to adjust the radius of the confidence regions for positive(minority) and negative (majority) Gaussians, respectively. They can be seen as weights for positive/negative classes. The unequal settings of β_+ and β_- afford the KNG algorithm the flexibility to favor one class more than another. This property is very useful in dealing with imbalanced data in which the majority class dominates. Thus, by assigning higher β_+, KNG can be more inclined to positive class and more positive instances can be recognized. This can be shown in the following illustration example. In Figure 3-3, we apply GMM to find the Gaussian mixtures for positive/negative classes. Circles are positive instances and triangles are negative instances. The Gaussian mixtures are represented by the concentric circles where different circles represent different β values.

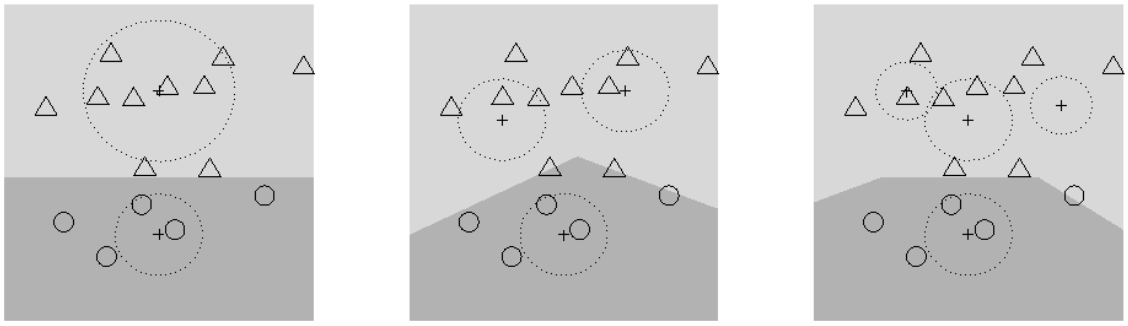


(a) Original data (b) positive Gaussian mixture (c) negative Gaussian mixture

Figure 3-3 Finding Gaussian mixtures for positive/negative classes

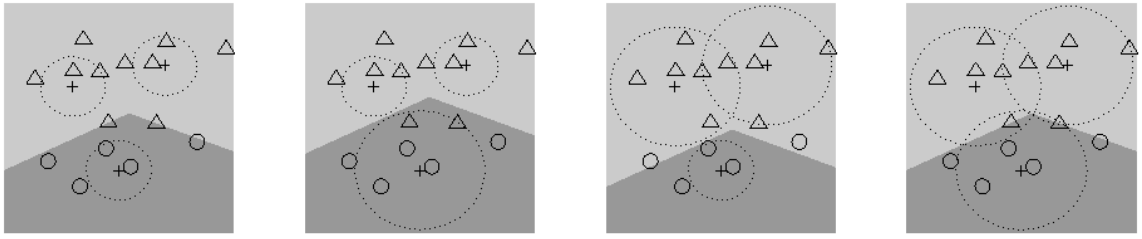
KNG algorithm has five parameters to tune in order to achieve its best classification performance: number of nearest Gaussians k, number of positive Gaussians n_+, number of negative Gaussians n_-, and adjusting factors β_+, β_-. Number of nearest Gaussians k adjusts the number of Gaussians in finding the class boundary. When k is small, only the nearby Gaussians are essential in finding the boundary, while when k is large, many far-away Gaussians are involved in finding the boundary.

Figure 3-4 shows the impact of number of Gaussians to formation of class boundary. We keep k, β_+, β_-, n_+ as constant (all equal to one) while just change n_- to see how the increasing of number of Gaussians for one class would affect the formation of class boundary. When n_- equals n_+, the two classes are linearly separated by a straight line. When we increase n_- to 2(Figure 3-4(b)), the class boundary bends more towards the positive class (dark gray region) and thus more instances can be classified as negative. In addition, the linear boundary (in Figure 3-4(a)) becomes the intersection of two linear borderlines. If we further increase n_- (Figure 3-4(c)), the class boundary can be further refined, which shows as two intersections of three linear borderlines.



(a) $n_+=1, n_-=1$ (b) $n_+=1, n_-=2$ (c) $n_+=1, n_-=3$

Figure 3-4 Impact of number of Gaussians settings to formation of class boundary



(a) $\beta_+=1, \beta_-=1$ (b) $\beta_+=2, \beta_-=1$ (c) $\beta_+=1, \beta_-=2$ (d) $\beta_+=2, \beta_-=2$

Figure 3-5 Impact of different β_+, β_- settings to formation of class boundary

Figure 3-5 shows different settings of β_+ and β_- can push of class boundary towards certain class. Figure 3-5(a) shows the positive (dark gray) and negative (light gray) class regions with the equal setting of β_+ and β_- ($\beta_+=1$, $\beta_-=1$). The border of the two regions is the class boundary. From Figure 3-5(b) and Figure 3-5(c), we observe that increasing β_+ ($\beta_+=2$, $\beta_-=1$) can push the boundary towards negative class and thus more instances can be classified as positive while increasing β_- ($\beta_+=1$, $\beta_-=2$) can push the boundary towards positive class and thus more instances can be classified as negative. As aforementioned, β_+ and β_- are used as class-specific weights to adjust the radius of the confidence region for positive/ negative Gaussians (circles with dash

line). Thus the tuning of β_+ and β_- can push the class boundary towards certain class. For imbalanced datasets, the class boundary is always skewed towards the positive class since the negative class dominates. Thus, by assigning higher β_+, KNG can push the class boundary back to positive class and more positive instances can be recognized.

3.4 Experiments and results

In this section, we test the performance of KNG using seven UCI benchmark datasets. To evaluate the performance of the classifier, we use Gmean measure which has been widely used (Akbani et al., 2004),(Wang, 2008),(Imam et al., 2006) on imbalanced classifier for its ability to evaluate the performance of a classifier on both positive and negative classes. Gmean is defined as $\sqrt{acc^+ * acc^-}$, where acc^+ (also called sensitivity) and acc^- (also called specificity) are positive and negative class prediction accuracy, respectively.

The seven benchmark datasets we used in the experiments are collected from UCI Machine Learning Repository (Bache & Lichman, 2013). We call these datasets original datasets. The details of the original datasets are summarized in Table 3-2. The original multiclass datasets are preprocessed as binary class problems, and the number in name of dataset indicates the positive class. For example, in iris2, class 2 is used as positive class and all the other classes in the original data have been joined to represent the negative class. Based on the original datasets, we generate the imbalanced datasets by randomly removing 80% of the negative class instances. Then, we further add 20% of random noise to make the datasets both imbalanced and noisy. We call these datasets are I+N datasets. The noise is introduced using the following rules as literature (Sáez et al., 2013) did:

- Class noise: 20% of the class labels are randomly replaced by the opposite class labels

- Attribute noise: 20% of each attribute data are replaced by random values from the domain (value range) of that attribute

Table 3-2 The UCI dataset used in the experiments

Dataset	#Instance	#Features	Imbalance Ratio of Original dataset	Imbalance Ratio of Imbalanced dataset
breast_cancer	683	10	1.9	9.3
diabetes	768	8	1.9	9.3
iris2	150	4	2	10.0
mammographic	830	5	1.1	5.3
yeast1	1484	8	2.2	11.0
wine2	178	13	1.5	7.6
glass3	214	9	1.8	9.2

We compare the performance of KNG method with SVM, ANN, C4.5 and KNN. These methods are chosen because they are widely used in classification problems. The KNG method is developed using MATLAB. SVM is performed using the libsvm MATLAB codes (Chang & Lin, 2011). ANN, C4.5 and KNN are performed using a machine learning software WEKA 3.6.1 (Hall et al., 2009). In this study, we use grid search technique (Bergstra & Bengio, 2012) in the parameter tuning process since it's easy to implement. The search ranges of the parameters are summarized in Table 3-3. Each method is performed using a 10 fold cross validation technique. Because of the random nature of GMM method, the result of KNG algorithm is performed 20 times for each dataset, and the mean and standard deviation are reported.

Table 3-3 Search ranges of Parameters

Method	Parameter	Range
SVM(rbf_kernel)	γ	0-512
	C	0-2048
C4.5	confidence factor	0.1-0.5
KNN	# nearest neighbor k	1-9

ANN	learning rate	0.1-0.8
	momentum	0.2(constant)
KNG	# nearest Gaussians k	1-5
	#centers(+1 class, -1 class)	1-5
	adjusting factors β+, β-	0-3

Table 3-4 Experimental results of Gmean measures

Dataset	**SVM**		**C4.5**		**ANN**		**KNN**		**KNG**	
	Orig	I+N	Orig	I+N	Orig	I+N	Orig	I+N	Orig	I+N
breast_cancer	0.976	0.787	0.959	0.000	0.962	0.517	0.970	0.457	0.977 ± 0.001	0.967 ± 0.000
diabetes	0.712	0.136	0.690	0.000	0.710	0.331	0.683	0.283	0.721 ± 0.012	0.705 ± 0.000
iris2	0.954	0.548	0.910	0.000	0.960	0.763	0.960	0.000	0.959 ± 0.013	0.941 ± 0.011
mammographic	0.836	0.111	0.838	0.435	0.816	0.237	0.800	0.564	0.797 ± 0.000	0.789 ± 0.000
yeast1	0.618	0.179	0.658	0.000	0.643	0.000	0.647	0.418	0.674 ± 0.000	0.654 ± 0.000
wine2	0.986	0.463	0.952	0.000	0.979	0.497	0.964	0.676	0.981 ± 0.000	0.957 ± 0.000
glass3	0.716	0.509	0.710	0.246	0.673	0.392	0.808	0.448	0.728 ± 0.019	0.721 ± 0.059

Table 3-4 shows the experimental results of Gmean measures for both original and I+N datasets. For original datasets, KNG achieves best Gmean in three out of seven datasets, and for iris2, wine2 datasets, KNG is just marginal worse than the best method. This shows that KNG is comparable to other major widely used classification methods on original datasets. For I+N datasets, KNG greatly outperforms other methods in all seven datasets: for breast_cancer dataset, KNG (0.967) outperforms the second best method SVM (0.787) by 0.180; for diabetes dataset, KNG (0.705) outperforms the second best method ANN (0.331) by 0.374; for iris2 dataset, KNG (0.941) outperforms the second best method ANN (0.763) by 0.178; for mammographic dataset, KNG (0.789) outperforms the second best method KNN (0.564) by 0.225; for yeast1 dataset,

KNG (0.654) outperforms the second best method KNN (0.418) by 0.236; for wine2 dataset, KNG (0.957) outperforms the second best method KNN (0.676) by 0.281; for glass3 dataset, KNG (0.721) outperforms the second best method SVM (0.509) by 0.212. In summary, the average outperformance of KNG to the second best method is 0.24. In all, KNG method is very effective in dealing with imbalanced classification problem with noisy dataset.

Table 3-5 Robustness evaluation (Change of Gmean)

Dataset	SVM	C4.5	ANN	KNN	KNG
breast_cancer	-18.9%	-95.9%	-44.5%	-51.3%	-1.0%
diabetes	-57.6%	-69.0%	-37.9%	-40.0%	-1.6%
iris2	-40.6%	-91.0%	-19.7%	-96.0%	-1.8%
Mammographic	-72.5%	-40.3%	-57.9%	-23.6%	-0.8%
yeast1	-43.9%	-65.8%	-64.3%	-22.9%	-2.0%
wine2	-52.3%	-95.2%	-48.2%	-28.8%	-2.4%
glass3	-20.7%	-46.4%	-28.1%	-36.0%	-0.7%
Average	-43.8%	-71.9%	-42.9%	-42.7%	-1.5%

We further analyze the robustness of each method using the change of Gmean as robustness measure. Change of Gmean is defined using Gmean values of I+N datasets substracts that of original datasets. This measure shows that to what extent the co-existence of imbalance and noise can affect the performance of a classifier. The smaller the value is, the more robust the model is. As seen, SVM, C4.5, ANN and KNN all show dramatic performance drop for I+N datasets compared with original datasets. However, KNG maintains the minimal change of Gmean for all seven datasets, which is shown in Table 3-5. The average change of Gmean for KNG is less than 1.5 %, which is far better than the remaining four methods. This is because the traditional classification methods, SVM, C4.5, ANN, KNN work on the training raw data directly which is sensitive to data imbalance and noise and thus their performances are highly

affected by the co-existence of imbalance and noise. However, KNG works on data characteristics extracted from the training data which are less sensitive to data imbalance and noise, and thus KNG is able to preserve the performance when imbalance and noise occurs in datasets. In conclusion, KNG has very robust performance when imbalance and noise co-exist in the datasets.

3.5 Conclusion and discussion

In this research, we propose a discriminative and generative model fusion approach, KNG, to tackle classification problems with imbalance and noise issues jointly. Instead of modeling on the raw data directly, KNG applies GMM to model the training data as Gaussian mixtures and form adjustable confidence regions of each Gaussian which are less sensitive to data imbalance and noise. The classification is achieved by majority voting of the neighboring Gaussians for the testing instances. The experimental results on seven UCI datasets show that KNG is more effective in dealing with imbalanced dataset with noisy features than other commonly used classification methods.

In the experiments, we find the performance of KNG is highly dependent on the proper settings of parameters. As we can see in Table 3-3, there are five parameters to tune in the KNG algorithm, each of which has a wide search range. The parameters are tuned through grid search method in the experiments which is criticized for being inefficient (Bergstra & Bengio, 2012). In addition, the search ranges and step size of these parameters are determined by empirical experience which may not lead to optimal model performance. Facing all the above challenges, we plan to further improve the performance of KNG algorithm by employing advanced optimizer, such as Particle Swarm Optimization (Kennedy, 2010), in parameter optimization for future research.

CHAPTER 4

FEATURE SELECTION AND PARAMETER TUNING BASED ON PARTICLE SWARM OPTIMIZATION

4.1 Introduction

In Chapter 3, we propose a K Nearest Gaussian (KNG) algorithm to tackle the problem of imbalanced classification with noisy datasets. KNG applies Gaussian Mixture Model (GMM) to model the training data as Gaussian mixtures and form adjustable confidence regions of each Gaussian. Classification is achieved in a K-nearest neighbor (KNN) manner, where the majority voting of the neighboring Gaussians is used to classify the testing instances. Although experimental studies show that KNG algorithm is very promising, two issues may hinder the performance of KNG. Firstly, KNG may suffer from the redundancy among the features in the training data. This is because redundant features increase the sparseness of the training data in the feature space and thus make the EM modeling of the GMM less effective (Figueiredo et al., 2003). As a result, the Gaussian mixtures modeled by GMM may not be robust, which may undermine the applicability of GMM. Secondly, the success of the KNG algorithm, by our empirical experience, depends heavily on the tuning of parameters. However, the parameter tuning technique, grid search, has been criticized to be both ineffective and inefficient.

To further improve the performance of KNG, a refined subset of most informative features and a finely tuned set of parameters are expected. These issues are called feature selection problem and parameter tuning problem, respectively, in machine learning field. Feature selection and parameter tuning are generally treated as separate processes. That is, by applying certain feature selection technique, a feature subset is chosen. Then based on the chosen subset, certain

parameter tuning technique is applied to achieve best model performance. In this study, we propose a Particle Swarm Optimization (PSO) based framework to perform feature selection and parameter tuning jointly. PSO is a stochastic optimization algorithm which is widely used in many domain applications (Robinson, 2005),(Chen et al., 2008),(Xue, et al., 2012). It performs search using a swarm of particles that is updated by iterations. The feature and parameter settings can be put together to form a high dimensional particle space. Thus, the best particle achieved can reflect the joint contribution of features and parameters to the optimal model performance.

The rest of the paper is organized as follows: in Section 4.2 we discuss the related works. In Section 4.3 we describe the PSO-KNG algorithm in detail followed by the comparison experiments in Section 4.4. We conclude the findings and future work in Section 4.5.

4.2 Related works

4.2.1 Feature selection techniques

Feature selection is an important issue in machine learning field, especially for classification problems. This is mainly because the redundancy among the massive features can heavily increase computational cost and also hinder classification accuracy due to the phenomena of "*curse of dimensionality*" (Chen, 2009). Feature selection techniques attempt to find a subset of features which improves or reserves classification accuracy comparing to the full feature set, but significantly reduces computational cost. The reduced set of features can also improve the interpretability of the classification results which is crucial important for many application domains, such as medical diagnosis and credit card risk management fields.

Feature selection techniques generally fall into two broad categories: filter method and wrapper method (Yu & Liu, 2003). Filter method is a type of preprocessing method which explores the general properties of the data to select subset of features without involving any classification

algorithm. The commonly used filter methods include Relief, fast correlation-based filtering (FCF), Minimum-Redundancy-Maximum-Relevance (mRmR), just to name a few. Filter method runs fast, and can be easily applied to many domain applications since it is classifier independent. However, it ignores the interaction between features and classifiers which may lead to sub-optimal classification performance. Wrapper method uses a predefined search procedure in the feature space to generate feature subsets, and the best subset is chosen based on its performance of certain predefined classifier. The commonly used wrapper methods include Sequential forward selection, SVM Recursive Feature Elimination (SVM-RFE), etc. Wrapper method shows better performance than filter methods since it considers the feature- classifier interaction by using the classifier performance as evaluation of the selected feature subsets. However, wrapper method shows higher computational cost comparing to filter method, due to the fact that predefined classifier needs to run on many different feature subsets until it finds the best subset. Besides, wrapper method is classifier dependent and thus its effectiveness is limited by certain application context.

4.2.2 Parameter tuning techniques

Parameter tuning is another important issue in machine learning field. It refers to the process of selecting proper parameters to build the classification model. Generally, the success of a classifier highly depends on the proper selection of parameters. In practice, the most commonly used parameter tuning technique is grid search method which searches the parameters exhaustively with predefined search range and step size. However, grid search has been criticized in many literatures being inefficient for its high computational cost. Besides, the predefined step size discretizes the search space of parameters which hinders its effectiveness. Gradient based method (Keerthi et al., 2007) is another commonly used parameter tuning

technique which finds the parameters in an iterative manner. The search direction and step size are determined by the gradient of some validation function (such as accuracy, Gmean measure, etc) with respect to the parameters. Gradient based method requires the validation function to be differentiable with respect to the parameter in order to calculate the gradient. However, in many applications the validation function does not meet the differentiation requirement and thus the application of gradient based method is limited by certain application context.

In this study, we use PSO method to perform feature selection and parameter tuning jointly. PSO is a population-based stochastic optimization technique. It is able to search very large space of candidate solutions with fast speed and can be used in almost any domain applications since it does not have specific requirement for the optimization problem (such as differentiable requirement). The detail of PSO is introduced in the following section.

4.2.3 Particle swarm optimization

Particle swarm optimization (PSO) is a population-based stochastic approach for optimization problems. It is first proposed by Kennedy and Eberhart (1995) to simulate the social behavior of bird flocks and fish school. PSO uses a number of particles to form a swarm, and the swarm moves around in the predefined N-dimensional search space to search for the best solution. To update the position, particles keep tracking their own best positions (personal best, ***pbest***) and also the best value of the whole swarm (global best, ***gbest***) by exchanging information with other particles. The velocity and position of each particle are updated by ***pbest*** and ***gbest*** values in each iteration. The mathematical equations for velocity and position are:

$$V_i^{t+1} = \omega^t V_i^t + c_1 r_1 \left(pbest_i^t - S_i^t\right) + c_2 r_2 \left(gbest^t - S_i^t\right) \tag{4.1}$$

$$S_i^{t+1} = S_i^t + V_i^{t+1} \tag{4.2}$$

Where i is the particle index, t is time, V_i^{t+1} is the velocity of the particle i at time t+1 , V_i^t is the velocity of the particle i at time t, w^t is the inertial weight for time t , c_1, c_2 are acceleration coefficients, r_1, r_2 are random number between 0 and 1. S_i^t is the position of particle i at time t. $\boldsymbol{pbest_i^t}$ is the ***pbest*** of particle i at time t, $\boldsymbol{gbest^t}$ is the ***gbest*** of the swarm at time t. There are three parts of the right side of Equation 4.1. The first part provides the particle the ability of exploring new search space areas. The second part is a "self- learning" part, which allows the particle to learn its personal history. The third part can be seen as a "social" part, which allows the particle to collaborate with other particles. These three parts enables the particle to stochastically search for best solution.

4.2.4 Variants of PSO

Over the years, extensive research has been made to further improve the performance of PSO. Generally, the variants of PSO fall into three broad areas. The first area of research focuses on the formulation of PSO. For example, Shi et al. (1998) introduce the inertia weight w into the original version of PSO to balance the global search and local search. Clerc and Kennedy (2002) conduct theoretical analysis on swarm dynamics and introduce constriction coefficients to control the convergence tendency of particles. Barrera and Coello (2009) use electrostatic interaction between particles to update the positions of particles to solve the multimodal optimization problem. Kennedy and Eberhart (1997) revise the position updating function using certain discretization rules to make the PSO algorithm work for discrete domain problems.

The second area of research concentrates on the learning strategies for each particle. In FIPSO (Mendes et al., 2004), a fully informed PSO is proposed where the velocity of particle is updated by all the neighbors instead of only the best performer of the swarm. In dynamic multi-swarm (DMS-PSO) (Liang & Suganthan, 2005), the particle population is divided into many small

swarms in a dynamic way that they are regrouped frequently and the information is exchanged among them. In UPSO (Parsopoulos & Vrahatis, 2005), a unified framework is proposed where the local and global variant of PSO is combined into one framework. In example-based learning PSO (ELPSO) (Huang et al., 2012), particles are learning from an example set of multiple global best particles to update the position. The diversity of the particles in the example set helps ELPSO to avoid premature convergence.

The third area of research explores the integration of PSO with other optimization techniques. Higashi and Iba (2003) combine PSO with Gaussian mutation of genetic algorithm to expand the search space. Wang et al (2007) propose a hybrid PSO (HPSO) where they add a Cauchy mutation on the global best particle so that the swarm is able to escape from local optima. Kao and Zahara (2008) combine the crossover and mutation operations in GA with the flying of particles in PSO into one optimization algorithm, which results in better solution quality and convergence rate. Hu et al. (2012) integrate PSO with multiple adaptive search methods (PSO-MAM) so that the algorithm can select the most appropriate search method for a given optimization problem. In addition, an adaptive Cauchy mutation is integrated to prevent PSO-MAM from premature convergence.

4.2.5 Applications of PSO

PSO has been widely used in many domain applications. For instance, Chen et al. (2008) apply PSO on medical imaging registration where PSO is used to adjust the parameters of the registration method to maximize the similarity measure between the reference images and testing images. Robinson (2005) applies PSO to characterize the reliability of bulk power networks. Specifically, a swarm of particles plays the role as ‘virtual power engineers’ which are used to identify vulnerable network elements that may cause wide spread damage. Chen and Zhu (2010)

apply PSO to portfolio management where PSO is used to construct optimal risky portfolios for financial investments. The experimental results show PSO outperforms other optimization method such as Genetic Algorithm. Ujjin and Bentley (2003) employ PSO to fine-tune a profile-matching algorithm of a recommender system to learn personal preference of users and provide tailored suggestions. The experiments show that PSO outperforms genetic algorithm and pearson algorithm with improved prediction accuracy and much less running time.

PSO has also been widely used to improve the performance of many classification algorithms for general classification problems. For instance, in (Garšva & Danenas, 2014), PSO is used to find the best parameter settings of SVM with different kernel functions. The experimental results on UCI datasets show that PSO outperforms other optimization methods such as direct search (grid search) and simulated annealing in terms of accuracy and sum of TP ratios. In (Vilovic et al., 2009), PSO is used to train the weights of a feedfoward ANN model. The paper concludes that PSO has faster convergence and better sum of the square measure than gradient descent method for ANN algorithm. In PSODT (Chen et al., 2014), a PSO based decision tree method is used in gene selection for cancer identification. Experiment shows that PSODT outperforms SVM and other benchmark methods in accuracy measure. In RFC+PSO (Sami et al., 2012), a random forest classifier with PSO algorithm is proposed to deal with the automatic image annotation problem. The experiments show that PSO greatly improves the performance of RFC with respect to precision and recall measures.

In this research, we apply PSO to improve the performance of KNG algorithm. Especially, we tackle two specific issues, feature selection and parameter tuning, which might have big impacts on KNG algorithm. Based on the superior performance of PSO technique on various applications, we believe that PSO would also improve the KNG algorithm with respect to

classification accuracy as well as computational cost. The detail of the proposed PSO-KNG algorithm is discussed in next section.

4.3 Proposed algorithm: PSO-KNG

In this study, we propose a PSO-based method to tackle the feature selection and parameter tuning issues jointly to improve the performance of KNG algorithm. Recall that KNG algorithm has five parameters to be finely tuned. In the grid search settings, these five parameters form five nested loops which makes the KNG algorithm computational costly. The search range and number of search steps are listed in Table 4-1.

Table 4-1 Parameters in KNG algorithm

Parameter	**Range**	**# Search Steps**
# positive GMM centers n+	[1:1:5]	5
# negative GMM centers n-	[1:1:5]	5
# nearest Gaussians k	[1,3,5]	3
adjusting factors β+	[0.1:0.1:3]	30
adjusting factors β-	[0.1:0.1:3]	30

4.3.1 Particle representation

As we mentioned before, in PSO algorithm, the swarm of particles moves around in the N-dimensional search space. Each dimension in the search space is corresponding to one digit of the particle. The structure of particles (number of digits, range of each digit) is usually defined by the user. By properly setting the structure of particle, feature selection and parameter tuning can be accomplished jointly. Figure 4-1 illustrates the particle representation. Assuming that we have an input dataset with d features, the particle can be defined as:

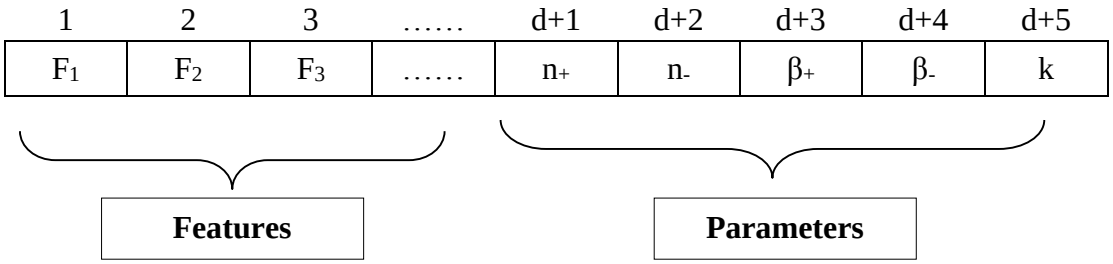


Figure 4-1 Particle representation

The formulation of the particle includes two parts, feature digits and parameter digits (shown in Figure 4-1). The feature digits are the features of the data, while the parameter digits are the parameters of KNG model. This formulation incorporates the features and parameters as one particle vector so that the search of PSO is toward the best feature and parameter combination. As a result, the feature selection and parameter tuning issues of KNG can be tackled jointly. In Figure 4-1, F_i represents the i^{th} feature in the feature set. The digits from 1 to d are the features in the input data, and the digits from d+1 to d+5 are the parameters of KNG algorithm. The F_i digits are binary digits with '1' or '0' values which refer to the selection or removal of the corresponding features. The position of Fi digits are updated using the following rules:

$$S_i^{t+1} = \begin{cases} \mathbf{1}, & \boldsymbol{if\ r} < \dfrac{\mathbf{1}}{\mathbf{1} + \boldsymbol{e}^{-v_i^{t+1}}} \\ \mathbf{0}, & \boldsymbol{otherwise} \end{cases} \tag{4.3}$$

Where r is a random number in [0,1].

4.3.2 PSO-KNG algorithm

Step 1 Input: number of particles in swarm (N), number of total iteration (iter_max), acceleration coefficients c_1 and c_2 and initial value of inertial weight w.

Step 2 At t = 0, initialize the swarm randomly.

Step 3 For each particle, select features based on the F_i values, and pass the values of parameters into KNG algorithm.

Step 4 Run KNG algorithm, obtain the values of fitness function and the corresponding pbest and gbest values. Update $pbest^t > pbest^{t-1}$, and $gbest^t > gbest^{t-1}$

Step 5 Update the particle position using Equation 4.1, Equation 4.2 and Equation 4.3.

Step 6 Repeat steps 3 and 4 until number of iteration reaches iter_max.

Step 7 Output best fitness function values with corresponding particle position.

Based on the superior ability of PSO in searching large spaces of candidate solutions, we believe that our proposed method PSO-KNG can further improve the KNG algorithm with higher classification performance and lower computational costs. To test the performance of PSO-KNG, we conduct experiments on the same datasets which has been used in Chapter 3. The details of the experiments are shown in Section 4.4.

4.4 Experiments and results

In this section, we test the performance of PSO-KNG algorithm on the same seven imbalanced and noise datasets as we used in Chapter 3. To compare with the original KNG algorithm, we mainly focus on the Gmean measure which shows the discrimination power of the model, and running time measure to show the computation cost.

We use the same search range of the parameters as in KNG algorithm. The step size is not needed since PSO can adjust the searching direction and speed automatically by learning the pbest and gbest, according to Equation 4.1. The parameters of PSO are chosen according to literatures (Xue et al., 2012),(Hu et al., 2012),(Allouani et al., 2012). The number of birds in swarm is set to 30, number of iteration (iter_max) is set to 100, acceleration coefficients c_1 and c_2 are set to 2, and the inertial weight w is updated according to the following function:

$$w^t = \left(w_{max} - \frac{w_{max} - w_{min}}{iter\_max} * t \right) \quad (4.4)$$

where w_{max} and w_{min} are set to 0.9 and 0.4, respectively.

Table 4-2 Experimental results of Gmean measures

Dataset	KNG	PSO-KNG (without FS)	PSO-KNG
breast_cancer	96.7 ± 0.0	97.6 ± 0.0	98.2 ± 0.0
diabetes	70.5 ± 0.0	70.7 ± 0.0	74.3 ± 0.0
iris2	93.4 ± 1.5	97.2 ± 0.3	99.5 ± 0.0
mammographic	78.9 ± 0.0	79.1 ± 0.0	79.2 ± 0.0
yeast1	65.4 ± 0.0	65.8 ± 0.0	66.7 ± 0.0
wine2	95.7 ± 0.0	97.0 ± 0.2	98.1 ± 0.0
glass3	72.1 ± 5.9	75.6 ± 5.2	83.5 ± 2.0

Table 4-2 shows the experimental results of Gmean measures for PSO-KNG, PSO-KNG (without FS) and original KNG algorithm. It also shows the number of original features and selected features using PSO-KNG. Both PSO based method, PSO-KNG and PSO-KNG (without FS), improves Gmean measure for all seven datasets. Comparing with original KNG algorithm, PSO-KNG (without FS) improves the learning by tuning parameters in a more refined way without predefined step size, and PSO-KNG further improves the learning by removing redundant features from the model and thus achieves the best performance among all three methods. PSO-KNG outperforms PSO-KNG (without FS) for all seven datasets, which indicates that handling the parameter tuning jointly with feature selection can achieve better model performance than dealing with parameter tuning alone. This also shows that the mutual influence

exists between data features and model parameters and should be considered in building the models.

Table 4-3 Optimized Parameters of PSO-KNG

Dataset	Number of original features	Number of selected features	k	n+	n-	β+	β-
breast_cancer	10	7	1	1	1	0.10	0.10
Diabetes	8	5	1	1	1	3.00	3.00
iris2	4	1	1	1	2	0.14	0.10
mammographic	5	3	3	2	2	1.76	3.00
yeast1	8	4	1	1	1	2.14	2.07
wine2	13	7	1	1	2	0.10	0.10
glass3	9	5	1	2	5	0.10	0.10

Table 4-3 lists the optimized parameters of PSO-KNG algorithm. PSO-KNG reduces the number of selected features to about half size of the full feature set for all seven datasets averagely, but achieves better Gmean measures for all seven datasets(as in Table 4-2). This shows that feature redundancy exists among the features and removing the redundant features improves learning. Six of seven datasets use 1 as the value for the number of nearest Gaussians k, which means the very nearest Gaussian contributes most to learning. The number of GMM centers n_+ and n_- show different combinations for different datasets, eg, (1,1), (1,2), (2,2), (2,5). However, n_- is always bigger than or equal to n_+, simply because negative class is the majority class which has more data instances than positive class. Most of the β_+ and β_- are equal or roughly equal, which shows that class boundary is mainly determined by the variance of the Gaussian mixtures.

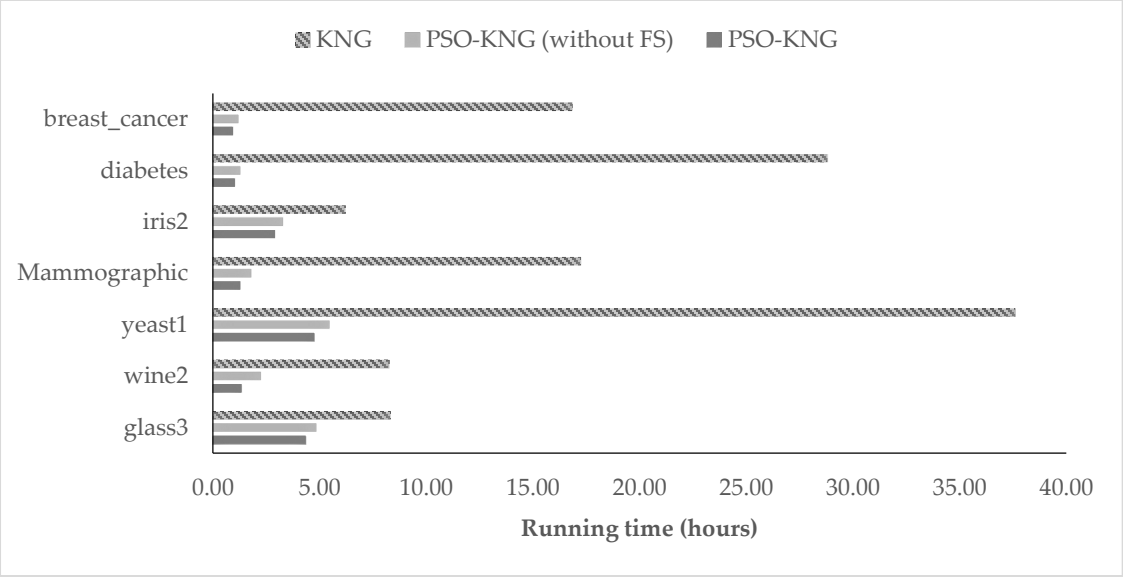


Figure 4-2 Experimental results of running time

Figure 4-2 shows the running time for each method. We can observe that the original KNG algorithm has the longest running time, while PSO based methods (with and without FS) show much less running time for all seven datasets. This is because, as aforementioned, the grid search method in original KNG algorithm uses nested loops to search for all five parameters. Each parameter setting is independent from other settings and thus the search must perform exhaustively for all possible combinations. However, PSO-KNG methods use stochastic search where the search direction and step size for each iteration can be learned based on the previous learning experience. This property makes PSO-KNG methods run much faster to find the best particle solution. PSO-KNG shows shorter running time than PSO-KNG (without FS) for all seven datasets, but not by much. This is because the reduced feature set for PSO-KNG leads to a reduced training time for KNG model in the EM modeling of Gaussian mixtures, which results in reduced total running time comparing with PSO-KNG (without FS).

In conclusion, PSO-KNG shows improved discrimination power and much lower computational cost than the original KNG algorithm.

4.5 Conclusion and discussion

In this study, we propose a PSO-KNG method to jointly tackle the feature selection and parameter issues in KNG algorithm. PSO considers the mutual influence of data features and model parameters by formulating them into one particle vector and thus can search the best feature and parameter combination jointly. Comparing with the grid search technique which is used in original KNG algorithm, PSO-KNG runs much faster since it searches the solutions stochastically where the search is toward the direction updated by the particle's learning experience of previous iterations and thus avoids exhaustive search. The experimental results show that PSO-KNG outperforms the original KNG algorithms in better Gmean measure and much lower computational cost.

CHAPTER 5

CONCLUSIONS AND FUTURE WORK

In this dissertation, we tackle the imbalanced classification problem with noisy dataset. Existing literature shows discriminative models are more effective in finding the class boundary, but the performance dropdown dramatically when imbalance and noise exists in data. On the other hand, generative models focus on modeling the data distributions which are less sensitive to data imbalance and noise, but are less effective in finding the class boundary. Due to the complementary nature of discriminative and generative models, we propose the model fusion based framework to tackle the imbalance classification problem with noisy dataset.

In Chapter 2, we focus on the general imbalanced classification problem. A comprehensive literature review on imbalanced classification methods has been made. Especially, we summarize the pros and cons of the existing studies on cost sensitive learning of support vector machines. A model fusion based method, CSG has been proposed which employs Gaussian mixture models to enforce the cost-sensitivity of the discriminative model cSVM. Experimental results on benchmark datasets and the medical imaging dataset show the effectiveness of CSG in dealing with imbalanced classification problems.

In Chapter 3, we expand the research scope to include data noise issue into the imbalanced classification problem. A comprehensive literature review on imbalance handing and noise handling techniques has been made. A model fusion based framework, KNG has been proposed which employs a generative model, GMM, to establish Gaussian mixtures and their corresponding confidence regions, and the final classification is achieved in a K nearest neighbor manner by majority voting of the neighboring Gaussians. Experimental results on benchmark

datasets show KNG greatly outperforms other commonly used classification methods in dealing with imbalanced classification problems with noisy dataset.

In Chapter 4, we address feature selection and parameter tuning issues which may hinder the performance the KNG algorithm in terms of classification accuracy and computational cost. Particle swarm optimization (PSO), a stochastic optimization technique, is comprehensively reviewed in this study and a PSO-KNG algorithm is proposed to tackle the feature selection and parameter tuning issues jointly. The experimental results show that PSO-KNG outperforms the original KNG algorithms in better Gmean measure and much lower computational cost.

This dissertation provides the ground work for discriminative and generative model fusion based framework for the problem of imbalanced classification with noisy dataset. Each chapter sets the stage for future research to take place. Specifically,

- For CSG algorithm, it follows a rear-end framework which is easy to understand and implement, but requires the fully execution of GMM and cSVM before the fusion step, which may be costly. To make the fusion in one step, we plan to explore ways of fusing GMM and cSVM in a front-end framework. A promising research direction is to combine the mathematical formulation of GMM and cSVM due to the fact that the mathematical formulation of Gaussian mixtures in GMM and that of RBF kernel in cSVM do share certain level of similarities (which can be seen in Chapter 2.3). Some work has been done by Deselaers et al. (2010) in which GMM is integrated with standard SVM in one mathematical formulation. However, their work does not take into account the cost sensitive learning, a critical issue for imbalanced classification problem. Thus, in future research, we plan to explore the ways of combining the mathematical formulations of GMM and cSVM to better handle the imbalanced classification problem.

- For KNG algorithm, although experimental results show its superior performance, we do find two issues which may hinder the performance of KNG. Thus, we employ PSO technique to further improve the performance of KNG in terms of classification accuracy and computational cost. Although the experimental results show that PSO-KNG greatly outperforms original KNG with better Gmean measure and much lower computational cost, the PSO technique we used in our study is just the basic version of PSO. It is our intention to explore various variants of PSO which can be used to better improve KNG algorithm on imbalanced classification with noisy dataset.

REFERENCES

Abdel-Halim, R. E., & Abdel-Halim, M. R. (2006). A review of urinary stone analysis techniques. Saudi medical journal, 27(10), 1462.

Adam, A., Ibrahim, Z., Shapiai, M. I., Chew, L. C., Jau, L. W., Khalid, M., & Watada, J. (2012). A TWO-STEP SUPERVISED LEARNING ARTIFICIAL NEURAL NETWORK FOR IMBALANCED DATASET PROBLEMS. INTERNATIONAL JOURNAL OF INNOVATIVE COMPUTING INFORMATION AND CONTROL, 8(5a), 3163-3172.

Akbani, R., Kwek, S., & Japkowicz, N. (2004). Applying Support Vector Machines to Imbalanced Datasets. Paper presented at the Machine Learning: ECML 2004, Berlin Heidelberg.

Alcalá, J., Fernández, A., Luengo, J., Derrac, J., García, S., Sánchez, L., & Herrera, F. (2011). KEEL Data-Mining Software Tool: Data Set Repository, Integration of Algorithms and Experimental Analysis Framework. Journal of Multiple-Valued Logic and Soft Computing, 17(2-3), 255-287.

Allouani, F., Boukhetala, D., & Boudjema, F. (2012). Particle swarm optimization based fuzzy sliding mode controller for the Twin Rotor MIMO system. Paper presented at the Electrotechnical Conference (MELECON), 2012 16th IEEE Mediterranean.

Bache, K. L., M. (2013). UCI Machine Learning Repository. Irvine, CA: University of California, School of Information and Computer Science.

Barrera, J., & Coello, C. A. C. (2009). A particle swarm optimization method for multimodal optimization based on electrostatic interaction. In MICAI 2009: Advances in Artificial Intelligence, 622-632.

Batista, G. E., Prati, R. C., & Monard, M. C. (2004). A Study of the Behavior of Several Methods for Balancing Machine Learning Training Data. ACM SIGKDD Explorations Newsletter, 6(1), 20-29.

Bazzani, A., Bevilacqua, A., Bollini, D., Brancaccio, R., Campanini, R., Lanconelli, N., ...& Romani, D. (2001). An SVM Classifier to Separate False Signals from Microcalcifications in Digital Mammograms. Physics in Medicine and Biology, 46(5), 1651.

Berardi, V. L., & Zhang, G. P. (1999). The effect of misclassification costs on neural network classifiers. Decision Sciences, 30(3), 659-682.

Bergstra, J. B., Y. (2012). Random search for hyper-parameter optimization. The Journal of Machine Learning Research, 13, 281-305.

Bishop, C. M. (1995). Neural Networks for Pattern Recognition: Oxford university press.

Bishop, C. M., & Nasrabadi, N. M. (2006). Pattern Recognition and Machine Learning. New York: springer.

Brefeld, U., Geibel, P., & Wysotzki, F. (2003). Support Vector Machines with Example Dependent Costs. Paper presented at the In Machine Learning: ECML 2003.

Breuning, M. H., & Hamdy, N. A. (2003). From gene to disease; SLC3A1, SLC7A9 and cystinuria. Nederlands tijdschrift voor geneeskunde, 147(6), 245.

Brodley, C. E., & Friedl, M. A. (2011). Identifying mislabeled training data. arXiv preprint(arXiv:1106.0219).

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

Chang, C. C., & Lin, C. J. (2011). LIBSVM : A Library for Support Vector Machines. ACM Transac-tions on Intelligent Systems and Technology, 2(27), 1-27.

Chawla, N. V. (2005). Data Mining for Imbalanced Datasets: An Overview In Data Mining and Knowledge Discovery Handbook (pp. 853-867): Springer.

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic Minority Over-sampling Technique. Journal of Artificial Intelligence Research, 16, 321-357.

Chawla, N. V., Japkowicz, N., & Kotcz, A. (2004). Editorial: Special Issue on Learning from Imbalanced Data Sets. ACM SIGKDD Explorations Newsletter, 6(1), 1-6.

Chen, K. H., Wang, K. J., Tsai, M. L., Wang, K. M., Adrian, A. M., Cheng, W. C., ... & Chang, K. S. (2014). Gene selection for cancer identification: a decision tree model empowered by particle swarm optimization algorithm. BMC bioinformatics, 15(1), 49.

Chen, L. (2009). Curse of Dimensionality. In Encyclopedia of Database Systems, 545-546.

Chen, Y., & Zhu, H. (2010). PSO heuristics algorithm for portfolio optimization. In Advances in Swarm Intelligence (pp. 183-190). Springer Berlin Heidelberg.

Chen, Y. W., Lin, C. L., & Mimori, A. (2008). Multimodal medical image registration using particle swarm optimization. Paper presented at the In Intelligent Systems Design and Applications, 2008. ISDA'08. Eighth International Conference on.

Chiang, D., Chiang, H. C., Chen, W. C., & Tsai, F. J. (2003). Prediction of Stone Disease by Discriminant Analysis and Artificial Neural Networks in Genetic Polymorphisms: a New Method. BJU International 91, 7, 661-666.

Clerc, M., & Kennedy, J. (2002). The particle swarm-explosion, stability, and convergence in a multidimensional complex space. Evolutionary Computation, IEEE Transactions on, 6(1), 58-73.

Collobert, R., & Bengio, S. (2004). Links Between Perceptrons, MLPs and SVMs. Paper presented at the In Proceedings of the Twenty-first International Conference on Machine Learning, ACM.

Cortes, C., & Vapnik, V. (1995). Support-vector Networks. Machine learning, 20(3), 273-297.

Cover, T., & Hart, P. (1967). Nearest neighbor pattern classification. Information Theory, IEEE Transactions on, 13(1), 21-27.

Dal Moro, F., Abate, A., Lanckriet, G. R. G., Arandjelovic, G., Gasparella, P., Bassi, P., ... & Pagano, F. (2006). A Novel Approach for Accurate Prediction of Spontaneous Passage of Ureteral Stones: Support Vector Machines. Kidney international, 69(1), 157-160.

De Falco, I., Della Cioppa, A., & Tarantino, E. (2007). Facing classification problems with particle swarm optimization. Applied Soft Computing, 3, 652-658.

Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum Likelihood from Incomplete Data via the EM Algorithm. Journal of the Royal Statistical Society. Series B (Methodological), 1-38.

Deselaers, T., Heigold, G., & Ney, H. (2010). Object classification by fusing SVMs and Gaussian mixtures. Pattern Recognition, 43(7), 2476-2484.

Duan, K. B., & Keerthi, S. S. (2005). Which is the best Multiclass SVM method? An Empirical Study: Springer Berlin Heidelberg.

Eliahou, R., Hidas, G., Duvdevani, M., & Sosna, J. (2010). Determination of renal stone composition with dual-energy computed tomography: an emerging application. In Seminars in Ultrasound, CT, and MRI, 31(4), 315-320.

Esmin, A. A., & Lambert-Torres, G. (2012). Application of particle swarm optimization to optimal power systems. International Journal of Innovative Computing, Information and Control, 8(3A), 1705-1716.

Estabrooks, A., Jo, T., & Japkowicz, N. (2004). A Multiple Resampling Method for Learning from Imbalanced Data Sets. Computational Intelligence, 20(1), 18-36.

Fauve, B. G., Evans, N. W., Pearson, N., Bonastre, J. F., & Mason, J. S. (2007). Influence of Task Duration in Text-independent Speaker Verification. Paper presented at the In Proc. Interspeech.

Figueiredo, M. A., Jain, A. K., & Law, M. H. (2003). A feature selection wrapper for mixtures. In Pattern Recognition and Image Analysis, 229-237.

Galhardas, H., Florescu, D., Shasha, D., & Simon, E. (2000). AJAX: an extensible data cleaning tool. ACM SIGMOD Record, 29(2), 590.

Garšva, G., & Danenas, P. (2014). Particle swarm optimization for linear support vector machines based classifier selection. Nonlinear Analysis, 19(1), 26-42.

GOEL, R., & WASSERSTEIN, A. G. (2012). Kidney Stones: Diagnostic and Treatment Strategies. Consultant, 52, 121-130.

Graser, A., Johnson, T. R., Bader, M., Staehler, M., Haseke, N., Nikolaou, K., . . . Becker, C. R. (2008). Dual energy CT characterization of urinary calculi: initial in vitro and clinical experience. Investigative radiology, 43(2), 112-119.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., & Witten, I. H. (2009). The WEKA Data Mining Software: An Update. SIGKDD Explorations, 11(1).

He, H., & Garcia, E. A. (2009). Learning from Imbalanced Data. Knowledge and Data Engineering, IEEE Transactions on, 21(9), 1263-1284.

He, H., Bai, Y., Garcia, E. A., & Li, S. (2008). ADASYN: Adaptive Synthetic Sampling Approach for Imbalanced Learning. Paper presented at the In Neural Networks, IJCNN 2008.(IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on.

He, M., Wu, T., Silva, A., Zhao, D. Y., & Qian, W. (2014). Augmenting Cost-SVM with Gaussian Mixture Models for Imbalanced Classification.

Hidas, G., Eliahou, R., Duvdevani, M., Coulon, P., Lemaitre, L., Gofrit, O. N., . . . Sosna, J. (2010). Determination of renal stone composition with dual-energy CT: in vivo analysis and comparison with x-ray diffraction. Radiology, 257(2), 394-401.

Higashi, N., & Iba, H. (2003). Particle swarm optimization with Gaussian mutation. Paper presented at the In Swarm Intelligence Symposium, 2003. SIS'03. Proceedings of the 2003 IEEE.

Holte, R. C., Acker, L., & Porter, B. W. (1989). Concept Learning and the Problem of Small Disjuncts. Paper presented at the Proceedings of the Eleventh International Joint Conference on Artificial Intelligence.

Hsieh, J., Nett, B., Yu, Z., Sauer, K., Thibault, J. B., & Bouman, C. A. (2013). Recent advances in CT image reconstruction. Current Radiology Reports, 1(1), 39-51.

Hu, M., Wu, T., & Weir, J. D. (2012). An intelligent augmentation of particle swarm optimization with multiple adaptive methods. Information Sciences, 213, 68-83.

Huang, H., Qin, H., Hao, Z., & Lim, A. (2012). Example-based learning particle swarm optimization for continuous optimization. Information Sciences, 1(182), 125-138.

Hui, H., Wang, W.-Y., & Mao, B.-H. (2005). Borderline-SMOTE: A new Over-sampling Method in Imbalanced Data Sets Learning. Paper presented at the In Advances in Intelligent Computing, Berlin Heidelberg.

Imam, T., Ting, K. M., & Kamruzzaman, J. (2006). z-SVM: An SVM for Improved Classification of Imbalanced Data. Paper presented at the In AI 2006: Advances in Artificial Intelligence, Berlin Heidelberg.

Japkowicz, N., Myers, C., & Gluck, M. (1995). A novelty detection approach to classification. Paper presented at the In IJCAI.

Jordan, A. (2002). On Discriminative vs. Generative Classifiers: A Comparison of Logistic Regression and Naive Bayes. Advances in Neural Information Processing Systems(14), 841.

Kaladhar, D., Krishna, A. R., & Varahalarao, V. (2012). Statistical and Data Mining Aspects on Kidney Stones: A Systematic Review and Meta-analysis (pp. 543): 1:543 doi:10.4172/scientificreports.

Kao, Y.-T., & Zahara, E. (2008). A hybrid genetic algorithm and particle swarm optimization for multimodal functions. Applied Soft Computing, 8(2), 849-857.

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

Keerthi, S. S., Sindhwani, V., & Chapelle, O. (2007). An Efficient Method for Gradient-Based Adaptation of Hyperparameters in SVM Models. In Advances in Neural Information Processing Systems, 673-680.

Kennedy, J. (2010). Particle swarm optimization In Encyclopedia of Machine Learning (pp. 760-766): Springer US.

Kennedy, J., & Eberhart, R. (1995). Particle Swarm Optimization. Proceedings of IEEE International Conference on Neural Networks, IV, 1942–1948.

Kennedy, J., & Eberhart, R. C. (1997). A discrete binary version of the particle swarm algorithm. In Systems, Man, and Cybernetics, 1997. Computational Cybernetics and Simulation., 1997 IEEE International Conference on, 5, 4104-4108.

Kim, D., & Lee, S. C. (2012). Pairwise Threshold for Gaussian Mixture Classification and its Application on Human Tracking Enhancement. Paper presented at the Advanced Video and Signal-Based Surveillance (AVSS) 2012 IEEE Ninth International Conference on.

Kriesel, D. (2011). A brief introduction to neural networks: Retrieved August,15.

Kubat, M., Holte, R., & Matwin, S. (1997). Learning when Negative Examples Abound. In Machine Learning*: ECML-97* (pp. 146-153). Springer Berlin Heidelberg.

Lasserre, J. (2008). Hybrid of generative and discriminative methods for machine learning PhD diss., PhD thesis: University of Cambridge.

Lavanya, D., & Rani, K. U. (2011). Performance Evaluation of Decision Tree Classifiers on Medical Datasets. International Journal of Computer Applications, 26(4), 1-4.

Lee, M. L., Ling, T. W., & Low, W. L. (2000). IntelliClean: a knowledge-based intelligent data cleaner. Paper presented at the In Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining.

Liang, J. J., & Suganthan, P. N. (2005). Dynamic multi-swarm particle swarm optimizer with local search. In Evolutionary Computation, 2005. The 2005 IEEE Congress on, 1, 522-528.

Lindsay, B. G. (1995). Mixture models: Theory, geometry, and applications: Mathematics.

Long, P. M., & Servedio, R. A. (2008). Random classification noise defeats all convex potential boosters. Paper presented at the In Proceedings of the 25th international conference on Machine learning.

Maciejewski, T., & Stefanowski, J. (2011). Local Neighbourhood Extension of SMOTE for Mining Imbalanced Data. Paper presented at the In Computational Intelligence and Data Mining (CIDM), 2011 IEEE Symposium on.

Maloof, M. A. (2003). Learning when Data Sets are Imbalanced and when Costs are Unequal and Unknown. Paper presented at the In ICML-2003 Workshop on Learning from Imbalanced Data Sets II.

Masnadi-Shirazi, H., Vasconcelos, N., & Iranmehr, A. (2012). Cost-Sensitive Support Vector Machines.

McLachlan, G. (2004). Discriminant Analysis and Statistical Pattern Recognition (Vol. 544): Wiley. com.

Mendes, R., Kennedy, J., & Neves, J. (2004). The fully informed particle swarm: simpler, maybe better. Evolutionary Computation, IEEE Transactions on, 8(3), 204-210.

Mingers, J. (1989a). An empirical comparison of pruning methods for decision tree induction. Machine learning, 4(2), 227-243.

Mingers, J. (1989b). An empirical comparison of selection measures for decision-tree induction. Machine learning, 3(4), 319-342.

Miranda, André LB, Garcia, L. P. F., Carvalho, A. C., & Lorena, A. C. (2009). Use of classification algorithms in noise detection and elimination. In Hybrid Artificial Intelligence Systems, 417-424.

NKUDIC. (2013). Kidney Stones in Adults. http://kidney.niddk.nih.gov/kudiseases/pubs/stonesadults/?control=Pubs

Parsopoulos, K. E., & Vrahatis, M. N. (2005). Unified particle swarm optimization in dynamic environments. In Applications of Evolutionary Computing, 590-599.

Pechenizkiy, M., Tsymbal, A., Puuronen, S., & Pechenizkiy, O. (2006). Class noise and supervised learning in medical domains: The effect of feature extraction. Paper presented at the In Computer-Based Medical Systems, 2006. CBMS 2006. 19th IEEE International Symposium on.

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

Quinlan, J. R. (1986). The effect of noise on concept learning Machine learning: An artificial intelligence approach (pp. 149-166): Morgan Kaufmann.

Quinlan, J. R. (1993). C4. 5: programs for machine learning: Vol. 1. Morgan Kaufmann.

Reynolds, D. A., & Rose, R. C. (1995). Robust Text-independent Speaker Identification using Gaussian Mixture Speaker Models. Speech and Audio Processing, IEEE Transactions on, 3(1), 72-83.

Riedel, M. An Introduction to Dual Energy Computed Tomography. http://ric.uthscsa.edu/personalpages/lancaster/DI2_Projects_2010/dual-energy_CT.pdf

Robinson, D. G. (2005). Reliability analysis of bulk power systems using swarm intelligence. Paper presented at the In Reliability and Maintainability Symposium, 2005. Proceedings. Annual.

Sáez, J. A., Galar, M., Luengo, J., & Herrera, F. (2013). Tackling the Problem of Classification with Noisy Data using Multiple Classifier Systems: Analysis of the Performance and Robustness. Information Sciences(247), 1-20.

Sami, M., Hassanien, A. E., El-Bendary, N., & Berwick, R. C. (2012). Incorporating random forest trees with particle swarm optimization for automatic image annotation. Paper presented at the In Computer Science and Information Systems (FedCSIS), 2012 Federated Conference on, IEEE.

Scales Jr, C. D., Smith, A. C., Hanley, J. M., & Saigal, C. S. (2012). Prevalence of kidney stones in the United States. European urology, 62(1), 160-165.

Scarfone, K., & Mell, P. (2007). Guide to Intrusion Detection and Prevention Systems (IDPS). NIST Special Publication, 800, 94.

Seiffert, C., Khoshgoftaar, T. M., Hulse, J. V., & Folleco, A. (2014). An empirical study of the classification performance of learners on imbalanced and noisy software quality data. Information Sciences, 259, 571-595.

Shi, Y., & Eberhart, R. (1998). A modified particle swarm optimizer. In Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence, 69-73.

Shon, T., Kim, Y., Lee, C., & Moon, J. (2005). A Machine Learning Framework for Network Anomaly Detection using SVM and GA. Paper presented at the In Information Assurance Workshop, 2005. IAW'05. Proceedings from the Sixth Annual IEEE SMC.

Tan, P.-N., Steinbach, M., & Kumar, V. (2006). Introduction to data mining: Pearson Addison Wesley.

Ujjin, S., & Bentley, P. J. (2003). Particle swarm optimization recommender system. Paper presented at the In Swarm Intelligence Symposium, 2003. SIS'03. Proceedings of the 2003 IEEE.

Veropoulos, K., Campbell, C., & Cristianini, N. (1999). Controlling the Sensitivity of Support Vector Machines. Paper presented at the Proceedings of the International Joint Conference on Artificial Intelligence.

Vilovic, I., Burum, N., & Milic, D. (2009). Using particle swarm optimization in training neural network for indoor field strength prediction. Paper presented at the In ELMAR, 2009. ELMAR'09. International Symposium.

Wang, H.-Y. (2008). Combination Approach of SMOTE and Biased-SVM for Imbalanced Datasets. Paper presented at the In Neural Networks, 2008. IJCNN 2008.(IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on, IEEE.

Wang, H., Liu, Y., & Zeng, S. (2007). A hybrid particle swarm algorithm with Cauchy mutation. In Swarm Intelligence Symposium, 2007. SIS 2007. IEEE, 356-360.

Wang, K., & Ren, Z. (2007). Enhanced Gaussian Mixture Models for Object Recognition using Salient Image Features. Paper presented at the Mechatronics and Automation, 2007. ICMA 2007. International Conference on.

Wu , G., & Chang, E. Y. (2002). Adaptive Feature-space Conformal Transformation for Imbalanced-data Learning. Paper presented at the MACHINE LEARNING-INTERNATIONAL WORKSHOP THEN CONFERENCE.

Wu, G., & Chang, E. Y. (2003). Class-boundary Alignment for Imbalanced Dataset Learning. Paper presented at the ICML 2003 Workshop on Learning from Imbalanced Data Sets II, Washington, DC.

Wu, G., & Chang, E. Y. (2004). Aligning Boundary in Kernel Space for Learning Imbalanced Dataset. Data Mining, ICDM'04. Fourth IEEE International Conference on. IEEE.

Wu, J. (2012). Chapter 58 – Urolithiasis Integrative Medicine, 3rd ed: WB Saunders Company.

Xiong, H., Pandey, G., Steinbach, M., & Kumar, V. (2006). Enhancing data analysis with noise removal. Knowledge and Data Engineering, IEEE Transactions on, 18(3), 304-319.

Xue, B., Zhang, M., & Browne, W. N. (2012). Multi-objective particle swarm optimisation (PSO) for feature selection. Paper presented at the In Proceedings of the fourteenth international conference on Genetic and evolutionary computation conference.

Yu, L., & Liu, H. (2003). Feature selection for high-dimensional data: A fast correlation-based filter solution. Paper presented at the In ICML.

Zhu, X., & Wu, X. (2004). Class noise vs. attribute noise: A quantitative study. Artificial Intelligence Review, 22(3), 177-210.