5-26-2006

Granular Support Vector Machines Based on Granular Computing, Soft Computing and Statistical Learning

Yuchun Tang

GRANULAR SUPPORT VECTOR MACHINES BASED ON GRANULAR COMPUTING, SOFT COMPUTING AND STATISTICAL LEARNING

by

YUCHUN TANG

Under the Direction of Yan-Qing Zhang

ABSTRACT

With emergence of biomedical informatics, Web intelligence, and E-business, new challenges are coming for knowledge discovery and data mining modeling problems.

In this dissertation work, a framework named Granular Support Vector Machines (GSVM) is proposed to systematically and formally combine statistical learning theory, granular computing theory and soft computing theory to address challenging predictive data modeling problems effectively and/or efficiently, with specific focus on binary classification problems. In general, GSVM works in 3 steps. Step 1 is granulation to build a sequence of information granules from the original dataset or from the original feature space. Step 2 is modeling Support Vector Machines (SVM) in some of these information granules when necessary. Finally, step 3 is aggregation to consolidate information in these granules at suitable abstract level. A good granulation method to find suitable granules is crucial for modeling a good GSVM.

Under this framework, many different granulation algorithms including the GSVM-CMW (cumulative margin width) algorithm, the GSVM-AR (association rule mining) algorithm, a family of GSVM-RFE (recursive feature elimination) algorithms, the GSVM-DC (data cleaning) algorithm and the GSVM-RU (repetitive undersampling) algorithm are designed for binary classification problems with different characteristics.

The empirical studies in biomedical domain and many other application domains demonstrate that the framework is promising.

As a preliminary step, this dissertation work will be extended in the future to build a Granular Computing based Predictive Data Modeling framework (GrC-PDM) with which we can create hybrid adaptive intelligent data mining systems for high quality prediction.

INDEX WORDS:

Data Mining, Machine Learning, Statistical Learning, Computational Intelligence, Granular Computing, Granular Support Vector Machines, Bioinformatics

GRANULAR SUPPORT VECTOR MACHINES BASED ON GRANULAR COMPUTING, SOFT COMPUTING AND STATISTICAL LEARNING

by

YUCHUN TANG

A Dissertation Submitted in Partial Fulfillment of Requirements for the Degree of

Doctor of Philosophy

in the College of Arts and Sciences

Georgia Stage University

2006

GRANULAR SUPPORT VECTOR MACHINES BASED ON GRANULAR COMPUTING, SOFT COMPUTING AND STATISTICAL LEARNING

by

YUCHUN TANG

Major Professor: Yan-Qing Zhang
Committee: Rajshekhar Sunderraman
Robert Harrison
Yichuan Zhao

Electronic Version Approved:

Office of Graduate Studies

College of Arts and Sciences

Georgia State University

May 2006

Acknowledgments

Firstly, my specific thanks go to my advisor, Dr. Yan-Qing Zhang, for his kind guidance and precise advisement during the process of my PhD dissertation. The dissertation would not have been possible without his helps.

Secondly, I would like to thank Dr. Rajshekhar Sunderraman, Dr. Robert Harrison, Dr. Yichuan Zhao, Dr. Zhen Huang and Dr. Xiaohua Hu for their well-appreciated support and assistance.

Finally, I want to thank my family and friends for their support and beliefs.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

Granular Computing-based Predictive Data Modeling	GrC-PDM
Granular Support Vector Machines	GSVM
Granular Support Vector Machines – Association Rule	GSVM-AR
Granular Support Vector Machines – Cumulative margin Width	GSVM-CMW
Granular Support Vector Machines – Data Cleaning	GSVM-DC
Granular Support Vector Machines – Recursive Feature Elimination	GSVM-RFE
Granular Support Vector Machines – Repetitive Undersampling	GSVM-RU

CHAPTER 1

INTRODUCTION

Knowledge discovery and data mining is known as the science of extracting useful information from large and complex datasets or databases. Specifically, predictive/supervised data mining is targeted at predicting the unknown value of a variable of interest given known values of other variables. There are two important distinct kinds of problems in predictive data mining: classification if the unknown variable is categorical; and regression if the unknown variable is real-valued [44]. For a classification problem, samples of different classes are accumulated, on which a classifier is modeled to predict future samples.

How to build effective and efficient models for supervised classification problems has been a hot research topic for a long time in data mining community and machine learning community. Effectiveness is targeted at evaluating a model in terms of accuracy (or other metrics in different contexts), while efficiency means to evaluate a model in terms of running time (or other metrics in different contexts). Usually efficiency is in inverse ratio with effectiveness: To get a more accurate classifier, a longer time is required for modeling. In many real-world applications, effectiveness is the key to evaluate if a classifier is good or not. However, in some other applications, due to real time requirement or due to very large size of the available dataset, a classifier with high efficiency is usually more preferable, at the prerequisite of not deteriorating effectiveness too much. That means a more desirable classifier in this context should run faster but still remain high accuracy. With the emergence of life science, including bioinformatics and computational biology, computational chemistry, medical informatics, the efficiency requirement is even necessary.

Besides accuracy, interpretability is another important metric to evaluate the effectiveness of a predictive data model. With good interpretability, a predictive data model (a classifier for classification problems) can be extended to build a decision support system to help humans to make decisions more reliably.

1.1 Problem definitions

In this dissertation, we focus on binary classification modeling. Although binary classification is the simplest classification problem, many works show that binary classification algorithms can be naturally extended to solve multiple classification or regression problems. (This extension itself is an interesting research topic and will not be covered in this dissertation.)

1.1.1 Binary classification

A general binary classification problem is defined as follows:

- Given l i.i.d. sample: $(x_1, y_1), (x_2, y_2), \ldots, (x_l, y_l)$ where $x_i \in R^d$, for $i = 1,2,\cdots,l$ is a feature vector of length d and $y_i = \{+1,-1\}$ is the class label (+1 for the positive class, and -1 for the negative class) for data point x_i,
- Assume the classes are mutually exclusive and exhaustive, which means every sample has one and only one class label,
- Find a classifier with the decision function $f(x,\theta)$ such that $y = f(x,\theta)$, where y is the class label for x, θ is a vector of unknown parameters in the function. These l samples are called "training data".

The performance of the classifier is usually measured in terms of misclassification error on unseen "testing data" which is defined in Eq. (1.1).

$$E(y, f(x,\theta)) = \begin{cases} 0 & if \ \ y = f(x,\theta), \\ 1 & otherwise \end{cases} \tag{1.1}$$

Based on the confusion matrix in Fig. 1.1, three metrics, named accuracy, sensitivity and specificity, are calculated to evaluate the performance:

- Accuracy is the fraction of correctly classified samples.
- Sensitivity is the fraction of the real positives that actually are correctly predicted as positives.
- Specificity is the fraction of the samples predicted as positives that really are positives.

$$accuracy = \frac{TN + TP}{TN + FN + FP + TP}. \tag{1.2}$$

$$sensitivity = \frac{TP}{TP + FN}. \tag{1.3}$$

$$specificity = \frac{TP}{TP + FP}. \tag{1.4}$$

By the definitions, the combination of sensitivity and specificity can be used to evaluate a model's balance ability so that we know if a model is biased to a special class. Notice that the sum of *FP* and *FN* is the number of misclassification errors on the unseen testing dataset.

	real negatives	real positives
predicted negatives	(TN) true negatives	(FN) false negatives
predicted positives	(FP) false positives	(TP) true positives

Figure. 1.1. confusion matrix

Recently, Area under ROC curve (AUC) has been well accepted as a better metric to evaluate a classifier's generalization capability [14]. AUC can indicate a model's balance ability between TP rate and FP rate (See Fig. 1.2) as a function of varying a classification threshold. As a result, we know if a model is biased to a special class. An area of 1 represents a perfect classification, while an area of 0.5 represents a worthless model.

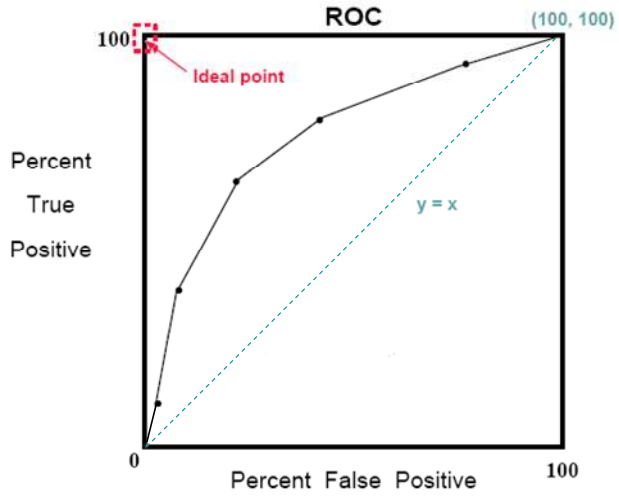


Figure. 1.2. the area under the ROC curve

$$TP - rate = \frac{TP}{TP + FN}. \tag{1.5}$$

$$FP - rate = \frac{FP}{FP + TN}. \tag{1.6}$$

There is a traditional academic point system to roughly guide the performance evaluation on the AUC metric:

$0.9 \le auc \le 1$	=	excellent	(A)
$0.8 \le auc < 0.9$	=	good	(B)
$0.7 \le auc < 0.8$	=	fair	(C)
$0.6 \le auc < 0.7$	=	poor	(D)
$0.5 \le auc < 0.6$	=	fail	(F)

1.1.2 Binary ranking

Some binary classification problem is more natural to be modeled as a binary ranking modeling. Protein homology prediction task is a good example. The target is to predict if a protein sequence is homologous to another pre-specified natural protein sequence. Because of biological complexity, it is difficult and arbitrary to say two protein sequences are absolutely homologous or not (1 or -1 is output); an output with "confidence" may be more helpful. In this way, many protein sequences could be ranked by their confidence to be homologous to the pre-specified protein sequence. As a result, biologists could quickly prioritize a list of protein sequences for further study and thus their working efficiencies can be enhanced.

A binary ranking problem is similar to a binary classification problem. The differences are

- the output is a real number in the field of [-1,1], and
- the absolute value of the output is useless. Intuitively, a good model should rank the unseen positive samples (in case of protein homology prediction, they are homologous protein sequences) close to the top and rank unseen negative samples (in case of protein homology prediction, they are non-homologous protein sequences) close to the bottom of the list.

1.1.3 Feature selection

Feature selection is closely related to binary classification. For a dataset with many input features, some features may be useless or even harmful for classification. A feature may be noisy itself, or worse, it may correlate with other features to hide real data distribution to induce overfitting.

Suppose there are d input features in the original dataset, the target of feature selection is to select d_i informative features while removing d_n non-informative features. $d_i > 0$, $d_n >= 0$,

$d_i + d_n = d$. The expectation is that the classifier modeled in the d_i feature space has better performance than the classifier modeled in the original feature space.

1.2 Challenges

With emergence of biomedical informatics, Web-based information retrieval, Internet Information Security and E-business, some new challenges are coming. Among them, noise, non i.i.d., sparseness and imbalance are four especially interesting ones and are abstract noticeable increase of interest from more and more researchers recently due to their pervasiveness in datasets from these application domains.

- Non i.i.d. (independent and identically distributed). The datasets accumulated under different contexts or even same contexts but at different time are significantly different.
- Noise. The dataset may have many noisy samples or noisy features.
- High dimensionality. The dataset may have a few samples but a huge number of features, which is known as "curse of dimensionality" in the data mining community.
- Imbalance. The dataset may have highly skewed sample distribution or highly skewed feature distribution. That means the samples/features for one class is significantly more than the samples/features for another class.

1.3 Organizations

The rest of this dissertation is organized as follows: In chapter 2, we discuss related works. After that, the general idea and framework of GSVM is presented in Chapter 3. Chapters 4-8 report five GSVM modeling algorithms, named GSVM-CMW, GSVM-AR, GSVM-RFE, GSVM-DC and GSVM-RU, respectively, for binary classification with different characteristics. Finally, we conclude this dissertation and direct the future work in Chapter 9.

CHAPTER 2

RELATED WORKS

Before introducing the GSVM framework, some related works are briefly reviewed in this chapter.

2.1 Linear classifiers

Finding the decision function $f(x,\theta)$ is equivalent to finding a decision boundary that maximally discriminate two classes in the feature space. The simplest form of a boundary is just a linear combination of the input features. This kind of classifiers is called linear classifiers. Let us imagine a simple example with only two input features, a decision boundary of a linear classifier is just a straight line. The boundary is generalized to be a hyperplane for higher dimensional feature space.

The perceptron is one of the earliest examples of a linear classifier [44]. A boundary is defined in Eq. 2.1.

$$h(x) = \sum_{i=1}^{d} w_i x_i + b = 0 . \qquad (2.1)$$

where w_i, $1 \le i \le d$ are unknown parameters called weights; b is an unknown parameters called bias.

If $h(x) > 0$, then x is assigned class label $+1$. If $h(x) < 0$, then x is assigned class label -1. That means the decision function

$$f(x,\theta) = sign(h(x)) . \qquad (2.2)$$

where θ is $(w_1, w_2 \cdots, w_d, b)$.

The values of unknown parameters are estimated by examining samples in the training dataset one by one in a way similar to gradient descent techniques. Usually the values with (possibly

locally) minimized misclassification error on the training dataset are selected as the "optimal" values.

Fisher presented one of the earliest forms of linear discriminant analysis for binary classification problems [35].

$$C = \frac{1}{n_{+1} + n_{-1}}(n_{+1}C_{+1} + n_{-1}C_{-1}), \tag{2.3}$$

$$S(w) = \frac{w'\mu_{+1} - w'\mu_{-1}}{w'Cw}. \tag{2.4}$$

where n_i is the number of samples which pertains to class i in the training dataset; C_i is $d \times d$ covariance matrix for class i estimated from the training dataset; μ_i is $d \times 1$ mean vector of class i estimated in the training dataset; $i = \{+1, -1\}$; w is a $1 \times d$ weight vector which decides the direction of a linear classifier.

The w with largest $S(w)$ defined in Equations 2.3-2.4 is taken as the weights of the classifier. The b is decided by prior probabilities of two classes or by minimizing the misclassification error. The same decision procedure as the perceptron showed in Equations 2.1-2.2 is adopted to decide a sample's class label.

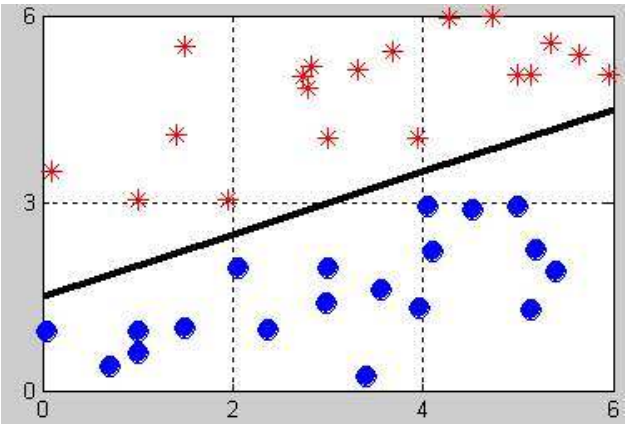


Figure. 2.1. The perceptron discriminates the classes with a linear boundary

2.2 Support Vector Machines

SVM is a superior classifier in that SVM embodies the Structural Risk Minimization (SRM) principle to minimize an upper bound on the expected risk [102,30,19,28,42].

$$R(\alpha) \le R_{emp}(\alpha) + \sqrt{\frac{h(\log(2l/h)+1) - \log(\eta/4)}{l}}, \tag{2.5}$$

where

$$R_{emp}(\alpha) = \frac{1}{2l}\sum_{i=1}^{l}\left|y_i - f(x_i,\alpha)\right| \text{ is empirical risk,} \tag{2.6}$$

h is non-negative integer called the Vapnik Chervonenkis (VC) dimension, $\eta \in [0,1]$ and the bound holds with probability $1-\eta$. α is the vector of unknown parameters.

Because structural risk is a reasonable trade-off between the error on the training dataset (the 1st factor of Eq. 2.5) and the complication of modeling (the 2nd factor of Eq. 2.5), SVM has a great ability to avoid overfitting and thus could be confidently generalized to predict new data that are not included in the training dataset.

2.2.1 Linear SVM

Geometrically, SVM modeling algorithm works for a binary classification problem by constructing a linear separating hyperplane with maximal margin as showed in Fig. 2.2. Finding the optimal separating hyperplane of SVM requires the solution to a convex quadratic programming problem, the Wolfe dual formulation of which is showed in Equations 2.7-2.9 [19].

maximize

$$L_D = \sum_i \alpha_i - \frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j K(x_i, x_j) \tag{2.7}$$

subject to

$$0 \le \alpha_i \le C, \tag{2.8}$$

$$\sum_i \alpha_i y_i = 0 \tag{2.9}$$

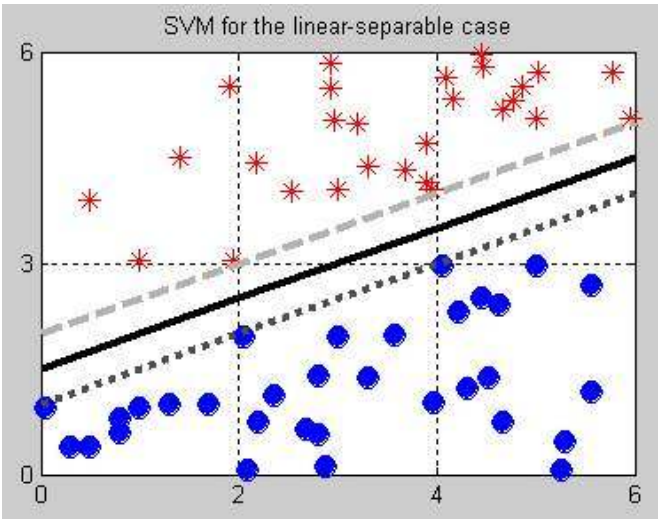


Figure. 2.2. SVM with maximal margin

For a linear separable classification problem, $K(x_i, x_j) = x_i \bullet x_j$ (the inner production of two samples). The geometry explanation is that the margin between classes could be maximized by maximizing L_D in Eq. 2.7. For linear SVM, the margin width can be calculated by the Equations 2.10-2.11.

$$w = \sum_{i=1}^{N_s} \alpha_i y_i x_i \tag{2.10}$$

$$margin\ width = 2/w \tag{2.11}$$

where N_s is the number of support vectors.

2.2.2 Kernel methods

Kernel functions are known to be a kind of elegant dimension-increasing-based methods [19] to transfer a linear non-separable problem into a linear separable problem. Nonlinear kernel

functions are introduced to implicitly map input sample from input feature space into a higher dimensional feature space, where a linear classification decision could be made. The following are some most common nonlinear kernel functions.

Polynomial kernel $K(\vec{x},\vec{y}) = (\gamma * (\vec{x} \bullet \vec{y}) + \theta)^d$ (2.12)

RBF kernel $K(\vec{x},\vec{y}) = \exp(-\gamma \parallel \vec{x} - \vec{y} \parallel^2)$ (2.13)

Sigmoid kernel $K(\vec{x},\vec{y}) = \tanh(\gamma * (\vec{x} \bullet \vec{y}) + \theta)$ (2.14)

2.3 Granular computing

Granular computing represents information in the form of some aggregates (called "information granules") such as subsets, classes, and clusters of a universe and then solves the targeted problem in each information granule [8,109,108]. On one hand, for a huge and complicated task, it embodies Divide-and-Conquer principle to split the original problem into a sequence of more manageable and smaller subtasks. On the other side, for a sequence of similar little tasks, it comprehends the problem at hand without getting buried in all unnecessary details. As opposed to traditional data-oriented numeric computing, granular computing is knowledge-oriented [108]. For a specific data mining task, we can embed prior knowledge or prior assumptions into the granular computing based data modeling process to improve performance both in terms of accuracy, efficiency and interpretability. Some formal models of information granules are:

- Set theory and interval analysis
- Fuzzy sets
- Rough sets
- Probabilistic sets
- Decision Trees
- Clusters

- Association rules

2.4 Soft computing

The basic ideas underlying soft computing in its current incarnation have links to many earlier influences, among them Prof. Zadeh's 1965 paper on fuzzy sets [114]; the 1973 paper on the analysis of complex systems and decision processes [115].

The principal constituents of soft computing (SC) are fuzzy logic (FL), neural network theory (NN) and probabilistic reasoning (PR), with the latter subsuming belief networks, evolutionary computing including DNA computing, chaos theory and parts of learning theory. For more detailed information and latest news on the soft computing, please refer to The Berkeley Initiative in Soft Computing (BISC) program (http://www-bisc.cs.berkeley.edu/).

CHAPTER 3

GRANULAR SUPPORT VECTOR MACHINES

GSVM is a hybrid model by systematically combining principles from statistical learning theory and granular computing theory [93-100].

3.1 Motivation

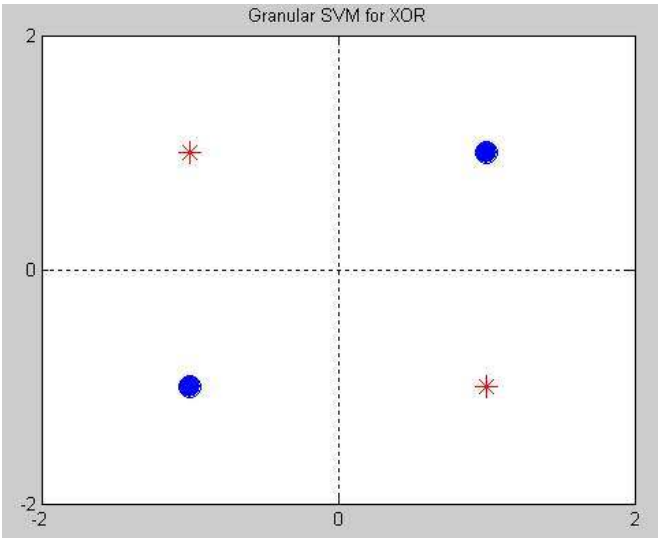


Figure. 3.1. XOR classification problem

For linear non-separable classification problems, two different ideas are usually adopted to transfer them into linear separable ones: dimension-increasing-based or partition-based (granular-computing-based). Fig. 3.1 shows the well-known XOR problem, which is a linear non-separable because there is not a line discriminating the two classes perfectly. Fig. 3.2 shows that how increasing dimensionality works to address the problem: We can add the 3rd dimension $z=xy$ to transfer the problem to be linear separable;

The dimension-increasing idea is the foundation of kernel methods. There are two problems: Firstly, up to now no one kernel method can guarantee to transfer a linear non-separable problem

into a linear separable problem, even the dimensionality is increased to be infinite. Secondly, it takes longer time to model a classifier because of increased dimensionality.

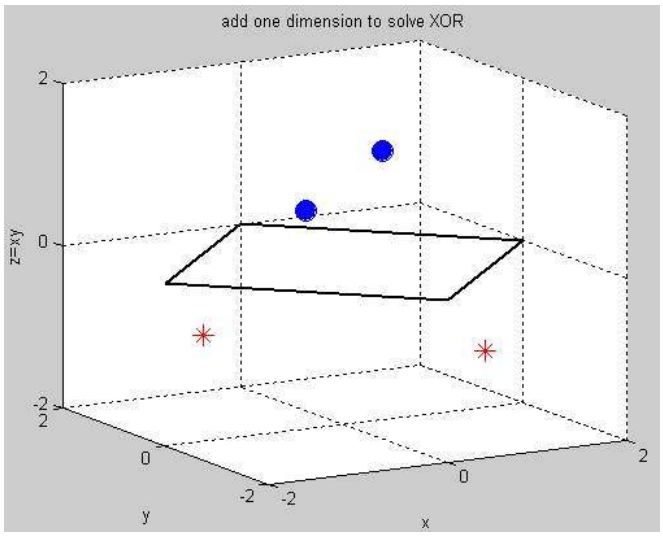


Figure. 3.2. increase one more dimension z=xy to transfer XOR problem to be linear separable

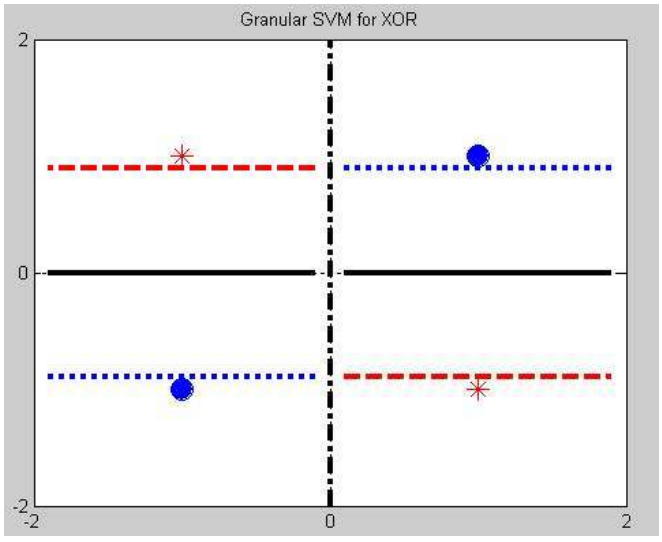


Figure. 3.3. partition the whole space to two granules to transfer XOR problem to be two smaller problems which are linear separable

Now let us see how to use the idea of granular computing to solve the XOR problem: If we can somehow split the whole space by x=0 into two subspaces (granules), the resulted classification subproblem in each granule is linear separable! And hence we can build two linear SVMs in these two granules. This process is showed in Fig. 3.3.

This simple XOR example stimulates us to design a novel classification model named Granular Support Vector Machines (GSVM).

3.2 GSVM Modeling

There are mainly three steps for GSVM modeling.

The first step is granulation. Many algorithms, such as Decision Trees, Association Rules, and Clustering algorithms, can be used to split the original feature space into a sequence of subspaces. Some other data mining techniques, including sampling, bagging, and boosting, can be used to build a sequence of subsets from the original dataset. Besides these generic algorithms, for a specific classification task, we even can design new granulation methods to embed prior knowledge or prior assumptions into the granulation process. Notice that some information granules may overlap so that some samples may appear in multiple information granules. Also notice that one granule (the original dataset or the original feature space) is already the optimal so that we even don't need granulation in this special case.

In general, multiple information granules are created at the first step. After that, some classifiers are modeled to solve sub classification problems in these granules. Here any classification algorithms can be used. However, we adopt SVM due to its strong statistical background and superior performance on many real world classification applications.

After that, in each information granule, we have raw data, we have a SVM classifier, and we even can extract knowledge in the form of a few critical rules or cases. So the final step is to

aggregate these information to build a single GSVM model. The aggregation can be executed in different abstract levels such as data fusion, decision fusion, knowledge fusion, or hybrid information fusion.

3.3 Comparison with SVM

SVM is inherently a contiguous model in that it uses a single contiguous hyperplane to halve the whole feature space. Is it reasonable to always assume that the classification boundary is contiguous? Here we argue that the boundary maybe discrete for many classification problems. So if we can somehow correctly split the whole feature space into a set of subspaces (information granules) and then build a SVM for some mixed ones of the subspaces, the resulting model is expected to capture the inherent data distribution of the classification problem at hand more accurately. Even for a contiguous classification boundary, the boundaries from suitably built subspaces could approximate it with enough accuracy. Currently, for the discrete or other linear non-separable classification problems, the only method is to use some kernel function to map the data to a new feature space in which the data is expected to be linear separable. But up to now no kernel function can guarantee the "linear separability".

SVM tries to find the optimal decision boundary by extracting important samples called Support Vectors (SVs). However, by extracting SVs in just one granule (the whole feature space, actually), it is prone to be affected by noisy samples or noisy features. Furthermore, whether a sample is extracted to be a SV is highly sensitive to parameters of the SVM. As a result, some important samples may be lost. If we can split the whole feature space into several overlapping granules, in each of which important samples are extracted as SVs, each sample can achieve more than one opportunity to be extracted. This way, information loss is decreased.

GSVM is a model which systematically and formally combines the principles from statistical learning theory, granular computing theory and soft computing theory. It works by building a sequence of information granules and then building SVM in some of information granules on demand.

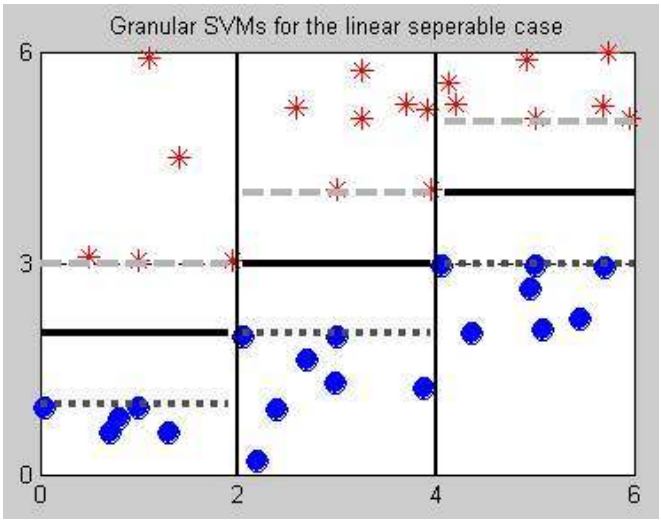


Figure. 3.4. GSVM can get better generalization by splitting the whole feature space with x=2 and x=4. As a result, there are three SVMs for the three information granules

Some potential advantages of GSVM are

- GSVM can get better generalization in a linear separable classification problem. Compared to Fig. 2.2, Fig. 3.4 shows that GSVM may improve the generalization capability by enlarging the margin width.
- GSVM can increase a linear non-separable problem's "linear separability", or even transfer a linear non-separable problem to totally linear separable as the XOR example demonstrates. That means GSVM could be a potential alternative to kernel functions by transferring a linear non-separable classification problem to a set of linear separable subproblems. In fact, these two methods are not contradictory so we can combine granule

functions and kernel functions in a GSVM to achieve better separability. One way is mapping the data to a new feature space with some kernel function at first, and then a GSVM is modeled in the new feature space; Another way is splitting the original feature space into a set of information granules at first, and then using different kernel functions to map the data in these information granules to different new feature spaces separately.

- In many real world data mining applications, what people expect is not only to get a model with small prediction error, but also to explain the reason why it works so well. As we know, SVM is almost unable to provide this kind of information. However, a few critical rules or cases can be extracted from information granules so that GSVM decision process is similar to human understandable Rule-Based Reasoning (RBR) or Case-Based Reasoning (CBR).
- Compared to SVM, GSVM is more possible to grasp inherent data distribution by trade-off between local significance of a subset of data and global correlation among different subsets of data. And hence, GSVM is expected to be effective to improve classification performance.
- GSVM may speed up the modeling process by eliminating redundant data locally. Moreover, GSVM is easy to be parallelized so that it is more efficient to be applied to huge data classification problems, which are common in biomedical application domain.
- Like SVM, GSVM could also be applied to multiple classification or regression problems without or with small modifications.

However, building suitable information granules is far from a trivial task. The key is to build the information granules somehow reasonably and effectively. Many questions need to be answered during modeling a GSVM:

- Top-down or bottom-up? Begin from the whole feature space and then gradually split it into smaller spaces (top-down)? Or begin from creating many tiny information granules and then gradually combine them into larger spaces?
- What is the stop condition? What time should we stop splitting or combining? How do we know a splitting or combining will result in overfitting? From the granular computing viewpoint, how do we know we already get optimal information granules? Notice here maybe the original whole feature space is itself an optimal granule so we even don't need to split it.
- If top-down, how to find the feature(s) the splitting hyperplane should be based on? Is it reasonable to split a space by a hyperplane orthogonal to a single feature? Or select a splitting hyperplane based on a group of features?
- After selecting the splitting feature (or features), how to decide the direction and the bias parameters of the splitting hyperplane $wx + b$?

Obviously, GSVM modeling is knowledge oriented and data dependent. There are no general answers for these questions. And hence, many different GSVM modeling algorithms are designed for binary classification with different characteristics.

CHAPTER 4

GSVM-CMW

This chapter proposes GSVM-CMW (cumulative margin width) algorithm for general binary classification problems.

4.1 GSVM-CMW algorithm

GSVM-CMW is a simple but efficient modeling method for building a linear GSVM in the top-down way. The hyperplane used to split the feature space is selected by extending statistical margin maximization principle as showed at section 2.2.1.

The margin between classes could be maximized by maximizing L_D in Eq. 2.7. For linear SVM, the margin width can be calculated by the Eq. 2.10-2.11.

Here we will find granules splitting hyperplane by extending this principle. Suppose we split the whole feature space by the hyperplane $x^k = c$, where x^k is the k^{th} input feature, and $c \in R$ is a constant value, and then we build two SVMs, named SVM1 and SVM2, for 2 subspaces, named subspace1 and subspace2, respectively. For comparison, we also build a SVM called SVM0 in the whole feature space. We define a "cumulative margin width" (CMW) as follows.

For linear SVMs, we define *CMW* in Eq. 4.1. The geometrical explanation is straightforward: the hyperplane with the smallest cumulative weight is selected to be the splitting hyperplane.

$$CMW = \frac{1}{w_1 + w_2} \tag{4.1}$$

where w_1, w_2 are weights calculated by Eq. 2.10 for SVM1,SVM2, respectively.

Unfortunately, for SVMs with nonlinear kernels, the margin width could not be directly calculated in this simple way because the separating hyperplane resides in an implicit high-

dimension feature space. Here we will define *CMW* in Eq. 4.2 to get sub-maximized margins because the margin between classes could be maximized by maximizing L_D in Eq. 2.7.

$$CMW = \frac{l_1}{l}(L_{D1} - L_{D01}) + \frac{l_2}{l}(L_{D2} - L_{D02}) \tag{4.2}$$

where $l = l_1 + l_2$,

l_1 is the number of training data in subspace1,

l_2 is the number of training data in subspace2,

L_{D1} is L_D of SVM1 calculated by Eq. 2.7,

L_{D2} is L_D of SVM2,

L_{D01} is L_D of SVM0 by only counting SVs in subspace1,

L_{D02} is L_D of SVM0 by only counting SVs in subspace2.

Now the problem to find the optimal splitting hyperplane is transformed to find the hyperplane with largest *CMW* value in Eq. 4.1 for linear kernel or Eq. 4.2 for non-linear kernel. Here many search algorithms can be candidates, for example, genetic algorithms. But for fairness of comparison, we use grid search heuristic to find a suboptimal solution [46]. Notice here we only search the hyperplane orthogonal to a single feature to simplify the searching process. We equally select *m* constants

$$c_1, c_2, \cdots, c_m \in (x^{k\_\min}, x^{k\_\max}) \text{ for each feature } x^k \tag{4.3}$$

where $x^{k-\min}$ is the minimum value of x^k in the training dataset, while $x^{k-\max}$ is the maximum value.

As a result, there are altogether *dm* hyperplane candidates. The hyperplane with largest *CMW* value will be selected as the splitting hyperplane. If there are many hyperplanes with the largest

CMW value, the hyperplane with the largest training accuracy is selected because we can expect the better accuracy could be generalized to new unseen data.

The above process can recursively applied to subspace1 and subspace2. Here once again for simplicity, we apply the above process only once to halve the original whole feature space and build one SVM for each subspace.

4.2 GSVM-CMW simulation

4.2.1 Environment

The hardware we used is a PC with P4-2.8MHz CPU and 256M memory. The software we used is OSU SVM Classifier Matlab Toolbox [67] which implements a Matlab interface to LIBSVM [23]. All simulations in this dissertation work are under this environment, so we will not repeat it in the following chapters.

4.2.2 Data Sets

TABLE 4.1
CHARACTERISTICS OF DATASETS USED FOR EXPERIMENTS

Dataset	Size	Attr	Ratio
Wisconsin Breast Cancer	683	9	239:444
Cleveland heart-disease	297	13	160:137
BUPA Liver Disorders	345	5	169:176

Note 1: Size = # of cases after removing cases with missing data, Attr = # of input features, Ratio = # of positive cases : # of negative cases.
Note 2: 16 cases in Wisconsin Breast Cancer and 6 cases in Cleveland heart-disease with missing values are removed.

Three public medical binary classification data from UCI data mining repository [68] are used for comparison:

- Wisconsin Breast Cancer dataset
- Cleveland heart-disease dataset

- BUPA Liver Disorders dataset

The detailed characteristics of datasets are listed in Table 4.1.

TABLE 4.2
TESTING ACCURACY COMPARISON ON WISCONSIN BREAST CANCER DATASET WITHOUT KERNEL MAPPING

Trial	Testing accuracy of linear SVM	Splitting hyperplane of linear GSVM	Testing accuracy of linear GSVM
1	0.9559	pc1= 0.2083	0.9780
2	0.9649	pc1= 0.2161	0.9649
3	0.9649	pc1= 0.2083	0.9868
4	0.9778	pc1= -0.2083	0.9822
5	0.9565	pc1= -0.2083	0.9739
Mean	**0.9640**		**0.9772**
Std	**0.0089**		**0.0084**

TABLE 4.3
TESTING ACCURACY COMPARISON ON CLEVELAND HEART-DISEASE DATASET WITHOUT KERNEL MAPPING

Trial	Testing accuracy of linear SVM	Splitting hyperplane of linear GSVM	Testing accuracy of linear GSVM
1	0.8776	pc10= -0.7093	0.8776
2	0.8100	pc1= -0.3349	0.7900
3	0.8586	pc4= -1.4573	0.8889
4	0.7959	pc10= -0.7602	0.7857
5	0.8500	pc10= -0.6992	0.8600
Mean	**0.8384**		**0.8404**
Std	**0.0342**		**0.0491**

TABLE 4.4
TESTING ACCURACY COMPARISON ON BUPA LIVER DISORDERS DATASET WITHOUT KERNEL MAPPING

Trial	Testing accuracy of linear SVM	Splitting hyperplane of linear GSVM	Testing accuracy of linear GSVM
1	**0.4897**	pc5= 0.0938	**0.4966**

TABLE 4.5
TESTING ACCURACY COMPARISON ON WISCONSIN BREAST CANCER DATASET WITH RBF KERNEL MAPPING

Trial	Testing accuracy of RBF SVM	Splitting hyperplane of RBF GSVM	Testing accuracy of RBF GSVM
1	0.9692	pc1= -1.1570	0.9604
2	0.9737	pc1= -1.1396	0.9605
3	0.9781	pc1= -1.1570	0.9693
4	0.9733	pc1= -1.1570	0.9733
5	0.9565	pc1= 1.1570	0.9652
Mean	**0.9702**		**0.9657**
Std	**0.0083**		**0.0056**

TABLE 4.6
TESTING ACCURACY COMPARISON ON CLEVELAND HEART-DISEASE DATASET WITH RBF KERNEL MAPPING

Trial	Testing accuracy of RBF SVM	Splitting hyperplane of RBF GSVM	Testing accuracy of RBF GSVM
1	0.7959	pc1= -0.5223	0.8265
2	0.8200	pc1= -0.3349	0.8000
3	0.7273	pc1= -0.5277	0.8687
4	0.7551	pc1= -0.3349	0.7857
5	0.8600	pc1= -0.3411	0.8900
Mean	**0.7917**		**0.8342**
Std	**0.0524**		**0.0444**

TABLE 4.7
TESTING ACCURACY COMPARISON ON BUPA LIVER DISORDERS DATASET WITH RBF KERNEL MAPPING

Trial	Testing accuracy of RBF SVM	Splitting hyperplane of RBF GSVM	Testing accuracy of RBF GSVM
1	**0.6690**	pc5= -0.0353	**0.6828**

4.2.3 Data Preprocessing

Firstly, we scale and normalize the input features to [-0.9, 0.9]. Out results show that the scaling does not deteriorate the performance of classifiers but speed up the training and testing process significantly.

Secondly, we split each dataset into training dataset and testing dataset. BUPA Liver Disorder dataset is already split into training data and testing data, so we make just one trial on it. For

other two datasets, we randomly split the data into training data and testing data with the conditions in Eq. 4.4-4.6. We make five trials on each of the two datasets.

$$S(training) : S(testing) = 2:1, \tag{4.4}$$

$$S(positive\_training) : S(positive\_testing) = 2:1, \tag{4.5}$$

$$S(negative\_training) : S(negative\_testing) = 2:1, \tag{4.6}$$

where $S(x)$ means the size of the dataset x.

4.2.4 Modeling

Two models are created for performance comparison. The first one is a general SVM in the whole space. For linear SVM, regulation parameter $C \equiv 1$; for RBF SVM, kernel parameter $\gamma \equiv 1$ and regulation parameter $C \equiv 1$.

The second model is GSVM-CMW. Here we only split the whole space to two information granules. The splitting hyperplane has the format $x^k = c$, where x^k is the 4th feature. It means we only search the hyperplanes orthogonal to a single feature. And then two SVMs with the same parameters as the general SVM above are built for both information granules.

4.2.5 Result

For Wisconsin Breast Cancer dataset, Table 4.2 shows that the performance of linear GSVM-CMW (97.72%) is better than linear SVM (96.40%) averaged on 5 trials. And Table 4.5 shows that the performance of RBF GSVM-CMW (96.57%) is a little worse than RBF SVM (97.02%). The best model is linear GSVM-CMW (97.72%).

For Cleveland Heart-disease dataset, Table 4.3 shows that the performance of linear GSVM-CMW (84.04%) is a little better than linear SVM (83.84%) averaged on 5 trials. And Table 4.6 shows that the performance of RBF GSVM-CMW (83.42%) is significantly better than RBF SVM (79.17%). Once again, the best model is linear GSVM-CMW (84.04%).

For BUPA Liver Disorders dataset, Table 4.4 shows that the performance of linear GSVM-CMW (49.66%) is a little better than linear SVM (48.97%). And Table 4.7 shows that the performance of RBF GSVM-CMW (68.28%) is better than RBF SVM (66.90%). The best model is RBF GSVM-CMW (68.28%).

The experiment results demonstrate, although the splitting hyperplane is limited to be orthogonal to a single feature and the number of information granules is fixed to be two, that our GSVM shows superior generalization capability. Except BUPA Liver Disorder dataset, standard deviations of other two datasets show that experiment results are stable and conceivable.

However, Table 4.5 also shows that, sometimes GSVM performs a little worse than SVM. (More specifically, Table 4.5 shows that RBF GSVM for Wisconsin Breast Cancer dataset is a little worse than RBF SVM). One possible reason is that the modeling method proposed here is too simplified. If we can find more appropriate criteria to search splitting hyperplanes which are not necessary to be orthogonal to a single feature, the performance of GSVM is expected to be improved further. Another possible reason is that two information granules may be not the optimal ones, maybe recursively splitting the space and thus building more information granules could improve the performance of GSVM, maybe the whole feature space is itself the optimal one so it is even unnecessary to split it. It means the result on Table 4.5 is not contradictory to our general GSVM idea, because it shows that just one information granule is more suitable than two information granules in this special case. The open problem is how to get the optimal or suboptimal information granules effectively so that the utility of GSVM modeling is not deteriorated? The adaptive granulation may be a good direction to explore. In the future, we will make more study and experiments on these issues.

4.3 GSVM-CMW-PCA algorithm

Principle Component Analysis (PCA) is a classic technique to transform the d-dimensional original input space to another d-dimensional "principle components space", in which features (called "principle components") are linear combinations of original input features and are orthogonal to each other [44]. The advantage of PCA is that the features are ranked based on the data's projection variance on them. The larger the projection variance is, the more useful the feature is expected to be for discriminating the classes. That means PCA could serve as a feature selection technique to ease a high-dimensional data classification problem.

For granular computing, what is interested is the first principle component. Because the first principle component captures the largest projection variance in the data, it is reasonable to expect that a partition along its orthogonal direction could be helpful to quickly find suitable information granules.

A simple but fast modeling algorithm, named GSVM-CMW-PCA, is proposed for building a GSVM in the top-down way. The hyperplane used to split the feature space is decided by statistical principle component analysis and margin maximization principle. The algorithm is similar to GSVM-CMW. The only difference is that we equally select m constants $c_1, c_2, \ldots, c_m \in (pc1\_min, pc1\_max)$ for the first principle component $pc1$, where $pc1\_min$ is the minimum value of $pc1$ in the training data, while $pc1\_max$ the maximum value. As a result, there are altogether m hyperplane candidates. The hyperplane with the largest *CMW* value will be selected as the splitting hyperplane. If there are more than one hyperplanes with the largest *CMW* value, the hyperplane with the largest training accuracy is selected because we can expect the better accuracy could be generalized to new unseen dataset.

4.4 GSVM-CMW-PCA simulation

4.4.1 Data Preprocessing

Two public medical datasets from UCI data mining repository [68] listed in Table 4.1 are used for comparison. One is Wisconsin Breast Cancer dataset; another is Cleveland heart-disease dataset. Firstly, we scale and normalize the input features to [-0.9, 0.9]. Our preliminary results show that the scaling does not deteriorate the performance of classifiers but speed up the training and testing process much more.

Secondly, we make PCA for the two datasets, here we will remove all d input features and keep all d principle components. That means the data is transferred from original d-dimensional input feature space to d-dimensional principle component space.

Thirdly, we randomly split the data into training data and testing data with the conditions in Equations 4.4-4.6. We make 21 trials for each dataset.

4.4.2 Modeling

Two models are created for performance comparison. The first one is building a general SVM with RBF kernel in the whole space (called RBF-SVM). The RBF kernel parameters (γ, C) are optimized by grid search heuristic in the solution space defined in Equations 4.7-4.8.

$$\gamma \in \{2^{-16}, 2^{-14}, 2^{-12}, 2^{-10}, 2^{-8}, 2^{-6}, 2^{-4}, 2^{-2}, 2^{0}, 2^{2}, 2^{4}\} \tag{4.7}$$

$$C \in \{2^{-6}, 2^{-4}, 2^{-2}, 2^{0}, 2^{2}, 2^{4}, 2^{6}, 2^{8}, 2^{10}\} \tag{4.8}$$

The best parameter pair is selected by leave-one-out cross-validation.

The second model is our linear GSVM-CMW-PCA. Here we only split the whole space to two information granules. The splitting hyperplane has the format $pc1 = c$, where $pc1$ is the first principle component. It means we only search the hyperplanes orthogonal to the first principle

component. And then one linear SVM with regulation parameter $C \equiv 1$ is built for each information granule.

4.4.3 Result

The first two columns in Table 4.8 and Table 4.9 show that linear GSVM-CMW-PCA is competitive to RBF-SVM in terms of testing accuracy. For the Wisconsin Breast Cancer dataset, the performance of linear GSVM is even better than RBF-SVM. More interestingly, as showed in Table 4.10 and Table 4.11, the modeling time (training time plus testing time) of linear GSVM is significantly shorter than the modeling time of RBF-SVM. The reason is that RBF-SVM need leave-one-out cross-validation but linear GSVM's parameter is decided by maximal *CMW* calculated in Eq. 4.1. As Table 4.1 shows, the two datasets used for experiments are small-sized (only several hundreds samples with about ten features). Thus we can expect the efficiency difference will be more significant for a real-world classification problem with larger size.

As we know, one disadvantage of SVM with some kernel function is that it is sensitive to the parameters. For example, if we use RBF kernel to model a SVM, slimly different values for parameters (γ, C) will result in significantly different testing accuracy, which is showed in the first column of Table 4.12 and Table 4.13.

For comparison, we also tried to use grid search plus leave-one-out to optimize linear GSVM. The regulation parameter *C* is selected from Eq. 4.9.

$$C \in \{2^{-7}, 2^{-6}, 2^{-5}, 2^{-4}, 2^{-3}, 2^{-2}, 2^{-1}, 2^{0}, 2^{1}, 2^{2}, 2^{3}, 2^{4}, 2^{5}, 2^{6}\} \tag{4.9}$$

By comparing the results in the third columns of Table 4.8 and Table 4.9 to the second columns, we can know that linear GSVM-CMW-PCA performs almost the same with or without grid search, that means linear GSVM-CMW-PCA is stable to the regulation parameter *C*. It implies that we can replace time-consuming grid search or other search algorithms which are targeted on

maximizing leave-one-out (or more general, k-folds cross-validation) validation accuracy with fast PCA-based information granulation method by extending margin maximization principle.

TABLE 4.8
TESTING ACCURACY COMPARISON ON WISCONSIN BREAST CANCER DATASET

Trial	RBF-SVM	linear GSVM	linear GSVM after grid search
0	0.9649	0.9693	0.9693
1	0.9649	0.9781	0.9781
2	0.9430	0.9649	0.9649
3	0.9605	0.9605	0.9605
4	0.9561	0.9605	0.9605
5	0.9781	0.9781	0.9781
6	0.9868	0.9825	0.9825
7	0.9605	0.9737	0.9737
8	0.9474	0.9781	0.9781
9	0.9649	0.9737	0.9737
10	0.9737	0.9825	0.9825
11	0.9693	0.9825	0.9825
12	0.9737	0.9781	0.9781
13	0.9561	0.9693	0.9693
14	0.9561	0.9649	0.9649
15	0.9781	0.9825	0.9825
16	0.9605	0.9649	0.9649
17	0.9781	0.9825	0.9825
18	0.9825	0.9781	0.9781
19	0.9605	0.9649	0.9649
20	0.9559	0.9780	0.9692
average	0.9653	0.9737	0.9733

TABLE 4.9
TESTING ACCURACY COMPARISON ON CLEVELAND HEART-DISEASE DATASET

Trial	RBF-SVM	linear GSVM	linear GSVM after grid search
0	0.7700	0.7700	0.7800
1	0.8900	0.8400	0.8700
2	0.8800	0.8900	0.8900
3	0.8000	0.8100	0.8000
4	0.8300	0.7800	0.7900
5	0.7800	0.8400	0.7900
6	0.8800	0.8800	0.8700
7	0.8700	0.8600	0.8200
8	0.8300	0.8300	0.8300
9	0.8500	0.8300	0.8400
10	0.8200	0.8400	0.8500
11	0.8200	0.8200	0.8200
12	0.8400	0.8300	0.8400
13	0.8400	0.8500	0.8300
14	0.7900	0.8200	0.8100
15	0.8500	0.8400	0.8500
16	0.8300	0.8200	0.8200
17	0.8800	0.9000	0.9100
18	0.8000	0.8000	0.8000
19	0.8700	0.8500	0.8600
20	0.8571	0.8776	0.8673
average	**0.8370**	**0.8370**	**0.8351**

TABLE 4.10
MODELING TIME COMPARISON ON WISCONSIN BREAST CANCER DATASET

Trial	RBF-SVM (seconds)	linear GSVM (seconds)
0	2387	≈1
1	2343	≈1
2	2331	≈1
3	2399	≈1
4	2324	≈1
5	2407	≈1
6	2432	≈1
7	2410	≈1
8	2350	≈1
9	2387	≈1
10	2402	≈1
11	2431	≈1
12	2387	≈1
13	2326	≈1
14	2351	≈1
15	2418	≈1
16	2370	≈1
17	2416	≈1
18	2469	≈1
19	2381	≈1
20	2410	≈1
average	**2387.2**	**≈1**

TABLE 4.11
MODELING TIME COMPARISON ON CLEVELAND HEART-DISEASE DATASET

Trial	RBF-SVM (seconds)	linear GSVM (seconds)
0	495	≈1
1	530	≈1
2	530	≈1
3	488	≈1
4	490	≈1
5	447	≈1
6	466	≈1
7	455	≈1
8	440	≈1
9	452	≈1
10	436	≈1
11	431	≈1
12	443	≈1
13	442	≈1
14	435	≈1
15	451	≈1
16	447	≈1
17	464	≈1
18	442	≈1
19	453	≈1
20	461	≈1
average	**461.8**	**≈1**

TABLE 4.12
STANDARD DEVIATION OF TESTING ACCURACY ON WISCONSIN BREAST CANCER DATASET WITH DIFFERENT MODEL PARAMETERS

TRIAL	RBF-SVM WITH DIFFERENT GAMMA AND C FROM EQ. 11-12	LINEAR GSVM WITH DIFFERENT C FROM EQ. 13
0	0.1324	0.0018
1	0.1340	0.0072
2	0.1267	0.0110
3	0.1314	0.0040
4	0.1274	0.0054
5	0.1380	0.0018
6	0.1387	0.0000
7	0.1337	0.0061
8	0.1278	0.0126
9	0.1329	0.0018
10	0.1374	0.0018
11	0.1341	0.0011
12	0.1337	0.0049
13	0.1284	0.0037
14	0.1302	0.0000
15	0.1374	0.0015
16	0.1307	0.0000
17	0.1357	0.0037
18	0.1365	0.0049
19	0.1325	0.0000
20	0.1317	0.0155
average	**0.1329**	**0.0042**

TABLE 4.13
STANDARD DEVIATION OF TESTING ACCURACY ON CLEVELAND HEART-DISEASE DATASET WITH DIFFERENT MODEL PARAMETERS

Trial	RBF-SVM with different Gamma and C from Eq. 11-12	linear GSVM with different C from Eq. 13
0	0.1138	0.0062
1	0.1414	0.0183
2	0.1576	0.0247
3	0.1254	0.0074
4	0.1279	0.0079
5	0.1195	0.0302
6	0.1586	0.0116
7	0.1336	0.0107
8	0.1260	0.0058
9	0.1414	0.0156
10	0.1294	0.0237
11	0.1308	0.0051
12	0.1406	0.0149
13	0.1341	0.0141
14	0.1268	0.0108
15	0.1474	0.0151
16	0.1258	0.0072
17	0.1636	0.0243
18	0.1200	0.0072
19	0.1451	0.0131
20	0.1442	0.0102
average	**0.1359**	**0.0135**

TABLE 4.14
RELATIONSHIP BETWEEN VALIDATION ACCURACY AND TESTING ACCURACY OF RBF-SVM ON WISCONSIN BREAST CANCER DATASET

Trial	HIGHEST VALIDATION ACCURACY	CORRESPONDING TESTING ACCURACY	HIGHEST TESTING ACCURACY
0	0.9758	0.9649	0.9737
1	0.9714	0.9649	0.9825
2	0.9802	0.9430	0.9605
3	0.9758	0.9605	0.9737
4	0.9802	0.9561	0.9605
5	0.9692	0.9781	0.9825
6	0.9626	0.9868	0.9912
7	0.9758	0.9605	0.9825
8	0.9780	0.9474	0.9605
9	0.9714	0.9649	0.9737
10	0.9670	0.9737	0.9825
11	0.9670	0.9693	0.9781
12	0.9692	0.9737	0.9781
13	0.9758	0.9561	0.9649
14	0.9780	0.9561	0.9605
15	0.9648	0.9781	0.9868
16	0.9780	0.9605	0.9649
17	0.9670	0.9781	0.9825
18	0.9670	0.9825	0.9825
19	0.9736	0.9605	0.9737
20	0.9737	0.9559	0.9780
average	**0.9725**	**0.9653**	**0.9749**

TABLE 4.15
RELATIONSHIP BETWEEN VALIDATION ACCURACY AND TESTING ACCURACY OF RBF-SVM ON CLEVELAND HEART-DISEASE DATASET

Trial	HIGHEST VALIDATION ACCURACY	CORRESPONDING TESTING ACCURACY	HIGHEST TESTING ACCURACY
0	0.8782	0.7700	0.8200
1	0.8376	0.8900	0.9100
2	0.8223	0.8800	0.9000
3	0.8579	0.8000	0.8300
4	0.8579	0.8300	0.8600
5	0.8376	0.7800	0.8400
6	0.8173	0.8800	0.8900
7	0.8376	0.8700	0.8700
8	0.8477	0.8300	0.8400
9	0.8325	0.8500	0.8600
10	0.8528	0.8200	0.8600
11	0.8528	0.8200	0.8500
12	0.8376	0.8400	0.8600
13	0.8426	0.8400	0.8500
14	0.8477	0.7900	0.8200
15	0.8426	0.8500	0.8700
16	0.8579	0.8300	0.8300
17	0.8173	0.8800	0.9200
18	0.8629	0.8000	0.8200
19	0.8173	0.8700	0.8800
20	0.8241	0.8571	0.8878
average	**0.8420**	**0.8370**	**0.8604**

As showed in Table 4.14 and Table 4.15, for RBF-SVM optimized by grid search based on leave-one-out cross-validation, the testing accuracy is not the highest when the validation accuracy is the highest. The phenomenon means there are some noises in the data. As a result,

the assumption of cross-validation, the data comes from the same implicit statistical distribution, does not hold any more. However, linear GSVM-CMW-PCA modeling algorithm extends maximal margin principle so that it does not need cross-validation, and thus could overcome noise problem. That means linear GSVM-CMW-PCA is robust to the noise.

4.5 Discussion

In this chapter, we present GSVM-CMW, an implementation method for modeling a GSVM by building information granules in the top-down way. The hyperplane used to split the feature space is selected by extending statistical margin maximization principle. The simulation results on three medical binary classification problems show that finding the splitting hyperplane is not a trivial task and GSVM does show some improvement on testing accuracy compared to building one single SVM in the whole feature space. More importantly and more interestingly, GSVM provides a new mechanism to address complex classification problems, which are common in medical or biological information processing applications. The modeling method for a GSVM is just the first step into this interesting research topic. In the future, we will try to find more appropriate modeling methods for GSVM and compare it to kernel-based SVM such as RBF-SVM.

We also explore to utilize PCA technology in this chapter for fast modeling a GSVM by building information granules in the top-down way. The hyperplane used to split the feature space is decided by applying extended statistical margin maximization principle on the first principle component. In this way, a GSVM could be modeled much faster while still remaining high accuracy. The experimental results on two medical datasets show that finding the splitting hyperplane is not a trivial task and linear GSVM is competitive to the well-known RBF kernel with optimal parameters in terms of testing accuracy, but linear GSVM could be modeled in

much shorter time and performs more stable (non-sensitive to parameters) and more robust (anti-noise due to extended margin maximization principle). More importantly and more interestingly, GSVM provides a new mechanism, which is competitive to kernel mapping method, to address complex classification problems with high accuracy and high speed.

CHAPTER 5

GSVM-AR

5.1 Association rules

Many previous works have reported that the frequent patterns occurred in the training dataset of a complex and huge classification problem could lead to measured improvement on testing accuracy [90,111,45]. The idea was named "association classification" [111].

For a binary classification problem with continuous features, an association rule is usually formed as:

$$if\ a_1 \in [v_{11}, v_{12}]\ and\ \ a_2 \in [v_{21}, v_{22}]\ and \ldots a_n \in [v_{n1}, v_{n2}],\ then\ \ \mathrm{y} = 1 (\mathrm{or}\ -1) \quad (5.1)$$

The support and confidence of an association rule for a binary classification problem are defined in Equations 5.2-5.3:

$$SUP(AR) = S_{PG} / S_W \quad (5.2)$$

$$COF(AR) = S_{PG} / S_G \quad (5.3)$$

where S_W is the size of training data with the same class label as the *THEN*-part of the association rule, S_G is the size of training data that satisfy the *IF*-part, while S_{PG} is the size of training data correctly classified by the association rule. Notice that S_W is defined in such a way that the support and confidence of an association rule are calculated based on a single class. As a result, the association rule mining will not be biased for major class in an unbalanced binary classification problem.

From Eq. 5.1, an association rule (or a set of association rules combined disjunctively) could be used to partition the feature space to find an information granule. So association rules mining is a

possible solution for granulation. The realization of a successful "association granulation" depends on the following two issues:

An association rule with high enough confidence could deduce a "pure" granule, in which it is unnecessary to build a classifier because of its high purity. If its support is also high, it could significantly simplify and speed up classification because it decreases the size of the training dataset.

A more general association rule with a shorter *IF*-part should be more possible to avoid overfitting training dataset. A short *IF*-part means a low model complication, which in turn means a good generalization possibility.

5.2 Algorithm

This chapter proposes to take advantage of association rules mining for modeling a GSVM in the top-down way. The hyperplane used to split the feature space is selected according to mined association rules with high confidence and significant support. Confidence of a good association rule should be as high as possible (should be at least higher than the validation accuracy of the best SVM in the whole space), while its support can not be too small, otherwise it is not useful (in other words, the support should be significant). Fig. 5.1 describes the GSVM-AR modeling algorithm. The basic idea is to extend “Positive Pure Granule” (PPG) and “Negative Pure Granule” (NPG) iteratively until GSVM-AR gets the best validation performance. If necessary, the cross-validation method could be used. Notice the support threshold is provided as an input, and the confidence threshold is set to be the validation accuracy of the general SVM in the whole feature space. For each feature, at most two association rules are mined. Therefore, if the time complexity for modeling a general SVM is $O(l^2 d)$, the time complexity for modeling a GSVM

is $O(l^2d^2)$ and dominated by the while loop to find a GSVM with the best validation performance. Notice the time complexity of MiningOneFeatureARs is $O(ld)$.

5.3 Simulation

5.3.1 Data description

Protein homology prediction between protein sequences is one of critical problems in computational biology. Protein sequences are very difficult to understand and model due to their complex random length nature. The sequential similarity measurement is believed to be useful to predict the structural or functional similarity of proteins and thus it is helpful to group proteins with similar function together. Due to this reason, it is a hot research topic for computational biologists and computer scientists in recent years. Various algorithms have been developed to measure the sequential similarity between two proteins [74,88]. From the viewpoint of data mining, protein homology prediction could be viewed as a predictive data mining task [44] because the goal is to predict the unknown value of a variable of interest given known values of other variables. More specifically, it could be modeled as a binary classification problem. If a protein sequence is homologous to a pre-specified protein sequence, it is classified to be a positive case and 1 is output, otherwise it is negative and -1 is output.

KDDCUP04 protein homology prediction task at http://kodiak.cs.cornell.edu/kddcup/index.html is used for experiment. The detailed characteristics of the dataset are listed in Table 5.1. From the table, we can see that the task could be modeled as a binary classification or a binary ranking problem: Given a protein sequence, the task is to predict whether it is homologous to the corresponding native sequence or not. There are 153 native sequences in the training dataset and 150 native sequences in the testing dataset. For each native sequence, there is a block of approximately 1000 protein sequences with class label (1 means homologous and 0 means non-

homologous). The class labels of protein sequences in testing dataset are unknown. 74 features are provided to describe the match (e.g. the score of a sequence alignment) between the native protein sequence and the sequence that is tested for homology. We can also see that the problem is highly unbalanced: there are only 1296 homologous protein sequences from altogether 145751 ones in the training dataset.

Four metrics are used for performance measures:

- TOP1: fraction of blocks with a homologous sequence ranked top 1 (maximize)
- RKL: average rank of the lowest ranked homologous sequence (minimize)
- RMS: root mean squared error averaged on blocks (minimize)
- APR: average of the average precision in each block. For a single block, APR could be approximately described as the area of precision-recall curves. (maximize)

RMS is a metric for accuracy evaluation, but is easier to show the differences between models than directly using error values. The other 3 metrics are rank-based, which means that the 3 metrics' values are decided by the order of ranking list, and the absolute values of predictions do not affect the performances. The four metrics are precisely defined in perf [20]. In our experiment, we use the corresponding code to calculate the four metrics.

TABLE 5.1
CHARACTERISTICS OF KDDCUP04 PROTEIN HOMOLOGY PREDICTION DATASETS

Dataset	Block	Size	Attr	Class	Ratio
Training	153	≈1000	74	2	1296 / 144455
Testing	150	≈1000	74	2	N/A

Note 1: Block = # of blocks, Size = # of protein sequences in each block, Attr = # of input features, Class = # of classes, Ratio = # of homologous sequences / # of non-homologous sequences.
Note 2: The data is without missing data.

```
MiningOneFeatureARs(TrainingData T, SupportThreshold Sth, ConfidenceThreshold Cth)
{
   y = the class label vector in T;
   WAR = empty set;

   for each input feature x
   {
      PR = {r | r is an association rule with the format "if x0<x<x1,then y=1", support>=Sth, and confidence >=Cth in T};
      R = {r | r is the rule with the highest confidence in PR};
      WAR = WAR + {r | r is the rule with the highest support in R};
      NR = {r | r is an association rule with the format "if x0<x<x1,then y=-1", support>=Sth, and confidence >=Cth in T};
      R = {r | r is the rule with the highest confidence in NR};
      WAR = WAR + {r | r is the rule with the highest support in R};
   }
    return WAR;
}

GSVM-AR(TrainingData T, SupportThreshold Sth)
{
   MG = WFS = the whole feature space on T;
   PPG = NPG = empty set;
   MG_SVM = the SVM modeled on the training data in MG optimized by grid search heuristic;
   GSVM-AR = {       if a sample x in PPG, then its class label y = 1;
                     if a sample x in NPG, then its class label y = -1;
                     if a sample x in MG,  then its class label y=the class label predicted by MG_SVM;
                }
  VP = the cross validation performance of GSVM-AR in WFS;
   Cth = the cross validation accuracy of GSVM-AR in WFS;

   WAR = MiningOneFeatureARs(T,Sth,Cth);
   while(WAR is not empty)
   {
            r = the association rule in WAR such that if r is added into PPG or NPG,
                the purity of PPG or NPG is the highest compared to adding any other rule in WAR;
            WAR = WAR - {r};

            if r is a positive rule
            {
                     newPPG = PPG + {r};
                     newNPG = NPG;
            }
            else
            {
                     newPPG = PPG;
                     newNPG = NPG + {r};
            }
            newMG = WFS - newPPG - newNPG;
            newMG_SVM = the SVM modeled on the training data in newMG optimized by grid search heuristic;
            newGSVM-AR = {    if a sample x in newPPG, then its class label y = 1;
                              if a sample x in newNPG, then its class label y = -1;
                              if a sample x in newMG,  then its class label y=the classlabel predicted by newMG_SVM;
                         }
            newVP = the cross validation performance of newGSVM-AR in WFS;
            if newVP is better than VP
            {
                     PPG = newPPG;
                     NPG = newNPG;
                     MG = newMG;
                     GSVM-AR = newGSVM-AR;
                     VP = newVP;
            }
   }
   return GSVM-AR;
}
```

Figure. 5.1. GSVM-AR modeling algorithm

Because of the absent of the class label in the testing dataset, only the training dataset is used in our experiment. And because our result is not on the original testing dataset, so it could not be compared with the current best results on the competition. That is, our goal is not to be involved in the competition to get the best result, but to use the data to show GSVM-AR's superiority to SVM.

5.3.2 Data preprocessing

Firstly, we scale and normalize the input features to [-0.9, 0.9]. The scaling is on each different block separately. The reason is that the protein sequences in different blocks are in different protein families, which are so remote that the similar absolute feature vectors can not mean similar homology behaviors. However, to avoid overfitting, the association rules are mined from non-scaled original data.

After scaling, we make five trials. In each trial, the data is randomly split into training dataset and testing dataset with the conditions in Equations 5.4-5.6. That is, 102 blocks are used for training and other 51 blocks used for testing.

$$S(training) : S(testing) = 2 : 1 \tag{5.4}$$

$$S(positive\_training) : S(positive\_testing) = 2 : 1 \tag{5.5}$$

$$S(negative\_training) : S(negative\_testing) = 2 : 1 \tag{5.6}$$

$S(x)$ means the number of blocks in the dataset *x*.

5.3.3 Modeling

In each trial, we select just 1 block for modeling and other 101 training blocks for validation. That is because our preliminary tests also show that it is even worse if we mix multiple blocks together for training a model. (We also skip the details because it is out of the scope of this chapter).

Two models are created for performance comparison. The first one is a general SVM in the whole space. The parameters of the SVM are optimized by grid search heuristic [46]:

In linear SVM, the regulation parameter C is optimized by grid search heuristic at Eq. 5.7.

$$C \in \{2^{-10},2^{-9.5},2^{-9},2^{-8.5},2^{-8},2^{-7.5},2^{-7},2^{-6.5},\\ 2^{-6},2^{-5.5},2^{-5},2^{-4.5},2^{-4},2^{-3.5},2^{-3},2^{-2.5},\\ 2^{-2},2^{-1.5},2^{-1},2^{-0.5},2^{0},2^{0.5},2^{1},2^{1.5},2^{2}\} \tag{5.7}$$

The RBF kernel parameters (γ, C) are optimized by grid search heuristic at Equations 5.8-5.9.

$$\gamma \in \{2^{-26},2^{-24},2^{-22},2^{-20},2^{-18},2^{-16},2^{-14},2^{-12},2^{-10},2^{-8}\} \tag{5.8}$$

$$C \in \{2^{5},2^{6},2^{7},2^{8},2^{9},2^{10},2^{11},2^{12},2^{13},2^{14},2^{15},2^{16}\} \tag{5.9}$$

We repeat this modeling process for each of 102 training blocks. After that, 5 blocks with best validation performance on a special metric are selected to build GSVM for comparison.

For GSVM-AR modeling, we mine association rules first. To avoid overfitting, the association rules should be as simple as possible. Due to this reason, only 1-feature association rules with the format $x0<=x<x1$ is mined. And only the rules with confidence higher than the general SVM's validation accuracy and significant support are kept as candidates.

TABLE 5.2
1-FEATURE ASSOCIATION RULES ON ORIGINAL TRAINING DATA WITH CONFIDENCE/SUPPORT IN 5 TRIALS

Rule	Trial 1	Trial 2	Trial 3	Trial 4	Trial 5
If attr5>78, then y=-1	100%/8666	100%/8976	100%/9141	100%/8992	100%/8753
If attr45>-4, then y=-1	100%/11642	100%/11703	100%/11075	100%/11234	100%/11786
If attr53<-2.06, then y=-1	100%/1080	100%/1041	100%/1060	100%/987	100%/1029
If attr58<-1.53, then y=-1	100%/1714	100%/1727	100%/1758	100%/1712	100%/1711
If attr68<-2.21, then y=-1	100%/2515	100%/2608	100%/2610	100%/2404	100%/2725
If attr58>8.3, then y=1	99.27%/547	99.56%/457	99.35%/465	99.42%/517	99.41%/510

Note:

Trial 1 (positive: negative = 879: 96055 in the training dataset).
Trial 2 (positive: negative = 909: 95911 in the training dataset).
Trial 3 (positive: negative = 810: 96139 in the training dataset).
Trial 4 (positive: negative = 886: 95580 in the training dataset).
Trial 5 (positive: negative = 919: 96892 in the training dataset).

The mined association rules with their support and confidence are listed in Table 5.2. In the table, the support and confidence for each rule are listed. For example, the sixth row shows a positive association rule. In trial 1, the training dataset has 879 homologous protein sequences, 547 ones of which satisfy *IF*-part of the rule, and 543 ones satisfy both *IF*-part and *THEN*-part:

If attr58>8.3, then y=1 with Confidence=543/547= 99.27%,

Support= 543/879= 61.77%.

The second row shows a negative association rule. In trial 1, the training dataset has 11642 protein sequences satisfy *IF*-part of the rule. And all of them satisfy *THEN*-part too:

If attr45>-4, then y=-1 with Confidence=11642/11642=100%,

Support=11642/96055= 12.12%.

After that, we iteratively combine association rules by disjunction to find the granules that are both pure and significant. When the process is completed, 3 granules are created: the granule induced by negative rules is named *NPG* because almost all protein sequences in the granule are non-homologous; the granule induced by positive rules is named *PPG* due to the similar reason; and the remaining space is named "Mixed Granule" (*MG*), in which a SVM with the same kernel as the general SVM is built. The 3 granules are decided by Equations 5.10-5.12:

$$PPG = \bigcup positive \quad association \quad rules, \tag{5.10}$$

$$NPG = \bigcup negative \quad association \quad rules - PPG, \tag{5.11}$$

$$MG = WFS - PPG - NPG, \tag{5.12}$$

where *WFS* means the whole feature space.

Notice that the overlapping area of *PPG* and *NPG* is accounted in *PPG*. That means the granulation is biased for homologous proteins to compensate for its minority.

For the protein prediction task,

PPG is formed by

If attr58 > 8.3, then y = 1

NPG is formed by

If attr58 <= 8.3 and

(attr5 > 78 or attr45 > -4 or attr53 < -2.06 or attr58 < -1.53 or attr68 < -2.21),

Then y = -1

And then we compare two models on the top 5 blocks for the 4 metrics. For protein sequences in the *PPG* and *NPG*, the outputs are 1s or -1s, respectively. Because SVM is originally designed for binary classification problems, for protein sequences in *MG*, if a metric is rank-based, we adopt the distance from the predicted protein sequence to the separating hyperplane (normalized to be in [-1, 1]) as its output.

5.3.4 Result

The experimental results are reported in Tables 5.3-5.6 and Figures 5.2-5.5. In each cell of a table, the performance of SVM is reported as the first number, while the performance of GSVM-AR as the second number.

For TOP1 metric, Table 5.3 and Fig. 5.2 show that the performance of GSVM-AR is a little better than SVM with both linear kernel (from 85.33% to 85.49% for testing data) and RBF kernel (from 85.10% to 85.41% for testing data). For example, for testing data with linear kernel, averagely to say, there are 51 * 85.33% = 43.52 blocks with a homologous protein sequence as the TOP1 in the ranking list predicted by SVM, while 51 * 85.49% = 43.60 blocks by GSVM-AR. The improvement is small because the protein sequences ranked as TOP1 in the lists are easiest to be predicted. So a general SVM is good enough to predict them.

For RKL metric, Table 5.4 and Fig. 5.3 show that GSVM-AR significantly outperform SVM. That is, the average rank of the lowest ranked homologous sequences is decreased significantly (from 78.02 to 71.01 for testing data with linear kernel, from 75.80 to 69.25 for testing data with RBF kernel,). When recall is set to be 1, GSVM-AR has higher precision than SVM. As a result, homologous sequences are clearer to be differentiated from non-homologous ones with GSVM-AR than with SVM.

For RMS metric, Table 5.5 and Fig. 5.4 show that the performance of GSVM-AR is also significantly better than SVM. That is, the average root mean squared error is decreased significantly (from 0.0554 to 0.0441 for testing data with linear kernel, from 0.0554 to 0.0440 for testing data with RBF kernel). That means GSVM-AR is more accurate. For example, approximately, for a testing block with 1000 protein sequences, 3.07 protein sequences are misclassified by SVM with RBF kernel, while only 1.94 ones are misclassified by GSVM-AR with RBF kernel.

TABLE 5.3
TOP1 ON VALIDATION/TEST SET IN 5 TRIALS
(MEAN ± STANDARD DEVIATION FROM BEST 5 BLOCKS)

Trial	Validation data with Linear kernel (%)	Testing data with Linear kernel (%)	Validation data with RBF kernel (%)	Testing data with RBF kernel (%)
1	91.09±0.00/91.09±0.00	78.82±2.56/79.61±2.23	91.09±0.70/91.09±0.70	78.82±2.56/79.61±2.23
2	85.55±0.54/85.55±0.54	90.98±1.07/90.98±1.07	85.55±0.54/85.55±0.54	90.59±1.64/90.59±1.64
3	86.14±0.00/86.14±0.00	89.42±1.75/89.42±1.75	86.14±0.70/86.14±0.70	89.02±1.07/89.02±1.07
4	87.72±0.54/87.72±0.54	85.88±3.22/85.88±3.22	87.53±0.54/87.53±0.54	85.49±4.07/85.88±3.22
5	90.50±0.54/90.69±0.54	81.57±1.07/81.57±1.07	90.30±0.44/90.50±0.54	81.57±1.07/81.96±0.88

Note:

Best 5 blocks in trial 1 are 210, 103, 73, 69, and 16.
Best 5 blocks in trial 2 are 170, 65, 274, 255, and 236.
Best 5 blocks in trial 3 are 210, 162, 144, 65, and 64.
Best 5 blocks in trial 4 are 170, 65, 16, 289, and 274.
Best 5 blocks in trial 5 are 73, 16, 261, 255, and 252.

TABLE 5.4
RKL ON VALIDATION/TEST SET IN 5 TRIALS
(MEAN ± STANDARD DEVIATION FROM BEST 5 BLOCKS)

Trial	Validation data with Linear kernel	Testing data with Linear kernel	Validation data with RBF kernel	Testing data with RBF kernel
1	67.69±2.38/62.63±1.48	93.62±17.23/85.00±14.44	66.88±2.48/61.79±1.61	92.21±16.92/83.67±13.93
2	91.33±6.01/83.73±4.45	41.29±8.46/35.94±7.35	87.60±5.97/81.16±5.08	35.41±7.48/31.06±5.85
3	71.39±2.99/64.80±3.33	88.05±8.97/81.84±7.80	66.34±5.15/61.17±5.05	90.99±6.77/84.51±5.66
4	73.53±6.81/68.05±5.82	80.02±10.45/73.42±8.59	72.42±6.15/67.27±5.33	78.11±8.61/71.91±7.40
5	71.64±1.71/65.60±2.31	87.11±6.97/78.83±5.55	68.91±4.20/63.70±4.02	82.29±8.57/75.11±6.96

Note:

Best 5 blocks in trial 1 are 55, 164, 303, 135, and 266.
Best 5 blocks in trial 2 are 55, 110, 13, 69, and 73.
Best 5 blocks in trial 3 are 289, 110, 2, 164, and 69.
Best 5 blocks in trial 4 are 55, 289, 164, 25, and 65.
Best 5 blocks in trial 5 are 13, 110, 73, 164, and 16.

TABLE 5.5
RMS ON VALIDATION/TEST SET IN 5 TRIALS
(MEAN ± STANDARD DEVIATION FROM BEST 5 BLOCKS)

Trial	Validation data with Linear kernel (%)	Testing data with Linear kernel (%)	Validation data with RBF kernel (%)	Testing data with RBF kernel (%)
1	5.33±0.03/3.87±0.21	5.90±0.09/5.24±0.20	5.31±0.03/3.87±0.23	5.88±0.06/5.25±0.20
2	5.79±0.04/4.66±0.09	4.98±0.14/3.46±0.09	5.79±0.04/4.66±0.09	4.98±0.14/3.46±0.09
3	5.32±0.06/4.05±0.09	5.95±0.07/4.63±0.08	5.31±0.05/4.05±0.09	5.95±0.07/4.63±0.07
4	5.49±0.05/4.19±0.06	5.47±0.09/4.49±0.08	5.49±0.05/4.13±0.02	5.45±0.09/4.44±0.11
5	5.57±0.02/4.22±0.14	5.40±0.07/4.20±0.02	5.62±0.12/4.21±0.15	5.45±0.09/4.21±0.04

Note:

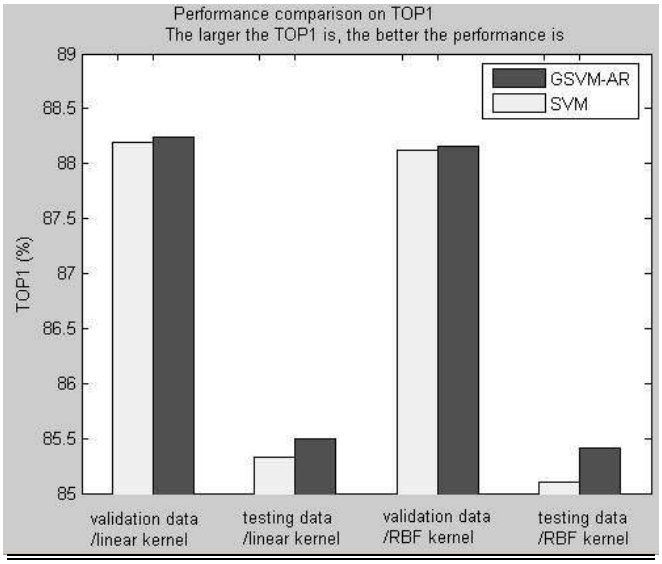
Best 5 blocks in trial 1 are 256, 103, 277, 271, and 212.
Best 5 blocks in trial 2 are 73, 48, 238, 256, and 277.
Best 5 blocks in trial 3 are 103, 212, 60, 7, and 48.
Best 5 blocks in trial 4 are 256, 231, 73, 277, and 103.
Best 5 blocks in trial 5 are 60, 16, 103, 7, and 48.

TABLE 5.6
APR ON VALIDATION/TEST SET IN 5 TRIALS
(MEAN ± STANDARD DEVIATION FROM BEST 5 BLOCKS)

Trial	Validation data with Linear kernel (%)	Testing data with Linear kernel (%)	Validation data with RBF kernel (%)	Testing data with RBF kernel (%)
1	83.30±0.47/83.53±0.57	75.78±1.23/76.38±0.86	83.28±0.52/83.49±0.60	75.94±1.37/76.51±0.96
2	77.61±0.24/77.77±0.21	86.19±1.22/86.47±1.20	77.52±0.26/77.77±0.15	86.06±1.12/86.34±1.13
3	78.39±0.86/78.68±0.87	82.92±0.84/83.01±0.85	78.36±0.88/78.60±0.89	83.08±1.08/83.14±1.09
4	81.45±0.36/81.66±0.45	79.39±1.36/79.72±1.41	81.41±0.45/81.63±0.54	79.08±1.78/79.38±1.85
5	83.39±0.44/83.63±0.45	76.03±0.55/76.28±0.56	83.31±0.65/83.56±0.68	76.16±0.53/76.42±0.59

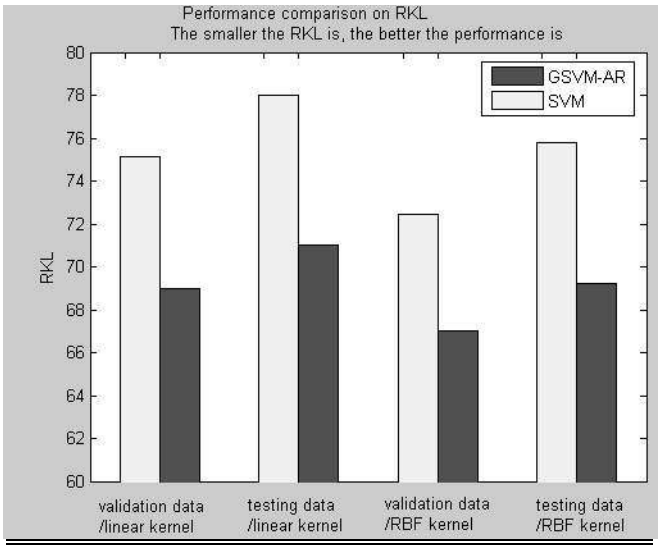
Note:

Best 5 blocks in trial 1 are 16, 27, 73, 255, and 103.
Best 5 blocks in trial 2 are 55, 73, 163, 256, and 170.
Best 5 blocks in trial 3 are 103, 64, 60, 274, and 243.
Best 5 blocks in trial 4 are 16, 27, 73, 170, and 274.
Best 5 blocks in trial 5 are 16, 73, 163, 103, and 170.



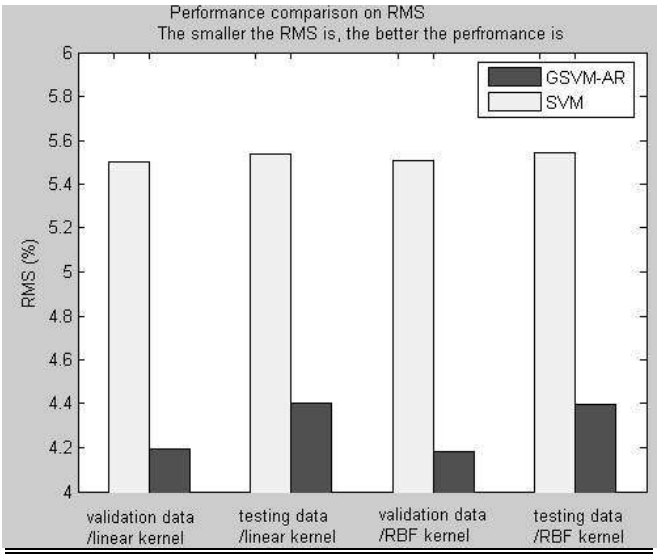
SVM/ GSVM-AR	Validation data with Linear kernel (%)	Testing data with Linear kernel (%)	Validation data with RBF kernel (%)	Testing data with RBF kernel (%)
Mean	**88.20/88.24**	**85.33/85.49**	**88.12/88.16**	**85.10/85.41**
Std	**2.32/2.36**	**5.07/4.84**	**2.32/2.37**	**5.00/4.60**

Figure . 5.2. Performance comparison on TOP1 metric averaged on 5 trials. The larger TOP1 is, the better the performance is. The results are grouped by different data/kernel pairs. In each group, the left bar shows the result of SVM, while the right GSVM-AR. The mean and standard deviation statistics are given in the above table. In each cell, the 1st number is the result of SVM, while the 2nd GSVM-AR.



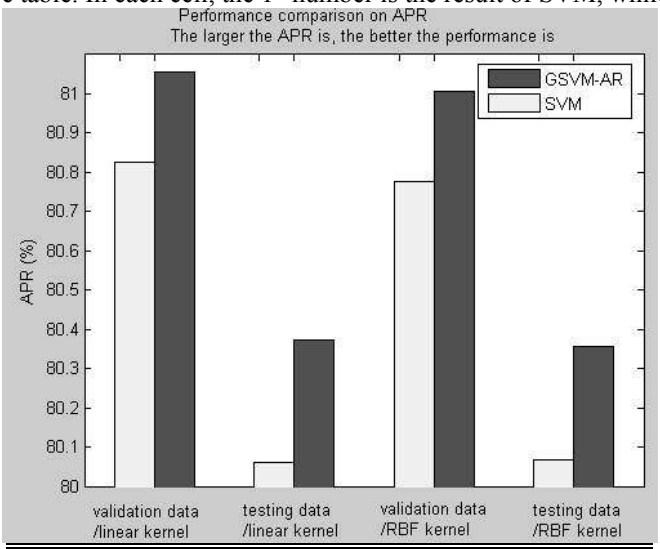
SVM/ GSVM-AR	Validation data with Linear kernel	Testing data with Linear kernel	Validation data with RBF kernel	Testing data with RBF kernel
Mean	**75.12/68.96**	**78.02/71.01**	**72.43/67.02**	**75.80/69.25**
Std	**9.42/8.48**	**21.72/20.17**	**9.24/8.56**	**23.31/21.56**

Figure. 5.3. Performance comparison on RKL metric averaged on 5 trials. The smaller RKL is, the better the performance is. The results are grouped by different data/kernel pairs. In each group, the left bar shows the result of SVM, while the right GSVM-AR. The mean and standard deviation statistics are given in the above table. In each cell, the 1st number is the result of SVM, while the 2nd GSVM-AR.



SVM/ GSVM-AR	Validation data with Linear kernel (%)	Testing data with Linear kernel (%)	Validation data with RBF kernel (%)	Testing data with RBF kernel (%)
Mean	**5.50/4.20**	**5.54/4.41**	**5.50/4.18**	**5.54/4.40**
Std	**0.18/0.29**	**0.37/0.60**	**0.19/0.30**	**0.37/0.60**

Figure. 5.4. Performance comparison on RMS metric averaged on 5 trials. The smaller RMS is, the better the performance is. The results are grouped by different data/kernel pairs. In each group, the left bar shows the result of SVM, while the right GSVM-AR. The mean and standard deviation statistics are given in the above table. In each cell, the 1st number is the result of SVM, while the 2nd GSVM-AR.



SVM/ GSVM-AR	Validation data with Linear kernel (%)	Testing data with Linear kernel (%)	Validation data with RBF kernel (%)	Testing data with RBF kernel (%)
Mean	**80.83/81.06**	**80.06/80.37**	**80.78/81.01**	**80.07/80.36**
Std	**2.52/2.53**	**4.22/4.12**	**2.54/2.54**	**4.20/4.09**

Figure. 5.5. Performance comparison on APR metric averaged on 5 trials. The larger APR is, the better the performance is. The results are grouped by different data/kernel pairs. In each group, the left bar shows the result of SVM, while the right GSVM-AR. The mean and standard deviation statistics are given in the above table. In each cell, the 1st number is the result of SVM, while the 2nd GSVM-AR.

For APR metric, Table 5.6 and Fig. 5.5 show that the performance of GSVM-AR is also better than SVM with both linear kernel (from 80.06% to 80.37%) and RBF kernel (from 80.07% to 80.36%).

The standard deviations in these tables show that experiment results are stable and conceivable.

5.4 Discussion

In this chapter, we propose the GSVM-AR algorithm for modeling a GSVM by building information granules in the top-down way with the aid of association rules. GSVM_AR works by building three information granules, called Positive Pure Granule, Negative Pure Granule, and Mixed Granule, respectively. Because of being generated from association rules with high confidence and significant support, the *PPG* and *NPG* have high purity. Therefore we only need to build a Support Vector Machine in *MG*.

The experimental results on KDDCUP04 protein homology prediction task show that finding the splitting hyperplane is not a trivial task (We should be careful to select the association rules to avoid overfitting) and GSVM-AR does show significant improvement compared to building one single SVM in the whole feature space. Although the association rules are limited to be 1-feature format (that means the splitting hyperplane is limited to be orthogonal to a single feature) and the number of information granules is fixed to be three, GSVM-AR shows superior generalization capability. Another advantage is that GSVM-AR is easy to be implemented.

CHAPTER 6

GSVM-RFES

6.1 Gene selection and cancer classification on Microarray expression data

6.1.1 Biological background

Every organism is composed of cell(s). In each cell, there is a nucleus, where the genetic material (DNA) is located. The coding segments of DNA, named "genes", contain the sequence information for specific proteins, which are macro-molecules that play the key roles on biochemical and biological function, regulation and development of the organism. As a matter of fact, all cells in the same organism have exactly the same genome. However, due to different tissue types, different development stages, and different environmental conditions, genes from cells in the same organism can be expressed in different combinations and/or different quantities during the transcription process from DNA to messenger RNA (mRNA) and the translation process from mRNA to proteins. These different gene expression patterns, including both the combination and quantity, thus account for the huge variety of states and types of cells in the same organism [89]. Different organisms have different genomes and different gene expression patterns.

Very recently, DNA microarray (including cDNA microarray and GeneChip) has been developed as a powerful technology for molecular genetics studies, which simultaneously measures the mRNA expression levels of thousands to tens of thousands genes. A typical microarray expression experiment monitors expression level of each gene multiple times under different conditions or in different tissue types (for example, healthy tissue versus cancerous tissue, one kind of cancerous tissue versus another cancerous tissue). By recording such huge

gene expression data sets, it opens the possibility to distinguish tissue types and to identify disease-related genes whose expression data are good diagnostic indicators [89,50,7,74,70,2,71]. From the viewpoint of data mining, it is a predictive data mining task [44] to distinguish different tissue types because the goal is to predict the unknown value of a variable (healthy or cancerous; if cancerous, which kind of cancer) of interest given known values of other variables (gene expression data). More specifically, it could be modeled as a classification problem. For example, one well-known problem by utilizing microarray gene expression data is to distinguish between two variants of leukemia, which are Acute Myeloid Leukemia (AML) and Acute Lymphoblastic Leukemia (ALL). The AML/ALL problem could be modeled as a binary classification problem: if a sample is ALL, it is classified to be a negative case and -1 is output, otherwise it is AML and 1 is output.

6.1.2 Challenges for bioinformatics scientists

A typical gene expression dataset is extremely sparse compared to a traditional classification dataset: the data usually comes with only dozens of tissues/samples but with thousands or even tens of thousands of genes/features. This extreme sparseness is believed to significantly deteriorate the performance of a classifier. As a result, the ability to extract a subset of informative genes while removing irrelevant or redundant genes is crucial for accurate classification. Furthermore, it is also helpful for biologists to find the inherent cancer-resulting mechanism and thus to develop better diagnostic methods or find better therapeutic treatments. From the data mining viewpoint, this gene selection problem is essentially a feature selection or dimensionality reduction problem. A good dimensionality reduction method should remove irrelevant or redundant gene features for classification. After removing these “non-informative” gene features, the inherent cancer-related data distribution pattern is expected to be more easily

recognized in the lower-dimensioned feature space formed by remained informative or important gene features. Consequently, a classifier modeled in the lower-dimension space can have better performance on predicting new tissues.

For example, the AML/ALL leukemia dataset has only 72 samples (tissues) with 7129 features (gene expression measurements). That means, without gene selection, we have to discriminate and classify such a few samples in such a high dimensional space. It is unnecessary or even harmful for classification because it is believed that no more than 10% of these 7129 genes are relevant to Leukemia classification [41].

As a brief summary, there are two highly-correlated challenging tasks for bioinformatics scientists:

- Gene Subset Extraction: given some tissues, extract cancer-related genes while remove irrelevant or redundant genes. Because genes should function in a complex non-independent way, so it is desirable to extract cancer-related genes together as a group than to extract them one by one.
- Cancer Classification: given a new tissue, predict if it is healthy or not; or categorize it into correct classes.

6.1.3 SVM for cancer classification

Based on [102], Support Vector Machine (SVM) is adopted for cancer classification in this work. SVM is a new generation learning system based on recent advances in statistical learning theory [19].

Due to extreme sparseness of microarray gene expression data, the dimension of input space is already high enough so that the cancer classification is already as simple as a linear separable

task [43]. It is unnecessary and even useless to transfer it to a higher implicit feature space with a non-linear kernel. As a result, in this work we adopt linear SVM [19] as the cancer classifier.

SVM is believed to be a superior model for sparse classification problems compared to other models [74,88,43]. However, the sparseness of microarray data is so extreme that even a SVM classifier is unable to achieve a satisfactory performance. A preprocessing step for gene selection can assist a SVM in finding a better separating hyperplane and thus to get more reliable classification.

6.1.4 Correlation-based feature ranking algorithms for gene selection

Gene selection can be viewed as a feature selection or dimensionality reduction problem. Currently, there are mainly two kinds of algorithms for gene selection:

Correlation-based feature ranking algorithms work in a forward selection way by ranking genes individually in terms of a correlation-based metric, and then the top ranked genes are selected to form the most informative gene subset [37,79,33].

Some commonly used ranking metrics are

Signal-to-Noise (S2N) [37]

$$w_i = \frac{|\mu_i(+) - \mu_i(-)|}{\sigma_i(+) + \sigma_i(-)}. \tag{6.1}$$

Fisher Criterion [79]

$$w_i = \frac{(\mu_i(+) - \mu_i(-))^2}{\sigma_i(+)^2 + \sigma_i(-)^2}. \tag{6.2}$$

T-Statistics [33]

$$w_i = \frac{|\mu_i(+) - \mu_i(-)|}{\sqrt{\frac{\sigma_i(+)^2}{n(+)} + \frac{\sigma_i(-)^2}{n(-)}}}. \tag{6.3}$$

In Equations 6.1-6.3, $\mu_i(+)$ and $\mu_i(-)$ are the mean values of the i^{th} gene's expression data over positive and negative samples in the training dataset, respectively. $\sigma_i(+)$ and $\sigma_i(-)$ are the corresponding standard deviations. $n(+)$ and $n(-)$ denote the numbers of positive and negative training samples, respectively. A larger w_i means that the i^{th} gene is more informative for cancer classification.

Correlation-based algorithms are straightforward and work efficiently (linear time to the size of the original gene set). However, a common drawback is that these algorithms implicitly assume that genes are orthogonal to each other and thus can only detect relations between class labels and a single gene. The mutual information such as redundancy or complementariness between multiple genes is missed out. For example, suppose that a simple XOR relationship between gene1 and gene2 is shown in Fig. 6.1:

If gene1 is expressed high and gene2 is expressed high, then the tissue maybe healthy.

If gene1 is expressed high and gene2 is expressed low, then the tissue maybe cancerous.

If gene1 is expressed low and gene2 is expressed high, then the tissue maybe cancerous.

If gene1 is expressed low and gene2 is expressed low, then the tissue maybe healthy.

However, this complementary relation cannot be grasped by a correlation metric because $\mu_i(+) - \mu_i(-)$ is zero.

6.1.5 SVM-RFE algorithm for gene selection

Backward elimination algorithms work by iteratively removing one "worst" gene at one time until the predefined size of the final gene subset is reached. In each loop, the remained genes are ranked again so that the relative rankings of genes are possible to be modified. Notice that correlation-based metrics cannot work in a back elimination way because the ranking is never modified.

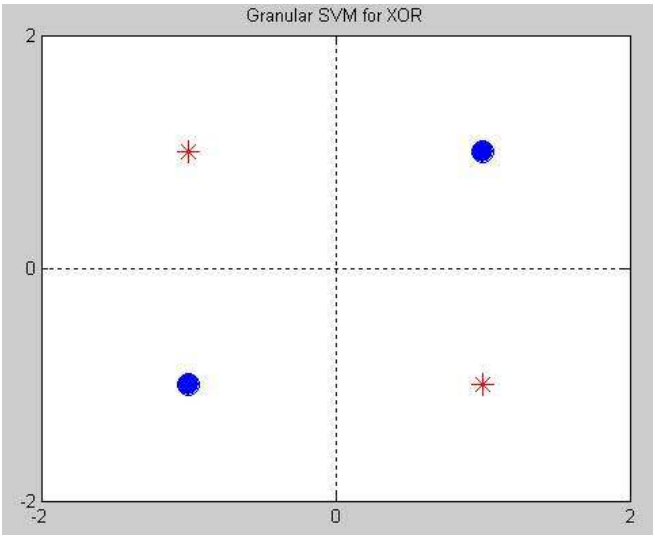


Figure. 6.1. XOR relationship between two genes can not be grasped by a correlation metric

Recently, a backward elimination algorithm called Support Vector Machine–Recursive Feature Elimination (SVM-RFE) algorithm was proposed and achieved notable performance improvement [43]. In the SVM-RFE, the removed gene should change the objective function J least.

$$J = \|w\|^2 / 2 . \tag{6.4}$$

in which w is calculated by Eq. 2.10, because only linear SVM is adopted.

The Optimal Brain Damage (OBD) algorithm [57] approximates the change of J by removing the i^{th} gene by expanding J in Taylor series to second order

$$\Delta J(i) = \frac{\partial J}{\partial w_i}\Delta w_i + \frac{\partial^2 J}{\partial w_i^2}(\Delta w_i)^2 . \tag{6.5}$$

At the optimum of J, the first order is neglected and the second order becomes

$$\Delta J(i) = (\Delta w_i)^2 . \tag{6.6}$$

Because removing the i^{th} gene means $\Delta w_i = w_i$, we can adopt w_i^2 as ranking criteria. The gene with the smallest w_i^2 has the smallest effect on classification thus is removed.

```
SVM-RFE(T,F,f,s)
initialize
    T := {training dataset}
    F := {all input features}
    f := filter_out_factor
    s := the size of final informative gene subset
begin
    while (the size of F > s)
        Train linear SVM on T in the feature space defined by F
        Rank the features of F by w_i^2 in the descending order
        if f = -1
            F2 := F - {the bottom ranked feature in F}
        elseif f=0
            F2 := F - {a number of features with largest ranks are
                removed so that the size of F2 is the closest
                smaller number of power of 2 }
        else
            F2 := F - {100f% of features in F with largest rank}
        end
        if the size of F2 < s
            adjust F2 to be composed of s top ranked features in F
        end
        F = F2
    end
    return F
end
```

Figure. 6.2. the SVM-RFE algorithm

In practical, more than one gene could be removed in one step. Fig. 6.2 describes the SVM-RFE algorithm in detail. The parameter *f*, here named "filter-out" factor, decides how many genes are removed in one step. Notice if $0<f<1$, $100f\%$ of bottom-ranked genes are removed at each step; if $f=-1$, only 1 gene is removed; if $f=0$, the least possible bottom-ranked genes are removed so that the number of remained genes is the power of 2 at the 1st step and then half of genes are removed at following steps. In each step, a new linear SVM is trained in a smaller feature space,

and thus each remained gene is assigned a new weight w_i^2 to be ranked again. This process is repeated until the pre-defined number of features is remained.

Obviously, the SVM-RFE with *f=-1* is most time-consuming. Suppose there are *d* genes, The SVM-RFE with *f=-1* works approximately in $O(d^2lgd)$ time for the ranking and the elimination process (ranking dominates here), while other SVM-RFEs with larger *f* values work in $O(dlgd)$ time.

For example, suppose a subset of 8 genes is expected to be extracted from a set of 100 genes. If *f=-1*, the SVM-RFE works in 92 steps by decreasing the size of gene set as $100 \to 99 \to 98 \to \cdots \to 10 \to 9 \to 8$; If *f=0.5*, the SVM-RFE works in 5 steps by decreasing the size of gene set as $100 \to 50 \to 25 \to 12 \to 8$; If *f=0*, the SVM-RFE works in 5 steps by decreasing the size of gene set as $100 \to 64 \to 32 \to 16 \to 8$. Finally, the genes can be ranked according to their elimination times: earlier elimination means lower rank.

6.1.6 Gene Categories

As what we estimated, there are four categories of genes in the original gene set:

- Informative genes, which are really cancer-related;
- Redundant genes, which are also cancer-related but there are some other informative genes functioning similarly but more significantly for cancer classification;
- Irrelevant genes, which are not cancer-related and their existence do not affect cancer classification;
- Noisy genes, which are not cancer-related but they have negative effects on cancer classification.

A desirable algorithm should extract genes of the 1st category while eliminate genes of the last 3 categories. However, it is difficult to fully implement this goal. Firstly, the cancer-related factors

are very possibly mixed with other non-cancer-related factors for classification. Secondly, some non-cancer-related factors may even have more significant effects on classifying the training dataset. It is actually the notorious "overfitting" problem. The thing comes even worse when the training dataset is too small to embody the inherent real data distribution, which is common for microarray gene expression data analysis. We believe that the noisy genes of the 4th category play the key role to hide the inherent cancer-related distribution and to confuse a classifier. Briefly to say, a noisy gene could have 3 kinds of negative effects on cancer diagnosis analysis:

- A noisy gene is possible to individually contribute to discriminate the training samples by some non-cancer-related factors so that it is ranked high.
- A noisy gene or a group of noisy genes may be complementary to some redundant genes so that these redundant genes are ranked higher.
- A noisy gene or a group of noisy genes may conflict with some informative genes so that these informative genes are ranked lower.

As a result, the inherent cancer-related distribution is blurred and the really informative genes are possible to be eliminated.

6.1.7 New Metrics for Gene Selection Algorithms Evaluation

For microarray gene expression data analysis, it is more desirable to select the most informative gene subset than to select a group of genes which are most informative individually. That is, a group of top-ranked genes may not be the best gene subset. Although the SVM-RFE is better than correlation-based methods because it avoids the orthogonality assumption, the preliminary simulation shows that it is unstable.

This introduces the problem of designing a reasonable evaluation mechanism to compare different gene selection algorithms. Currently, most relevant works are targeted to find a

"perfect" gene subset that produces 0 validation/testing errors (in other words, 100% accuracy) [71,41,43,37,79,33]. Because of easiness of cancer classification, if there are more than one perfect gene subsets, the smallest one is claimed to be the best one. For example, if algorithm A extracts a perfect gene subset with 10 genes and another algorithm B can only extract a perfect gene subset with 15 genes, we will say "A is better than B". This evaluation method is meaningful in some sense. However, due to the small size of available samples and high correlation among genes, the 15 genes extracted by algorithm B may be closer to uncover the real cancer-resulting mechanism. That is, finding a smallest perfect gene subset alone cannot justify a gene selection algorithm's superiority compared to others. In our work, 3 performance evaluation methods are adopted for different biological contexts.

- The special performance, which is the performance on the final gene subset with a user-defined size. Suppose the size is 15, algorithm A is better than algorithm B if and only if the SVM on the gene subset of 15 genes extracted by algorithm A has better performance than the SVM on the gene subset of 15 genes extracted by algorithm B.
- The best performance, which is the best performance taken on the final gene subsets between two user-defined sizes, one of which is the largest size and another is the smallest size. For example, if the largest size is 15 and the smallest size is 1, each algorithm will produce 15 gene subsets that are recursively embedded. From these 15 gene subsets, the best one is taken for comparison. Algorithm A is better than algorithm B if and only if the SVM on the best gene subset extracted by algorithm A has better performance than the SVM on the best gene subset extracted by algorithm B. This is the usually adopted method.

- The average performance, which is the performance averaged on the final gene subsets between two user-defined sizes, one of which is the largest size and another is the smallest size. Algorithm A is better than algorithm B if and only if the SVMs on the gene subsets extracted by algorithm A has better average performance than the SVMs on the gene subsets extracted by algorithm B.

These 3 performance evaluation methods evaluate a gene subset from different aspects. The special performance is used to evaluate a model's ability to find a high quality gene subset with the pre-defined size, the average performance demonstrates effectiveness of a model on average, while the best performance is used to select a best gene subset. We believe that these thorough comparisons will be more suitable to evaluate the final gene subsets as a whole.

6.2 Two-stage SVM-RFE algorithm

The two-stage SVM-RFE algorithm is proposed to find more informative gene subsets for more reliable cancer classification. It is designed to effectively eliminate most of irrelevant, redundant and noisy genes while keeping information loss small at the first stage, and then finely select the final key gene subset from survived genes at the second stage. Therefore, the two-stage SVM-RFE can overcome the instability problem of the SVM-RFE algorithm to achieve better algorithm utility. We have demonstrated that the two-stage SVM-RFE is an efficient algorithm because its time complexity is $O(d * log_2 d)$ where d is the size of the original gene set. More importantly, the two-stage SVM-RFE is significantly more accurate and more reliable than other gene selection methods on two gene expression datasets.

In the case of the AML/ALL dataset with 7129 gene features, compared to the "expected" SVM-RFE (that has the "expected" performance averaged on multiple SVM-RFEs with different "filter-out" factors), the two-stage SVM-RFE improves the special performance (accuracy from

86.77% to 97.06% and AUC from 84.36% to 96.43%), improves the best performance (accuracy from 96.18% to 100% and AUC from 96.00% to 100%), and improves the average performance (accuracy from 88.87%±4.99% to 93.15%±5.52% and AUC from 86.94%±6.05% to 92.27%±6.51%). The improvement on the average performance is significant (at 1%) both on accuracy and on AUC. Among the 50 genes identified by the two-stage algorithm, 30 of them have been identified previously by other algorithms as tumor-related genes. Based on biological experimental literatures, many of the 20 newly identified genes appear to be also tumor-related. The two-stage SVM-RFE seems to be able to identify more tumor-related genes.

In the case of the colon cancer dataset with over 2000 gene features, compared to the "expected" SVM-RFE, the two-stage SVM-RFE improves the special performance (accuracy from 89.84% to 96.77% and AUC from 88.65% to 96.48%), improves the best performance (accuracy from 90.16% to 96.77% and AUC from 89.31% to 96.48%), and improves the average performance (accuracy from 94.60%±6.23% to 97.13%±6.32% and AUC from 93.88%±6.96% to 96.70%±6.99%). The improvement on the average performance is significant (at 1%) both on accuracy and on AUC. The two-stage SVM-RFE effectively discovers 18 genes; all of them have been identified previously by other algorithms as tumor-related genes.

6.2.1 Instability of SVM-RFE

The reason of the good performance of the SVM-RFE is that it does not make the orthogonality assumption and thus can handle multiple features simultaneously [43]. However, many previous related research works make another implicit assumption on the SVM-RFE: a smaller "filter-out" factor should result in a better gene subset. If only one gene is eliminated at each step, the final gene subset should be the best one. Due to the efficiency reason, a larger "filter-out" factor is adopted [43]. Intuitively, it looks reasonable because more steps are executed and the

information loss in each step is smaller. However, a preliminary simulation on the AML/ALL dataset shows that the assumption is not always correct.

TABLE 6.1
SVM-RFE PERFORMANCE ON AML/ALL DATA BY TRAINING ON 38 SAMPLES AND TESTING ON 34 SAMPLES

"filter-out" factor	Leave-one-out validation accuracy on 64 genes	100 times .632 bootstrapping accuracy on 64 genes	prediction accuracy on 64 genes
0.9	1.0000	1.0000	0.9706
0.8	1.0000	1.0000	0.9412
0.7	1.0000	1.0000	0.8529
0.6	1.0000	1.0000	0.9706
0.5	1.0000	1.0000	0.9412
0.4	1.0000	1.0000	0.8529
0.3	1.0000	1.0000	0.8529
0.2	1.0000	1.0000	0.8235
0.1	1.0000	1.0000	0.9412
0	1.0000	1.0000	0.9118
-1	1.0000	1.0000	0.8529
Mean	1.0000	1.0000	0.9011
Std	0	0	0.0547

In Table 6.1, the first column is the value of "filter-out" factor *f*; the second column is leave-one-out validation accuracy on the training dataset with the gene subset of 64 top-ranked genes; the third column is .632 bootstrapping accuracy [15] (with 100 times balanced random sampling with replacement [27]) on the training dataset with the same gene subset; while the fourth column is corresponding prediction accuracy on the testing dataset. The detailed simulation context will be described in Section 6.3. The result shows that the assumption is not true in this case. For example, the SVM-RFE with f=0.8 performs much better than the SVM-RFE with f=0.2. Notice that the SVM-RFE with f=-1 is most time-consuming because only one gene is removed at each step. However, its performance on the testing dataset is even worse than the average performance of the 11 SVM-RFEs.

Furthermore, from Table 6.1, we notice that SVM-RFE is unstable because it is highly sensitive to *f*: different *f* values result in significantly different gene subsets which in turns result in SVM

classifiers with significantly different testing accuracies. For example, when the factor is 0.7, 0.3, or -1, the testing accuracy is 85.29% for 64 genes; however, when the factor is 0.8, 0.5, or 0.1, the testing accuracy is 94.12% for 64 genes. Cross-validation heuristic is usually adopted to estimate the optimal value of the unknown parameters. However, Table 6.1 also shows that the leave-one-out cross-validation accuracy is always 100%. Another common method for parameter selection is bootstrapping. Again, Table 6.1 shows that the bootstrapping accuracy is also always 100%. In other words, neither leave-one-out cross-validation nor bootstrapping is useful to predict the optimal *f* value. As a result, we doubt the utility of the SVM-RFE algorithm for microarray gene expression data analysis. The instability of the SVM-RFE algorithm may induce notorious overfitting phenomenon.

A careful exploration on the sensitivity of the SVM-RFE algorithm to *f*, the "filter-out" factor, should be helpful to find a more reliable algorithm for gene selection and cancer classification. Here "reliable" means "accurate and stable". We believe that the negative effects of noisy genes list in section 6.1.6 are the main reasons of the instability problem of the SVM-RFE algorithm. With different *f* values, SVM-RFEs eliminate different numbers of "worst" genes at each step and thus result in different gene compositions in the remaining gene set, which in turns result in different ranking for remaining genes. In this gene elimination process, there are two kinds of information loss. The first is caused by removing multiple genes at one step, while the second is caused by wrongly ranking due to negative effects of noisy genes. Although a larger *f* value may result in larger information loss of the first kind at each step, it may result in less information loss of the second kind by eliminating more irrelevant, redundant and noisy genes so that inherent cancer-related distribution is more possible to dominate in the following steps. With so many irrelevant, redundant and noisy genes, typically the information loss induced by wrongly

ranking is significantly larger than the information loss induced by larger *f* value. Therefore, a larger *f* value may result in a better gene subset.

6.2.2 Two-stage SVM-RFE algorithm: Different granules with different *f* values

To solve the instability problem, one naive way is to try exhaustive search for the optimal *f* value. However, it is extremely time-consuming. Furthermore, traditional parameter selection heuristics such as cross validation or bootstrapping cannot select good *f* values from bad *f* values, as demonstrated in the preliminary study on the AML/ALL dataset. And hence the new two-stage SVM-RFE algorithm is proposed to overcome the instability problem and still remain superior algorithm efficiency. To decrease the information loss induced by wrongly ranking, the first stage is designed to specifically eliminate irrelevant, redundant and noisy genes while keeping informative genes survived. In other words, the first stage can be viewed as a pre-filtering process: multiple SVM-RFEs with different *f* values are executed to get multiple different gene subsets which in turns are disjunctively combined into a "candidate gene set". By working multiple SVM-RFEs, a really informative gene is more possible to survive in at least one SVM-RFE's gene subset because the corresponding *f* value may eliminate noisy genes before they function for ranking the informative gene lower. Based on the similar reason, if a gene does not survive in any SVM-RFE's gene subset, it is more possible to be redundant, irrelevant, or noisy. If its size is properly selected, the candidate gene set by disjunctively combining multiple gene subsets is much smaller than the original gene set and is mainly composed of really informative genes.

After that, the information loss caused by wrongly ranking by SVM-RFE is relatively trivial. On the other hand, the information loss caused by larger *f* value is relatively large. Therefore, at the second stage, SVM-RFE with f=-1 is adopted to finely select the final key gene subset by

eliminating just one gene at a time. In this way, a better final key gene subset can be extracted. The framework of the two-stage SVM-RFE is given in Fig. 6.3.

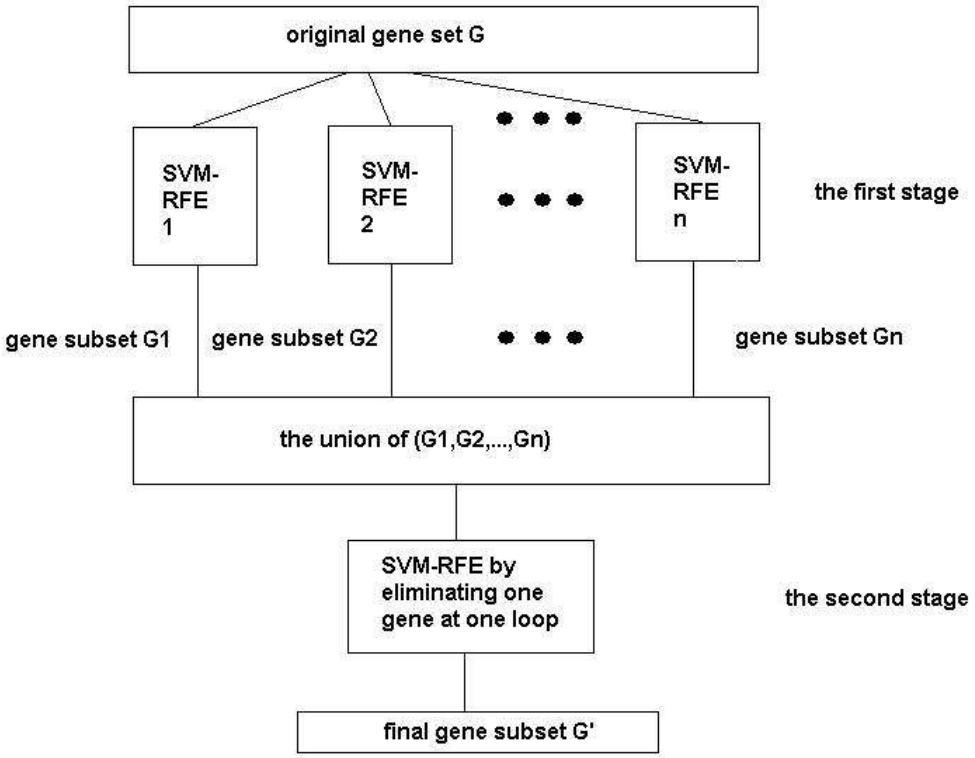


Figure. 6.3. the two-stage SVM-RFE algorithm

If the *f* value of each "child" SVM-RFE at the first stage is between [0.1, 1), the two-stage SVM-RFE works in $O(n * d * log_2 d)$ time, where n is the number of SVM-RFEs at the first stage. Usually $n << d$, which means that it has the same efficiency as the correlation-based ranking algorithms and the SVM-RFEs with $f \geq 0.1$ and runs much faster than the SVM-RFE with f=-1 ($O(d^2 * log_2 d)$).

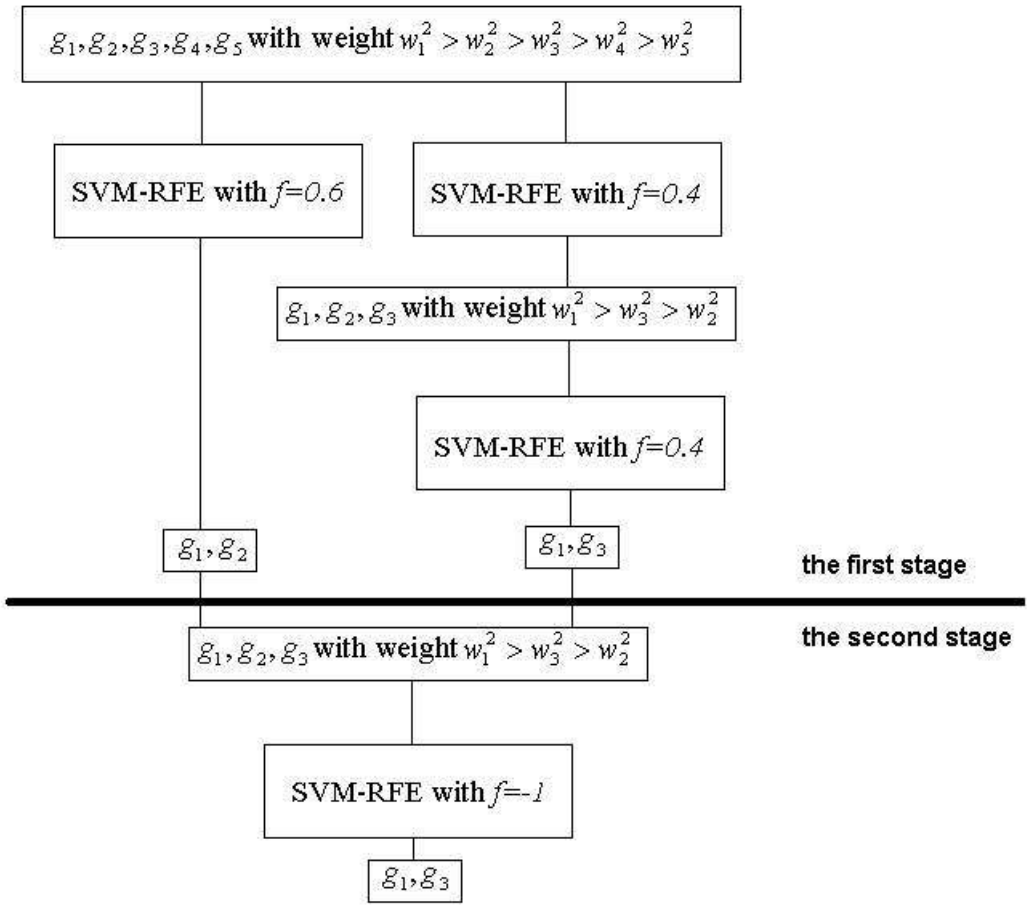


Figure. 6.4. One example to show the two-stage SVM-RFE will result in more accurate and more stable

For example, there are 5 genes named g_1, g_2, g_3, g_4, g_5. g_2 is redundant to g_3. g_4 is noisy and is complementary to g_2. The weights $w_1^2 > w_2^2 > w_3^2 > w_4^2 > w_5^2$ for the SVM in the original 5-dimension space. Notice here $w_2^2 > w_3^2$ because g_4 's complementary function on g_2. Suppose the goal is to find a gene subset of size 2, the best subset is $\{g_1, g_3\}$ but the result of the SVM-RFE with *f=0.6* is $\{g_1, g_2\}$. However, if *f=0.4*, g_4, g_5 are eliminated and then a new SVM is modeled with $w_1^2 > w_3^2 > w_2^2$ in the 3-dimension space because of the elimination of g_4, and thus the correct subset can be found. That means the SVM-RFE cannot guarantee to get the best subset in

this case. However, the two-stage SVM-RFE as showed in Fig. 6.4 can get the best result. The motivation here is to coarsely eliminate most uninformative genes while keep informative genes as many as possible at the 1st stage. And then the survived genes are finely filtered out to form the final gene subset at the 2nd stage.

6.3 Two-stage SVM-RFE simulation

6.3.1 Data description

Two datasets are used for simulations. The 1st one is the AML/ALL leukemia dataset mentioned above [58]. The training dataset consists of 38 samples (27 ALL and 11 AML) from bone marrow specimens, while the testing dataset has 34 samples (20 ALL and 14 AML), which are prepared under different experimental conditions and include 24 bone marrow and 10 blood sample specimens. The 7129 features correspond to some normalized gene expression values extracted from the microarray image: 6817 of them come from human genes and the other 312 come from control genes.

The colon cancer dataset is also used for comparison [58]. For the colon cancer dataset, there are 22 normal tissues and 40 colon cancer tissues. Gene expression information of colon cancer on more than 6500 genes were measured using oligonucleotide microarray and 2000 of them with highest minimum intensity were extracted to form a matrix of 62 tissues × 2000 gene expression values. Similar to the AML/ALL dataset, some non-human genes are included for control. Because there is no natural training/testing partition for the colon cancer dataset, all 62 samples are used for training and the leave-one-out validation is used for model evaluation [43].

The characteristics of both datasets are listed in Table 6.2.

TABLE 6.2
CHARACTERISTICS OF DATASETS USED FOR SIMULATIONS

Dataset	#samples	Ratio	#genes
AML/ALL	72	47:25	7129
Colon cancer	62	40:22	2000

6.3.2 Data preprocessing

The same as [41], the original datasets are simply normalized by decreasing the mean of corresponding gene vector from each gene expression data and then dividing by the corresponding standard deviation. As a result, each gene vector has 0 for mean and 1 for standard deviation. To avoid overfitting, the mean and standard deviation are calculated by using the training dataset. If leave-one-out validation or bootstrapping heuristic is used, the validation data is kept out from calculating these two values.

For the AML/ALL dataset, natural training/testing partition is used.

Because there is no natural training/testing partition for the colon cancer dataset or the lymphoma dataset, the leave-one-out validation is used [43]: in each fold, one sample is leaved for validation and other samples are used for training. Another evaluation heuristic adopted is balanced .632 bootstrapping [15]: random sampling with replacement is repeated for 100 times on each of the two datasets. Each tissue sample appears exactly 100 times in the computation to reduce variance [27].

6.3.3 Modeling

In the simulations, nine SVM-RFEs with $f \in \{0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9\}$ are used at the 1st stage and one SVM-RFE with $f=-1$ is used at the 2nd stage. Fig. 6.5 shows the pseudocode of the two-stage SVM-RFE used in the simulations. Notice each gene subset (G_i and G') has the

same size *s*. *s=64* for the AML/ALL dataset, and *s=4* for the colon cancer dataset. *Gc* is the candidate gene set.

For fairness of comparison, the SVM-RFE with each *f* value is executed again to extract a gene set with the same size as *Gc* firstly, and then *f=-1* is used to extract the final gene subset.

```
Two-stage SVM-RFE(T,G,s)
initialize
    T := {training dataset}
    G := {all genes}
    s := the size of final informative gene subset
begin
    for i :=1 to 9 step by 1
        G_i := SVM-RFE(T,G,0.1*i,s)
    end
    Gc := union of G_1, G_2, …, G_9
    G' := SVM-RFE(T,Gc,-1,s)
    return G'
end
```

Figure. 6.5. pseudocode of the two-stage SVM-RFE algorithm

Seven different algorithms are compared:

- The two-stage SVM-RFE.
- Signal to Noise (S2N) correlation-based ranking algorithm with weights calculated by Eq. 6.1.
- Fisher Criterion (FC) correlation-based ranking algorithm with weights calculated by Eq. 6.2.
- T-Statistics (TS) correlation-based ranking algorithm with weights calculated by Eq. 6.3.
- "Default" SVM-RFE with f=0 which is suggest by [43] and is actually adopted by many previous related works.
- "Expected" SVM-RFE, the performance of which is the average performance of the 10 SVM-RFEs with different f values in the field of [0,0.9]. It is the expected performance of SVM-RFE because of the instability of SVM-RFE.

- "Slowest" SVM-RFE with f=-1 which is the most time-consuming SVM-RFE.

For each of the seven algorithms, the special performance, the best performance and the average performance defined are reported.

- For the special performance, $s = 64$ for the AML/ALL dataset and $s = 4$ for the colon cancer dataset.
- For the best performance, $1 \le s \le 64$ for the AML/ALL dataset and $1 \le s \le 4$ for the colon cancer dataset. (That is, for the AML/ALL dataset, there are 64 final gene subsets whose sizes decrease one by one from 64 to 1, and the best performance of them is reported. For the colon cancer dataset, there are 4 final gene subsets.)
- For the average performance, $1 \le s \le 64$ for the AML/ALL dataset and $1 \le s \le 18$ for the colon cancer dataset. (For the AML/ALL dataset, there are 64 final gene subsets; for the colon cancer dataset, there are 18 final gene subsets. Notice here our goal to select more than 4 gene subsets for the colon cancer dataset is to make the average performance statistically significant.)

The selection of s value is based on the practical utilities of the extracted gene subsets: The smallest number of genes is desirable for further biological study because it is very expensive or even impractical for biologists to pursue cancer study on a large number of genes; On the other hand, the prediction is not accurate/reliable if too few genes are selected. In previous research works, the s value is usually decided arbitrarily or by a biologist. However, we notice that it is difficult for biologists to decide such a value precisely. Instead of that, it is easier for biologists to decide a field instead of a value. In this work, the lower-bound of the field is always assumed to be 1. So the biologist only needs to decide the upper-bound.

In the tables of the performance evaluation results, "Best" denotes the "Best Performance", "Mean" denotes the "Average Performance", and "Std" denotes the standard deviation of the "Average Performance", respectively.

Furthermore, a paired t-test for one tailed hypothesis is conducted to demonstrate the significance of the improvement by the two-stage SVM-RFE: "p-value" in the accuracy evaluation tables denotes the significance level the null hypothesis $\mu_x \leq \mu_y$ can be rejected, where *x* is the vector of the accuracy results of the two-stage SVM-RFE, and *y* is the vector of the accuracy results of the compared algorithm.

6.3.4 Statistical Analysis on the AML/ALL dataset

TABLE 6.3
ACCURACY COMPARISON ON THE 7 DIFFERENT ALGORITHMS ON THE AML/ALL DATASET BY TRAINING ON 38 SAMPLES AND TESTING ON 34 SAMPLES

models	64 genes	Best (<=64)	Mean (<=64)	Std (<=64)	p-value by ttest (<=64)
S2N correlation	0.9706	0.9706	0.9063	0.0425	0.0010
FC correlation	0.9118	0.9118	0.8465	0.0380	<0.0001
TS correlation	0.9118	0.9118	0.8585	0.0370	<0.0001
"default" SVM-RFE	0.9118	0.9706	0.8980	0.0671	0.0006
"expected" SVM-RFE	0.8677	0.9618	0.8887	0.0499	0.0019
"slowest" SVM-RFE	0.8529	0.9412	0.8768	0.0363	<0.0001
two-stage SVM-RFE	0.9706	1.0000	0.9315	0.0552	N/A

TABLE 6.4
AREA UNDER ROC CURVE COMPARISON ON THE 7 DIFFERENT ALGORITHMS ON THE AML/ALL DATASET BY TRAINING ON 38 SAMPLES AND TESTING ON 34 SAMPLES

models	64 genes	Best (<=64)	Mean (<=64)	Std (<=64)	p-value by ttest (<=64)
S2N correlation	0.9643	0.9643	0.8868	0.0504	0.0001
FC correlation	0.8929	0.8929	0.8183	0.0436	<0.0001
TS correlation	0.8929	0.8929	0.8309	0.0427	<0.0001
"default" SVM-RFE	0.8929	0.9750	0.8798	0.0824	0.0002
"expected" SVM-RFE	0.8436	0.9600	0.8694	0.0605	0.0043
"slowest" SVM-RFE	0.8214	0.9286	0.8525	0.0424	<0.0001
two-stage SVM-RFE	0.9643	1.0000	0.9227	0.0651	N/A

For the AML/ALL dataset, the two-stage SVM-RFE extracts 169 genes for the candidate gene set at the first stage, and then it continues to eliminate one gene at each step until final 64 genes are selected at the second stage.

Both accuracy and AUC comparisons from Tables 6.3-6.4 show that

- the two-stage SVM-RFE is more reliable than correlation-based methods. Compared to S2N, the two-stage SVM-RFE has the same special performance (accuracy=97.06% and AUC=96.43%), improves the best performance (accuracy from 97.06% to 100% and AUC from 96.43% to 100%), and improves the average performance (accuracy from 90.63%±4.25% to 93.15%±5.52% and AUC from 88.68%±5.04% to 92.27%±6.51%). The improvement on the average accuracy is significant (at 1%) both on accuracy and on AUC.
- the two-stage SVM-RFE is more reliable than the "default" SVM-RFE by improving the special performance (accuracy from 91.18% to 97.06% and AUC from 89.29% to 96.43%), by improving the best performance (accuracy from 97.06% to 100% and AUC from 97.50% to 100%), and by improving the average performance (accuracy from 89.80%±6.71% to 93.15%±5.52% and AUC from 87.98%±8.24% to 92.27%±6.51%). The improvement on the average performance is significant (at 1%) both on accuracy and on AUC.
- the two-stage SVM-RFE is more reliable than the "expected" SVM-RFE by improving the special performance (accuracy from 86.77% to 97.06% and AUC from 84.36% to 96.43%), by improving the best performance (accuracy from 96.18% to 100% and AUC from 96.00% to 100%), and by improving the average performance (accuracy from 88.87%±4.99% to 93.15%±5.52% and AUC from 86.94%±6.05% to 92.27%±6.51%).

The improvement on the average performance is significant (at 1%) both on accuracy and on AUC.

- the two-stage SVM-RFE is even more reliable than the "slowest" SVM-RFE by improving the special performance (accuracy from 85.29% to 97.06% and AUC from 82.14% to 96.43%), by improving the best performance (accuracy from 94.12% to 100% and AUC from 92.86% to 100%), and by improving the average performance (accuracy from 87.68%±3.63% to 93.15%±5.52% and AUC from 85.25%±4.24% to 92.27%±6.51%). The improvement on the average performance is significant (at 1%) both on accuracy and on AUC. Moreover, two-stage SVM-RFE is significantly faster than the "slowest" SVM-RFE as analyzed above.

The results also demonstrate that the two-stage SVM-RFE is the best algorithm in terms of balance ability with the highest AUC values. The superior balance ability of the two-stage SVM-RFE validates our estimation: the two-stage SVM-RFE eliminates irrelevant, redundant and noisy genes more effectively because it extracts positive-related genes and negative-related genes in balance. That means two-stage SVM-RFE takes advantage of the mutual information between genes more effectively than other algorithms.

TABLE 6.5
ACCURACY OF TWO-STAGE SVM-RFES WITH DIFFERENT "FILTER-OUT" FACTORS AT THE SECOND STAGE ON THE AML/ALL DATASET BY TRAINING ON 38 SAMPLES AND TESTING ON 34 SAMPLES

"filter-out" factor	64 genes	Best (<=64)	Mean (<=64)	Std (<=64)	p-value by ttest (<=64)
-1	0.9706	1.0000	0.9315	0.0552	N/A
-2	0.9412	0.9412	0.8493	0.0575	<0.0001
-3	0.8235	0.9412	0.8612	0.0551	<0.0001
-4	0.9412	1.0000	0.9283	0.0497	0.2789
-5	0.9118	0.9706	0.8998	0.0370	0.0010
-6	0.8824	0.9706	0.9007	0.0394	0.0014

TABLE 6.6
AREA UNDER ROC CURVE OF TWO-STAGE SVM-RFES WITH DIFFERENT "FILTER-OUT" FACTORS AT THE SECOND STAGE ON THE AML/ALL DATASET BY TRAINING ON 38 SAMPLES AND TESTING ON 34 SAMPLES

"filter-out" factor	64 genes	Best (<=64)	Mean (<=64)	Std (<=64)	p-value by ttest (<=64)
-1	0.9643	1.0000	0.9227	0.0651	N/A
-2	0.9286	0.9286	0.8178	0.0693	<0.0001
-3	0.7964	0.9393	0.8425	0.0684	<0.0001
-4	0.9286	1.0000	0.9220	0.0556	0.4567
-5	0.8929	0.9643	0.8824	0.0420	0.0005
-6	0.8571	0.9643	0.8811	0.0468	0.0003

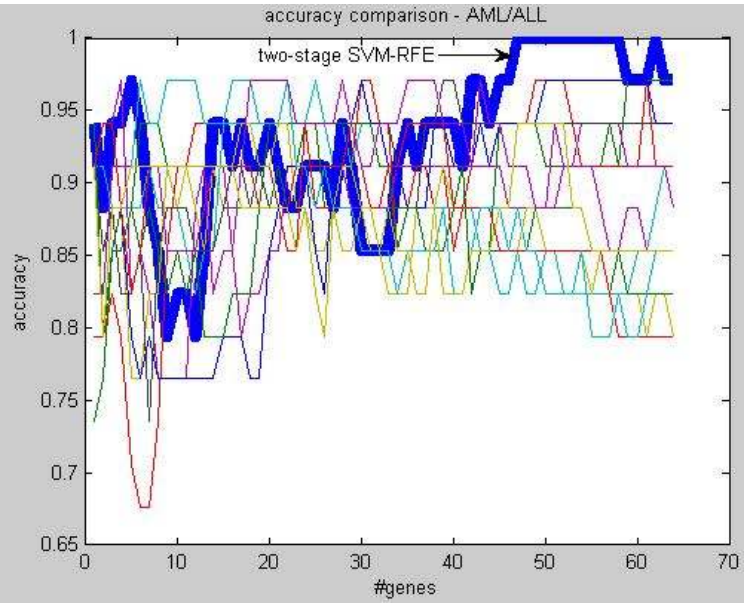


Figure. 6.6. two-stage SVM-RFE extracts most reliable gene subsets on the AML/ALL dataset (The prediction accuracy is 100% from 47 genes to 58 genes)

To justify that the first stage effectively eliminates irrelevant, redundant and noisy genes, we also try different *f* values at the second stage. Due to small number (169) of the candidate genes, at the second stage, we try to remove 1-6 gene(s) at each step. Tables 6.5-6.6 show that the SVM-RFE with *f*=-1 has the best performance. That means, for the AML/ALL dataset, the information loss induced by large f values dominates while the information loss induced by wrongly ranking is relatively trivial at the second stage.

For the AML/ALL dataset, gene subsets with 64 genes to 1 gene are extracted. In Fig. 5.6, each curve denotes the testing accuracy with different numbers of genes for an algorithm. As demonstrated, no algorithm can find high quality gene subsets if the upper-bound is too small and the extracted gene subsets are not reliable. With the increase of number of genes, two-stage SVM-RFE can select better gene subsets for more reliable AML/ALL classification than other algorithms.

6.3.5 Biological Analysis on the AML/ALL dataset

TABLE 6.7
PERFORMANCE OF TWO-STAGE SVM-RFE ON THE AML/ALL DATASET BY TRAINING ON 38 SAMPLES AND TESTING ON 34 SAMPLES

#genes	accuracy	auc	#genes	accuracy	auc	#genes	accuracy	auc
64	0.9706	0.9643	42	0.9706	0.9643	20	0.9412	0.9393
63	0.9706	0.9643	41	0.9118	0.9036	19	0.9118	0.9036
62	1	1	40	0.9412	0.9393	18	0.9118	0.9036
61	0.9706	0.9643	39	0.9412	0.9393	17	0.9412	0.9393
60	0.9706	0.9643	38	0.9412	0.9393	16	0.9118	0.9036
59	0.9706	0.9643	37	0.9412	0.9393	15	0.9412	0.9393
58	1	1	36	0.9118	0.9036	14	0.9412	0.9393
57	1	1	35	0.9412	0.9393	13	0.8529	0.8214
56	1	1	34	0.9118	0.9036	12	0.7941	0.7500
55	1	1	33	0.8529	0.8321	11	0.8235	0.7857
54	1	1	32	0.8529	0.8321	10	0.8235	0.7857
53	1	1	31	0.8529	0.8321	9	0.7941	0.7500
52	1	1	30	0.8529	0.8321	8	0.8529	0.8214
51	1	1	29	0.9118	0.9036	7	0.8824	0.8571
50	1	1	28	0.9412	0.9393	6	0.9412	0.9286
49	1	1	27	0.8824	0.8679	5	0.9706	0.9643
48	1	1	26	0.9118	0.9036	4	0.9412	0.9393
47	1	1	25	0.9118	0.9036	3	0.9412	0.9393
46	0.9706	0.9643	24	0.9118	0.9036	2	0.8824	0.8679
45	0.9706	0.9643	23	0.8824	0.8786	1	0.9412	0.9286
44	0.9412	0.9393	22	0.8824	0.8786			
43	0.9706	0.9643	21	0.9118	0.9143			

TABLE 6.8
MOST IMPORTANT GENES SELECTED BY TWO-STAGE SVM-RFE ON THE AML/ALL DATASET (GENES WITH * ARE PREVIOUSLY IDENTIFIED GENES)

rank	GAN	Description of Gene Function	References
1	M23197*	Human differentiation antigen (CD33)	[41][11][55][76][101]
2	X85116*	Integral membrane protein (Protein 7.2b)	[41][11][55]
3	X95735*	Homo sapiens Zyxin	[41][43][11][17]
4	U22376*	Human C-myb	[41][11][17][65][78]
5	D49950*	Interferon-gamma inducing factor (IL-18)	[11][17][40][32]
6	Y12670*	Leptin receptor gene-related protein	[41]
7	M37435	macrophage-specific colony-stimulating factor (CSF-1)	
8	M24400*	Human chymotrypsinogen	[105]
9	U50136*	Huaman leukotriene C4 synthase (LTC4S)	[41][11][17][55]
10	M55150*	Human fumarylacetoacetate hydrolase	[41][11][17]
11	M83652*	Complement omponent properdin	[41][11]
12	M29610*	Glycophorin E	[6]
13	M19507*	Myeloperoxidase	[11][55]
14	X06948*	High affinity IgE receptor alpha-subunit (FcERI)	[72][54]
15	X70297*	Nicotinic acetylcholine receptor alpha-7 subunit	[11]
16	L08246*	Myeloid cell differentiation protein (MCL1)	[41]
17	U82759*	Homeodomain protein HoxA9	[41][43][17][55]
18	X16901*	RAP30 subunit of transcription initiation factor RAP30/74	[4]
19	M60298*	Erythrocyte membrane protein band 4.2 (EPB42)	[18]
20	M62762*	Vacuolar H+ ATPase proton channel subunit	[41][11][17]
21	U92459*	Metabotropic glutamate receptor 8	[64]
22	U63289*	RNA-binding protein CUG-BP/hNab50 (NAB50)	[17]
23	U43292*	MDS1B (MDS1)	[34]
24	M96326*	Azurocidin	[41][11]
25	J04621	Heparan sulfate proteoglycan core protein	
26	M81933*	Cdc25A	[17]
27	M86406*	Skeletal muscle alpha 2 actinin (ACTN20)	[52]
28	X81479	EMR1 hormone receptor	
29	M63138*	Cathepsin D (catD)	[41][11][55]
30	X13839*	Vascular smooth muscle alpha-actin	[118]
31	M16038*	Tyrosine kinase encoded by lyn mRNA	[41][11][17]
32	L49229*	Retinoblastoma susceptibility protein (RB1)	[22][66]
33	M68891*	GATA-binding protein (GATA2)	[9]
34	U25128*	PTH2 parathyroid hormone receptor	[69]
35	M61853*	Cytochrome P4502C18 (CYP2C18)	[55]
36	U21689*	Glutathione S-transferase-P1c	[87]
37	M84526*	Adipsin/complement factor D	[41][11][55]
38	M20902*	Apolipoprotein C-I (VLDL)	[55]
39	X05409*	Mitochondrial aldehyde dehydrogenase I	[53]
40	X04085*	Catalase (EC 1.11.1.6) 5'flank and exon 1	[41]
41	D21851*	Mitochondrial leucyl-tRNA synthetase	[51]
42	M98539*	Prostaglandin D2 synthase	[84]
43	M22960*	Protective protein	[11] [55]
44	X55668*	Proteinase 3	[11]
45	J05500*	Beta-spectrin (SPTB)	[59]
46	U37055*	Hepatocyte growth factor-like protein	[80][77]
47	M26708*	Prothymosin alpha (ProT-alpha),	[103]
48	Y10207*	CD171 protein	[55]
49	X58431*	Homeobox protein encoded by Hox2.2 gene	[11]
50	X13294*	A-Myb	[31]

Table 6.7 shows the testing performance in terms of accuracy and AUC on the AML/ALL dataset for gene subsets with different sizes from 64 genes to 1 gene. Each gene subset is achieved by removing one “worst” gene from its closest larger gene subset. Notice that from 63 genes to 62 genes or from 59 genes to 58 genes, the testing accuracy is improved. It shows

maybe a noisy gene is removed here. Similarly, the testing accuracy is deteriorated from 62 genes to 61 genes or from 47 genes to 46 genes because maybe an informative gene is removed. Because the testing accuracy for the AML/ALL dataset is 100% for the gene subsets from 58 genes to 47 genes selected by the two-stage SVM-RFE, here we select top 50 genes to analyze their biological functions related to leukemia classification in Table 6.8. The "*" signed genes in the case of AML/ALL are also identified by the other approaches in previous works, while the other ones are newly found by the two-stage SVM-RFE.

Besides these common genes in the case of AML/ALL, many of the novel genes discovered by the two-stage SVM-RFE have already been demonstrated in literatures that they are directly or indirectly related with cancer. For instance, human chymotrypsinogen (Rank No. 8 in Table 6.8) is one of protease proenzymes, which show remarkable selective effects that result in growth inhibition of tumor cells with metastatic potential [75], glycophorin E (Rank No. 12 in Table 6.8) is in glycophorin family, which is related to erythroid differentiation in the murine erythroleukemia cell line [92], inactivation of the retinoblastoma gene (Rank No. 32 in Table 6.8) is a common event in parathyroid tumorigenesis [22,66], the hepatocyte growth factor-like protein (Rank No. 46 in Table 6.8) interacts with RON [80], which is strongly expressed in renal oncocytomas and renal cell carcinoma [77].

6.3.6 Statistical Analysis on the colon cancer dataset

For the colon cancer dataset, the two-stage SVM-RFE extracts 18 genes for the candidate gene set at the first stage, and then it continues to eliminate one gene at each step until final 4 genes are selected at the second stage.

With leave-one-out validation, Tables 6.9-6.10 show that

- the two-stage SVM-RFE is more reliable than correlation-based methods. Compared to S2N, the two-stage SVM-RFE improves the special performance (accuracy from 85.48% to 96.77% and AUC from 83.64% to 96.48%), improves the best performance (accuracy from 85.48% to 96.77% and AUC from 83.64% to 96.48%), and improves the average performance (accuracy from 86.11%±2.54% to 97.13%±6.32% and AUC from 84.06%±3.47% to 96.70%±6.99%). The improvement on the average performance is significant (at 1%) both on accuracy and on AUC.
- the two-stage SVM-RFE is more reliable than the "default" SVM-RFE by improving the special performance (accuracy from 88.71% to 96.77% and AUC from 86.14% to 96.48%), by improving the best performance (accuracy from 91.94% to 96.77% and AUC from 91.70% to 96.48%), and by improving the average performance (accuracy from 95.07%±5.89% to 97.13%±6.32% and AUC from 94.19%±6.06% to 96.70%±6.99%). The improvement on the average performance is significant (at 1%) both on accuracy and on AUC.
- the two-stage SVM-RFE is more reliable than the "expected" SVM-RFE by improving the special performance (accuracy from 89.84% to 96.77% and AUC from 88.65% to 96.48%), by improving the best performance (accuracy from 90.16% to 96.77% and AUC from 89.31% to 96.48%), and by improving the average performance (accuracy

from 94.60%±6.23% to 97.13%±6.32% and AUC from 93.88%±6.96% to 96.70%±6.99%). The improvement on the average performance is significant (at 1%) both on accuracy and on AUC.

- the two-stage SVM-RFE is even more reliable than the "slowest" SVM-RFE by improving the special performance (accuracy from 95.16% to 96.77% and AUC from 95.23% to 96.48%), by improving the best performance (accuracy from 95.16% to 96.77% and AUC from 95.23% to 96.48%), and by improving the average performance (accuracy from 96.06%±5.70% to 97.13%±6.32% and AUC from 95.86%±5.82% to 96.70%±6.99%). Although the improvement on the average performance is not significant (p-value=0.0518 on accuracy and p-value=0.1494 on AUC), as we claimed before, two-stage SVM-RFE is much faster than the "slowest" SVM-RFE.

TABLE 6.9
ACCURACY COMPARISON ON THE 7 DIFFERENT ALGORITHMS ON THE COLON CANCER DATASET BY LEAVE-ONE-OUT VALIDATION

models	4 genes	Best (<=4)	Mean (<=18)	Std (<=18)	p-value by ttest (<=18)
S2N correlation	0.8548	0.8548	0.8611	0.0254	<0.0001
FC correlation	0.8226	0.8548	0.8611	0.0266	<0.0001
TS correlation	0.7742	0.7903	0.8181	0.0302	<0.0001
"default" SVM-RFE	0.8871	0.9194	0.9507	0.0589	0.0012
"expected" SVM-RFE	0.8984	0.9016	0.9460	0.0623	0.0076
"slowest" SVM-RFE	0.9516	0.9516	0.9606	0.0570	0.0518
two-stage SVM-RFE	0.9677	0.9677	0.9713	0.0632	N/A

TABLE 6.10
AREA UNDER ROC CURVE COMPARISON ON THE 7 DIFFERENT ALGORITHMS ON THE COLON CANCER DATASET BY LEAVE-ONE-OUT VALIDATION

models	4 genes	Best (<=4)	Mean (<=18)	Std (<=18)	p-value by ttest (<=18)
S2N correlation	0.8364	0.8364	0.8406	0.0347	0.0000
FC correlation	0.7807	0.8261	0.8390	0.0345	<0.0001
TS correlation	0.7841	0.7966	0.8130	0.0253	<0.0001
"default" SVM-RFE	0.8614	0.9170	0.9419	0.0606	0.0035
"expected" SVM-RFE	0.8865	0.8931	0.9388	0.0696	0.0087
"slowest" SVM-RFE	0.9523	0.9523	0.9586	0.0582	0.1494
two-stage SVM-RFE	0.9648	0.9648	0.9670	0.0699	N/A

With balanced .632 bootstrapping (100 times random sampling with replacement), similar improvement is observed in Tables 6.11-6.12.

The same as the AML/ALL dataset, the results also demonstrate two-stage SVM-RFE is the best model in terms of balance ability, while the gene subsets extracted by the original SVM-RFEs are biased. We also try different f values at the second stage on the colon cancer dataset. Similar result to the AML/ALL dataset is observed. Due to space limit, we skip to report the result here.

TABLE 6.11
ACCURACY COMPARISON ON THE 7 DIFFERENT ALGORITHMS ON THE COLON CANCER DATASET BY 100 TIMES BOOTSTRAPPING

models	4 genes	Best (<=4)	Mean (<=18)	Std (<=18)	p-value by ttest (<=18)
S2N correlation	0.8447	0.8513	0.8314	0.0098	<0.0001
FC correlation	0.8302	0.8504	0.8312	0.0101	<0.0001
TS correlation	0.7609	0.7714	0.8011	0.0319	<0.0001
"default" SVM-RFE	0.8995	0.8995	0.9236	0.0574	<0.0001
"expected" SVM-RFE	0.8849	0.8869	0.9263	0.0622	0.0106
"slowest" SVM-RFE	0.9394	0.9394	0.9487	0.0633	0.0706
two-stage SVM-RFE	0.9588	0.9588	0.9572	0.0681	N/A

TABLE 6.12
AREA UNDER ROC CURVE COMPARISON ON THE 7 DIFFERENT ALGORITHMS ON THE COLON CANCER DATASET BY 100 TIMES BOOTSTRAPPING

models	4 genes	Best (<=4)	Mean (<=18)	Std (<=18)	p-value by ttest (<=18)
S2N correlation	0.8313	0.8411	0.8325	0.0193	<0.0001
FC correlation	0.7944	0.8187	0.8292	0.0156	<0.0001
TS correlation	0.7879	0.7942	0.8218	0.0319	<0.0001
"default" SVM-RFE	0.8997	0.8997	0.9279	0.0592	<0.0001
"expected" SVM-RFE	0.8901	0.8911	0.9326	0.0683	0.0083
"slowest" SVM-RFE	0.9461	0.9461	0.9526	0.0634	0.1626
two-stage SVM-RFE	0.9710	0.9710	0.9589	0.0719	N/A

6.3.7 Biological Analysis on the colon cancer dataset

Similarly, with leave-one-out cross validation (Table 6.13) or .632 bootstrapping (Table 6.14), top 18 genes selected by the two-stage SVM-RFE can induce highly accurate classification. Therefore, we report them in Table 6.15 for biological analysis. All of the 18 genes have been previously reported in bioinformatics literature as colon cancer-related genes. Gene No. 11 (myosin light chain gene) in Table 6.15 is an interesting example. Recent research has indicated that tumor necrosis factor-induced cytoskeletal rearrangement driven by activity of myosin light chain kinase (MLCK), which may affect expression of myosin light chain (MLC), is necessary for tumor necrosis factor-dependent nuclear factor kappa-B activation and amplification of pro-

survival signals [104], which may influence tumor growth. Therefore, the MLC gene seems to be an informative gene on tumor development. Unlike the original SVM-RFE algorithm that eliminates this gene from its top ranking [43], our algorithm ranks it top 11; this gene is also ranked high by other approaches [11,86,38].

TABLE 6.13
PERFORMANCE OF TWO-STAGE SVM-RFE ON THE COLON CANCER DATASET BY LEAVE-ONE-OUT VALIDATION

#genes	accuracy	auc	#genes	accuracy	auc	#genes	accuracy	auc
18	1	1	12	1	1	6	1	1
17	1	1	11	1	1	5	1	1
16	1	1	10	0.9839	0.9773	4	0.9677	0.9648
15	1	1	9	1	1	3	0.9032	0.8739
14	1	1	8	1	1	2	0.8548	0.8159
13	1	1	7	1	1	1	0.7742	0.7739

TABLE 6.14
PERFORMANCE OF TWO-STAGE SVM-RFE ON THE COLON CANCER DATASET BY 100 TIMES BOOTSTRAPPING

#genes	accuracy	auc	#genes	accuracy	auc	#genes	accuracy	auc
18	0.9675	0.9735	12	0.9952	0.9971	6	0.9789	0.9874
17	0.9741	0.9794	11	0.9952	0.9971	5	0.9820	0.9889
16	0.9807	0.9845	10	0.9864	0.9889	4	0.9588	0.9710
15	0.9811	0.9858	9	0.9908	0.9925	3	0.8806	0.8706
14	0.9882	0.9907	8	0.9930	0.9946	2	0.8574	0.8325
13	0.9903	0.9928	7	0.9974	0.9978	1	0.7323	0.7346

TABLE 6.15
MOST IMPORTANT GENES SELECTED BY TWO-STAGE SVM-RFE ON THE COLON DATASET

Rank	GAN	Description of Source or Gene Function	Possible functions to colon cancer
1	H08393	Soares infant brain 1NIB	[43][11][38]
2	H64807	Alu repetitive element	[43][55][38]
3	T57882	Stratagene fetal spleen	[38]
4	M92287	Cyclin D3 (CCND3)	[38]
5	H55916	Peptidyl-prolyl cis-trans isomerase	[11][38]
6	T62947	Stratagene lung	[43][11][55][117][38]
7	R88740	ATP Synthase coupling Factor 6	[43][55][86][38]
8	H01418	Soares placenta Nb2HP	[38]
9	H49870	Soares adult brain N2b5HB55Y	[86][38]
10	T79831	Protein-tyrosine phosphatase	[38]
11	J02854	20-kDa myosin light chain (MLC-2)	[43][11][86][38]
12	H16096	Soares infant brain 1NIB	[55][38]
13	Z50753	GCAP-II/uroguanylin precursor	[11][55][86][38]
14	J04102	Erythroblastosis virus oncogene homolog 2 (ets-2)	[38]
15	H81558	TAR1 repetitive element	[43][38]
16	R87126	Alu repetitive element	[11][86][117][38]
17	M76378	Cysteine-rich protein (CRP)	[11][55][86][38]
18	U00968	Srebp-1	[55][38]

6.3.8 Summary on two-stage SVM-RFE simulation

- the two-stage SVM-RFE is the best algorithm for gene selection compared to other 6 methods for three datasets in terms of accuracy. That means two-stage SVM-RFE is an effective algorithm for cancer classification.
- the two-stage SVM-RFE is the best algorithm for gene selection compared to other 6 methods for three datasets in term of AUC. That means the two-stage SVM-RFE effectively takes advantage of the correlation among genes to select the informative gene subset.
- S2N algorithm performs well on the AML/ALL dataset but performs badly on the colon cancer dataset. It means the correlations among gene expression data are more important for the colon cancer diagnosis than for the AML/ALL diagnosis.
- the "slowest" SVM-RFE performs well on the colon cancer dataset but performs badly on the AML/ALL dataset. It once again shows that the SVM-RFE is an unstable algorithm.

If the *f* value of each "child" SVM-RFE in the 1st stage is between [0.1, 1), the two-stage SVM-RFE works in $O(d)$ time. That means it has the same efficiency as the S2N algorithm or the SVM-RFEs with $f>0.1$ and runs much faster than the SVM-RFE with $f=-1$ ($O(d^2)$).

The two-stage SVM-RFE algorithm has identified a subset of genes, which are consistent with the genes (100% identical in the case of colon cancer and 60% identical in the case of AML/ALL) discovered by other conventional algorithms. Many of the common genes are directly or indirectly related to tumor activities. For instance in the case of AML/ALL, high-level CD33 (differentiation antigen) activity is observed in AML[76,101], C-myb (a transcription factor) is associated with cell apoptosis (programmed cell death; disruption of C-myb expression can disrupt tumor growth [65,78]), and IL-18 (Interferon-gamma inducing factor, a cytokine mainly produced by macrophages) affect T-cell activation [40,32], which can influence tumor development.

Previous works on the SVM-RFE assume that the smaller "filter-out" factor should result in the better performance. If at each step only one gene is eliminated, the final gene subset should be the best one. Only due to the efficiency reason, larger "filter-out" factor is adopted [43]. Our work shows that the assumption is not always correct because the SVM-RFE with *f=-1* (that is, at each step only one "worst" gene is eliminated) cannot always achieve better performance than SVM-RFE with a larger "filter-out" factor. Actually, it is even worse on the AML/ALL dataset. As a result, selecting larger "filter-out" factor is not only due to efficiency reason, but even necessary for effectiveness reason. Currently, the "filter-out" factor is decided arbitrarily [43,33]. Unfortunately, our work also shows that the SVM-RFE is unstable: SVM-RFEs with different "filter-out" factors have significantly different performances. And there is no simple monotonic relation between the "filter-out" factor and the performance because of the complex correlations among genes. As a result, it is difficult for the original SVM-RFE to find the optimal "filter-out" factor.

Therefore, to find a more informative gene subset for more reliable prediction, the two-stage SVM-RFE algorithm is presented in this work. Firstly, the two-stage SVM-RFE algorithm avoids the problem to select the optimal "filter-out" factor and thus overcome the instability problem of the original SVM-RFE algorithm. Secondly, the two-stage SVM-RFE has the same time complexity (linear to the size of the original gene set) as the correlation-based S2N ranking algorithm and the original SVM-RFE algorithm (except the "slowest" SVM-RFE, which runs in quadratic time). More importantly, the two-stage SVM-RFE performs much better in terms of generalization capability (more accurate to predict new samples) on two publicly available gene expression datasets. Because of the inherent advantage to discriminate informative features from noisy or redundant features, we expect that this superior performance could also be true in

processing other similar datasets with extreme sparseness such as Web text mining, image pattern recognition, and other bioinformatics problems.

To the best of our knowledge, this is the 1st work to point out the instability problem of the SVM-RFE algorithm resulted by choosing different "filter-out" factors. Similarly, we expect that the same instability problem also exists in other RFE algorithms. This is another interesting future work to explore, which may generalize the similar two-stage SVM-RFE algorithms in other areas. By increasing stability and accuracy, this two-stage algorithm can predict and classify tumor types or subtypes more precisely and it also has potential to identify more tumor-related genes. In the case of AML/ALL, 50 genes are identified by the two-stage algorithm, where 30 of them have been identified previously by other algorithms as tumor-related genes, and many of the 20 newly identified genes appear also to be tumor-related. In the case of colon cancer, this two-stage SVM-RFE effectively discovers 18 genes; all of them have been identified previously by other algorithms as tumor-related genes.

In summary, the two-stage SVM-RFE has advantages over other conventional algorithms. Of course, the newly identified genes by this algorithm need to be further confirmed experimentally, which may generate more insights for cancer mechanism, treatment and study. Nevertheless, this predication of these cancer-informative genes will help to stimulate and guide detailed studies on the gene functions.

6.4 GSVM-RFE algorithm

6.4.1 Inflexibility of current algorithms

Both correlation-based algorithms (S2N as an example) and backward elimination algorithms (SVM-RFE as an example) are inflexible in that the same gene subset is always extracted in multiple different runs. However, biologically, there may be multiple different gene subsets which regulate cancer in different ways. As a result, the biological analysis on the single gene subset extracted by these algorithms may loss other cancer-related information, especially when some selection bias [3] is introduced in the gene selection process.

To extract multiple informative gene subsets for reliable cancer classification, the GSVM-RFE algorithm is proposed.

6.4.2 Relevance Index

"Relevance Index" (RI) was used to measure the relevance of a feature to a cluster in [112] to ease an unsupervised clustering process. Here the idea is extended as the first step of GSVM-RFE. The point here is to pre-filtering some non-relevant genes to ease the following gene selection and supervised classification. Because a gene is possible to be negatively expressed or positively expressed, Equations 6.7-6.8 define the negative relevance index and the positive relevance index to measure the negative correlation and the positive correlation of a gene with the cancer being studied, respectively.

$$R_{i-} = 1 - \sigma_{i-}^2 / \sigma_i^2, \tag{6.7}$$

$$R_{i+} = 1 - \sigma_{i+}^2 / \sigma_i^2, \tag{6.8}$$

where σ_i^2, σ_{i-}^2, and σ_{i+}^2 are the variances of the projected values on the i^{th} gene of the whole training samples, the negative training samples, and the positive training samples, respectively. A large negative relevance index value means the local variance among negative samples is small

compared to the global variance among the whole samples. Fig. 6.7 shows an example how the two relevance Index metric work.

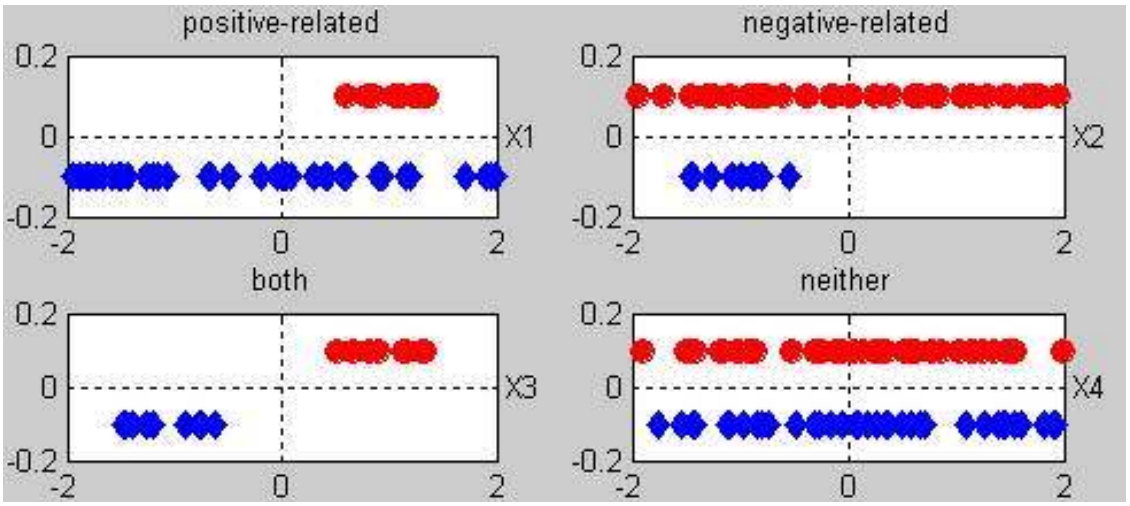


Figure. 6.7. gene X1 is positive-related; gene X2 is negative-related; gene X3 is both positive-related and negative-related; gene X4 is irrelevant

To apply RI metric for gene selection, a negative filtering threshold $\alpha_- \in [0,1)$ and a positive filtering threshold $\alpha_+ \in [0,1)$ need to be decided. The i^{th} gene is "negative-related" if $R_{i-} \geq \alpha_-$ because the projections of the negative samples are closer than the projections of the whole samples on the i^{th} gene. Similarly, it is "positive-related" if $R_{i+} \geq \alpha_+$. If $R_{i-} < \alpha_-$ and $R_{i+} < \alpha_+$, it is "non-related". A gene may be both negative-related and positive-related. These two filtering thresholds should be selected carefully: firstly, they can not be too large, otherwise the information loss may happen because some cancer-related genes are wrongly eliminated; secondly, they should be selected "in balance", which means negative-related genes and positive-related genes should be selected in balance, otherwise the minor genes are possible to be totally eliminated to result in performance degradation, especially when negative-related genes are significantly larger than positive-related genes in the original dataset or visa versa.

6.4.3 Fuzzy C-Means clustering

RI metric alone can not extract good gene subsets. The shortcoming of RI is that it assumes the independence between different genes. As we know, the assumption is not true for microarray gene expression data. If the filtering thresholds are too large, many informative genes will be wrongly eliminated.

Some genes may be similarly regulated and similarly expressed. And hence these genes may play a similar role in cancer classification. As a result, if genes with similar expression patterns are grouped together into clusters, a few typical genes in a cluster may be selected and other genes in the cluster may be safely eliminated without significant loss of information. On the other hand, an informative gene may contribute to cancer classification with complex correlations with multiple different clusters. Therefore, after the pre-filtering by RI metric, Fuzzy C-Means [12] is adopted to group genes into different function clusters.

The Fuzzy C-Means clustering algorithm groups genes into K clusters with centers $c_1, \cdots c_k, \cdots c_K$ in the training samples space. (That is, each training sample is a dimension of the space). Fuzzy C-Means assigns a real-valued vector $U_i = \{\mu_{1i}, \cdots \mu_{ki}, \cdots, \mu_{Ki}\}$ to each gene. $\mu_{ki} \in [0,1]$ is the membership value of the i^{th} gene in the k^{th} cluster. The larger membership value indicates the stronger association of the gene to the cluster. Membership vector values μ_{ki} and cluster centers c_k can be obtained by minimizing

$$J(K,m) = \sum_{k=1}^{K}\sum_{i=1}^{N}(\mu_{ki})^m d^2(x_i, c_k), \tag{6.9}$$

$$d^2(x_i, c_k) = (x_i - c_k)^T A_k (x_i - c_k), \tag{6.10}$$

$$\sum_{k=1}^{K} \mu_{ki} = 1, 0 < \sum_{i=1}^{N} \mu_{ki} < N, \tag{6.11}$$

where $1 \le i \le N$ and $1 \le k \le K$ [12].

In Eq. 6.9, K and N are the number of clusters and the number of genes in the dataset, respectively. $m > 1$ is a real-valued number which controls the 'fuzziness' of the resulting clusters, μ_{ki} is the degree of membership of the i^{th} gene in the k^{th} cluster, and $d^2(x_i, c_k)$ is the square of distance from i^{th} gene to the center of the k^{th} cluster. In Eq. 6.10, A_k is a symmetric and positive definite matrix. If A_k is the identity matrix, $d^2(x_i, c_k)$ corresponds to the square of the Euclidian distance. Eq. 6.11 indicates that empty clusters are not allowed.

6.4.4 GSVM-RFE algorithm

The new GSVM-RFE algorithm is proposed in this work for more reliable gene selection. It works in three stages. Fig. 6.8 shows a sketch of the GSVM-RFE algorithm.

At the first stage, RI metric is used to coarsely group genes into two granules: "relevant granule" and "irrelevant granule". The relevant granule consists of negative-related genes and positive-related genes, while the irrelevant granule is comprised of irrelevant genes (genes with small *RI+* values and small *RI-* values). Only genes in the relevant granule are survived for the following stages. The assumption is that irrelevant genes are not so useful for cancer classification or even possible to correlate other genes in some unknown complex ways to confuse Fuzzy C-Means to get good clustering results or confuse SVMs to get good classification results. This pre-filtering process can dramatically decrease the number of candidate genes on which Fuzzy C-Means works. Therefore, it can improve both the efficiency and the effectiveness of the following stages of GSVM-RFE.

At the second stage, in each step survived genes are clustered by Fuzzy C-Means into several "function granules". In each function granule, a linear SVM is modeled and genes in the function granule are ranked by their wi^2 value in Eq. 6.6 in the descending order.

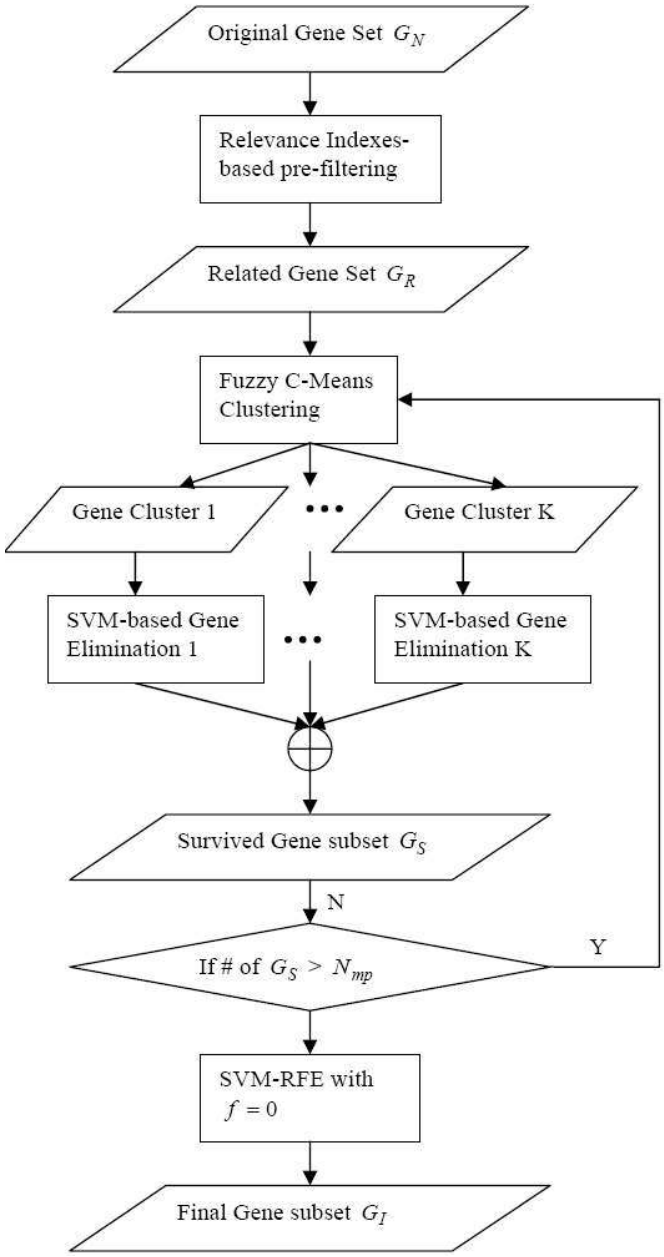


Figure. 6.8. GSVM-RFE algorithm

The higher-ranked genes are selected as new survived genes, and then all survived genes in these function granules are combined disjunctively into the next step. This process is repeated until the number of survived genes is less than or equal to a pre-specified threshold N_{mp}. By using the Fuzzy C-Means clustering algorithm, GSVM-RFE explicitly groups genes with similar expression patterns into clusters and then the lower-ranked genes in each cluster could be safely removed because the more significant genes with similar functions will survive. Furthermore, due to complex correlation between genes, the similarity is by no means a "crisp" concept. Fuzzy C-Means deals with complex correlation between genes by assigning a gene into several clusters with different membership values. Therefore, a really informative gene achieves more than one opportunities to survive.

At the third stage, SVM-RFE with $f=0$ is used to finely select the final "most informative" gene subsets.

The filter-out factor $f>0$ in the second stage. As a result, GSVM-RFE is a $O(d^2)$ algorithm because the clustering process dominates. In practice, because the pre-filtering stage by RI metric eliminates most of genes, the expected time is much faster.

Because different runs of Fuzzy C-means generate different clusters, different runs of GSVM-RFE should extract different gene subsets in general. This flexibility makes GSVM-RFE more suitable for gene selection than traditional methods. In a gene regulation network, many different gene subsets may regulate cancer in different ways. These multiple different gene subsets may be more helpful for cancer study by minimizing information loss. Moreover, the genes that survive in multiple subsets deserve higher priority for biological study.

6.5 GSVM-RFE simulation

6.5.1 Modeling

The same as [41], the original dataset is simply normalized so that each gene vector has 0 for mean and 1 for standard deviation. To avoid overfitting, for testing accuracy evaluation, the mean and standard deviation are calculated on the training dataset; while for leave-one-out validation accuracy evaluation on the training dataset, the validation sample is kept out from calculating these two values.

The regulation parameter $C \equiv 1$ for the linear SVMs. For the SVM-RFE algorithm, the filter-out factor $f = 0.5$ is used at the first stage to coarsely select a set of N_{mp} genes. At the second stage, $f = 0$ is used to finely select the final "most informative" gene subsets. The performances of the linear SVMs on the gene subsets between N_{fu} genes and N_{fl} genes are reported for comparison.

For the Fuzzy C-Means algorithm, the "fuzziness degree" $m = 1.15$, the maximal iteration number is 100, and the minimal improvement $\varepsilon = 10^{-5}$. For the GSVM-RFE algorithm, at the second stage, in each step survived genes are grouped into 5 clusters, in each of which one linear SVM is modeled to select genes with the filter-out factor $f = 0.5$, and then all of the 5 subsets of survived genes are combined disjunctively into the next step.

Notice in each step, the fuzzy membership values are defuzzified in such a way that a gene is always grouped into the cluster with the largest membership value and the cluster with the second largest membership value. The assumption is that different gene function groups are clustered based on their expression strengths. Some genes whose expression strengths are between two groups may be better to be clustered into the two groups at the same time. This way, each gene achieves two opportunities to survive at the following selection process.

This recursive process is repeated until the number of survived genes is less than or equal to N_{mp}. The third stage of GSVM-RFE is the same as the second stage of the above SVM-RFE algorithm.

In the following, a gene subset is referred to be a "perfect" gene subset if the SVM modeled in the gene subset space achieves 100% leave-one-out cross-validation accuracy on the training dataset and also 100% prediction accuracy on the testing dataset.

6.5.2 Data description on the prostate cancer dataset

The first set of experiments is on the prostate cancer dataset for tumor versus normal classification [58]. The training dataset consists of 102 prostate samples (52 with tumors and 50 without tumors); while the testing dataset has 34 samples (25 with tumors and 9 without tumors). The two datasets are prepared under different biological experimental contexts. There is a nearly 10-fold difference in overall microarray intensity between them [58]. The 12600 features correspond to some normalized gene expression values extracted from the microarray image.

Here negatives are defined to be the normal prostate samples without tumor, while positives are the tumor samples. The genes distribution in the prostate cancer dataset is highly imbalanced between negative-related genes and positive-related genes. If $\alpha_+ = \alpha_- = 0.5$, 4761 positive-related genes and only 110 negative-related genes are survived. To alleviate the imbalance, $\alpha_+ = 0.75$ and $\alpha_- = 0.5$ are used to select 721 positive-related genes and 110 negative-related genes. There is no overlapping between positive-related genes and negative-related genes. At the following stages, N_{mp}=64, N_{fu}=10, N_{fl}=1.

We run GSVM-RFE 20 times. For each run, 10 stratified gene subsets are extracted with 10, 9, …, 1 gene(s). The testing accuracies of the linear SVMs on the 10 gene subsets are recorded. The highest one is called "best accuracy" and the mean of them is called "average accuracy".

6.5.3 Statistical Analysis on the prostate cancer dataset

Table 6.16 shows that GSVM-RFE significantly improves accuracy compared with S2N and SVM-RFE. The best accuracy (the first column) of the "expected" GSVM-RFE averaged on the 20 runs is 99.71% and the average accuracy (the second column) is 90.18%.

TABLE 6.16
TESTING ACCURACY ON THE PROSTATE CANCER DATASET

model	Best (<=10)	Mean (<=10)	Std (<=10)
S2N	0.9118	0.7941	0.0832
SVM-RFE	0.9412	0.8177	0.0818
expected GSVM-RFE	0.9971	0.9018	0.0803

TABLE 6.17
TESTING AUC ON THE PROSTATE CANCER DATASET

model	Best (<=10)	Mean (<=10)	Std (<=10)
S2N	0.8333	0.6111	0.1571
SVM-RFE	0.9244	0.6804	0.1811
expected GSVM-RFE	0.9962	0.8394	0.1478

Table 6.17 demonstrates that GSVM-RFE has good average performance (83.94%) while S2N and SVM-RFE have poor average performances (61.11% and 68.04%) in terms of the AUC metric. In other words, GSVM-RFE is much better than S2N and SVM-RFE with higher AUC values.

TABLE 6.18
TESTING SENSITIVITY ON THE PROSTATE CANCER DATASET

model	Best (<=10)	Mean (<=10)	Std (<=10)
S2N	1.0000	1	0
SVM-RFE	1.0000	0.9720	0.0329
expected GSVM-RFE	1.0000	0.9720	0.0562

TABLE 6.19
TESTING SPECIFICITY ON THE PROSTATE CANCER DATASET

model	Best (<=10)	Mean (<=10)	Std (<=10)
S2N	0.8929	0.7872	0.0743
SVM-RFE	0.9600	0.8282	0.0988
expected GSVM-RFE	0.9980	0.9125	0.0833

Furthermore, Table 6.18 and Table 6.19 compare sensitivity and specificity among the three algorithms. As shown in the results, only GSVM-RFE demonstrates superior balance ability between the negative class and the positive class (tumor prostate) with both high average sensitivity (97.20%) and also high average specificity (91.25%). Firstly, Relevance Index-based pre-filtering selects positive-related genes and negative-related genes in balance. Secondly, FCM explicitly groups genes into different clusters based on their expression patterns so that informative genes from different function granules (clusters) are selected in balance.

Fig. 6.9 visualizes the average performance comparison among S2N, SVM-RFE and expected GSVM-RFE. The goal of such an average performance comparison is to verify that the performance gain of GSVM-RFE is statistically significant. However, in practice, biologists do not care about the average performance but the best gene subset(s). As a result, we will only submit the gene subsets extracted by the runs with good performance while discarding the gene subsets from the runs with bad performance.

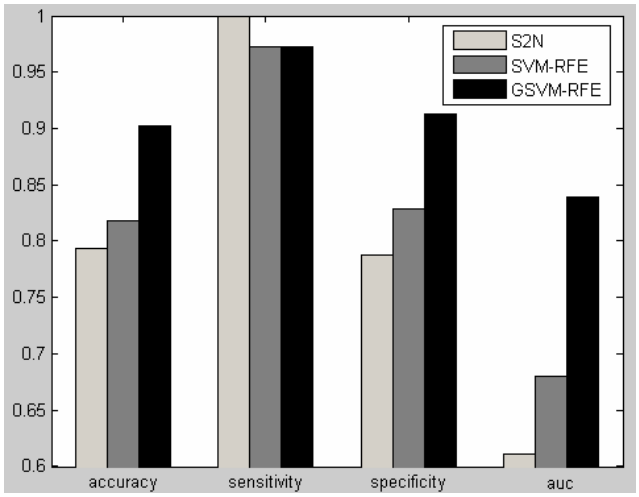


Figure 6.9. average performance comparison on the prostate cancer dataset

More importantly, a perfect gene subset is extracted in 18 runs of GSVM-RFE. To the best of the authors' knowledge, it is the first time that such a perfect gene subset is reported for the prostate cancer dataset. Interestingly, 18 runs of GSVM-RFE extract the exactly the same perfect gene subset with 8 genes. Although these 18 runs produce different clusters, the same 8 genes can always be extracted. It strongly convinces us the 8 genes are critical for prostate cancer. The other two perfect gene subsets with 15 and 17 genes, respectively, are also extracted by other runs of GSVM-RFE. Due to the length limit, we have to skip them here.

6.5.4 Biological Analysis on the prostate cancer dataset

TABLE 6.20
A PERFECT GENE SUBSET ON THE PROSTATE CANCER DATASET

rank/ index	GAN	Description of Gene Function	References
1/6185	X07732*	hepatoma mRNA for serine protease hepsin	[110]
2/4649	M16942		
3/5821	AF044311		
4/5045	AL080150		
5/10537	AF045229*		[110]
6/6368	AB017363		
7/11818	M21535	erg protein (ets-related gene)	
8/5402	W27944	39g8 retina	

Table 6.20 lists the perfect subset of 8 genes. The "*" signed genes have been already identified by other approaches in previous works, while the other ones are newly found by GSVM-RFE.

6.5.5 Statistical Analysis on the AML/ALL dataset

The AML/ALL leukemia dataset [58] mentioned above is also used in the experiments. Here negatives are defined to be the ALL samples, while positives are AML samples. At the first stage, $\alpha_+ = 0.5$ and $\alpha_- = 0.5$ are adopted to extract 1685 positive-related genes and 432 negative-related genes. Only the 2020th gene (GAN: M55150) is both negative-related and positive-related. N_{mp}=169, N_{fu}=64, N_{fl}=1. We run GSVM-RFE 20 times.

Table 6.21 shows that GSVM-RFE is most accurate compared to S2N and SVM-RFE. The best accuracy of the "expected" GSVM-RFE averaged on the 20 runs is 96.91% and the average accuracy is 91.75%.

Table 6.22 demonstrates that GSVM-RFE has excellent average performance (90.64%), S2N has good average performance (86.54%), while SVM-RFE has fair average performances (76.29%) in terms of the AUC metric. In other words, GSVM-RFE is better than S2N and SVM-RFE with higher AUC values.

TABLE 6.21
TESTING ACCURACY ON THE AML/ALL DATASET

model	Best (<=20)	Mean (<=20)	Std (<=20)
S2N	0.9412	0.8883	0.0537
SVM-RFE	0.9412	0.8029	0.0541
expected GSVM-RFE	0.9691	0.9175	0.0370

TABLE 6.22
TESTING AUC ON THE AML/ALL DATASET

model	Best (<=20)	Mean (<=20)	Std (<=20)
S2N	0.9286	0.8654	0.0632
SVM-RFE	0.9286	0.7629	0.0666
expected GSVM-RFE	0.9668	0.9064	0.0423

TABLE 6.23
TESTING SENSITIVITY ON THE AML/ALL DATASET

model	Best (<=20)	Mean (<=20)	Std (<=20)
S2N	1.0000	0.9950	0.0154
SVM-RFE	1.0000	0.9900	0.0205
expected GSVM-RFE	0.9975	0.9695	0.0330

TABLE 6.24
TESTING SPECIFICITY ON THE AML/ALL DATASET

model	Best (<=20)	Mean (<=20)	Std (<=20)
S2N	0.9091	0.8466	0.0554
SVM-RFE	0.9091	0.7575	0.0611
expected GSVM-RFE	0.9681	0.9017	0.0455

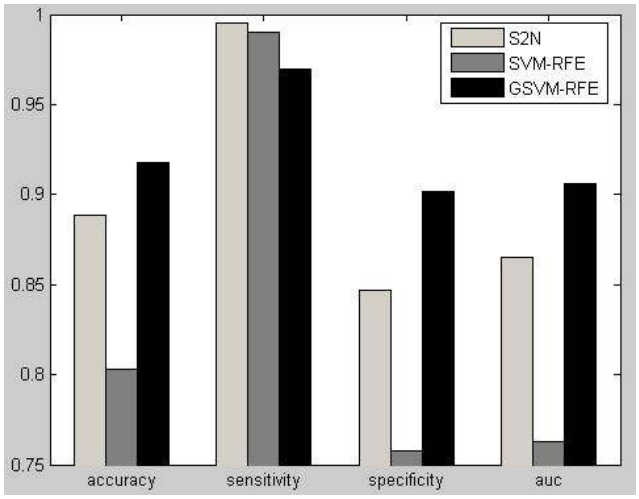


Figure 6.10. average performance comparison on the AML/ALL dataset

Furthermore, Table 6.23 and Table 6.24 compare sensitivity and specificity among the three algorithms. Only GSVM-RFE demonstrates excellent balance ability between the negative class and the positive class (AML) with both high average sensitivity (96.95%) and also high average specificity (90.17%). Once again, it proves that GSVM-RFE can extract positive-related genes and negative-related genes in balance.

Fig. 6.10 visualizes the average performance comparison among S2N, SVM-RFE and expected GSVM-RFE.

6.5.6 Biological Analysis on the AML/ALL dataset

TABLE 6.25
A PERFECT GENE SUBSET ON THE AML/ALL DATASET

rank/ index	GAN	Description of Gene Function	References
1/1834	M23197*	Human differentiation antigen (CD33)	[41][11][55][76][101]
2/6539	X85116*	Integral membrane protein (Protein 7.2b)	[41][11][55]
3/2020	M55150*	Human fumarylacetoacetate hydrolase	[41][11][17]
4/4535	X74262*	RETINOBLASTOMA BINDING PROTEIN P48	[41][11][17]
5/461	D49950*	Interferon-gamma inducing factor (IL-18)	[11][17][40][32]
6/4847	X95735*	Homo sapiens Zyxin	[41][37][11][17]
7/3221	U43885	Grb2-associated binder-1 mRNA	
8/5950	M29610*	Glycophorin E	[6]
9/6169	M13690*	C1NH Complement component 1 inhibitor	[55]

Table 6.25 lists a perfect subset with 9 genes to analyze their biological functions related to leukemia classification. Notice in Table 6.25, M55150, the only gene both negative-related and positive-related, is extracted and ranked at the third place. The "*" signed genes have been already identified to be cancer-related in previous works, while the other ones are newly identified by GSVM-RFE. The other two perfect gene subsets with 8 and 16 genes (not reported here), respectively, are also extracted by other runs of GSVM-RFE.

Many genes in these perfect gene subsets have already been reported to be directly or indirectly related to tumor activities. For instance, high-level CD33 (differentiation antigen) activity is observed in AML [76,101], IL-18 (Interferon-gamma inducing factor, a cytokine mainly produced by macrophages) affect T-cell activation [40,32], which can influence tumor development.

Besides, the novel genes discovered by this new GSVM-RFE algorithm may also be directly or indirectly related with cancer. Furthermore, GSVM-RFE extracts genes together as a group. For example, 8 from 9 genes in Table 6.25 have been reported to be cancer-related in different biological or bioinformatics literature. However, it is novel for GSVM-RFE to extract them together. In this way, the inherent regulation mechanism and correlations among these genes may be explored more effectively and more efficiently.

6.6 Discussion

6.6.1 Natural training/testing partition

One challenge of cancer classification on microarray gene expression data is that the data is generally not independent and identically distributed (i.i.d.) as traditional data mining models assume. Experiments under different experimental conditions (or even the same experimental conditions but at different time) produce different expression data. As a result, an ideal algorithm

should be able to model a classifier from the dataset under one experimental condition and generalize it well onto other datasets under different experimental conditions. To this end, a classifier should discriminate the samples not by the experimental condition related factors but by the cancer-related factors.

For the AML/ALL dataset, some previous works [43,37,33] combined the training dataset and the testing dataset together into one whole dataset on which leave-one-out validation heuristic was used for model evaluation. However, in each fold of leave-one-out validation, many training samples come from the same biological condition as the validation sample. As a result, it is easier to model a classifier to correctly classify the validation sample under the help of some factors that are experimental condition-related but possibly not cancer-related. That is, the leave-one-out validation is prone to model a classifier overfitting the biological experiment condition-related factors. And hence really informative gene subsets may not be extracted. As a result, this work still adopts the original training/testing partition. With this "natural" partition, a classifier is expected to be dominated by really cancer-related factors if both high validation accuracy on the training dataset and high prediction accuracy on the testing dataset are observed.

To justify this idea, we also try leave-one-out validation on the whole AML/ALL dataset to run GSVM-RFE for gene selection. The results (not reported here due to length limit) show that almost all runs of GSVM-RFE can extract a small subset with only 5 or 7 genes with 100% leave-one-out accuracy. However, based on these gene subsets, a SVM built on the training dataset performs not well on the testing dataset.

The similar overfitting happens in the experiments on the prostate cancer dataset. Some previous research work [110] adopted leave-one-out validation on the 102 training samples. However, our experiments show that a lot of gene subsets have 100% leave-one-out validation accuracy but

very low prediction accuracy on the 34 testing samples. Therefore, leave-one-out validation accuracy is not a convincible criterion for gene selection because of the strong negative effects of biological experiment condition-related factors.

6.6.2 Size of the final gene subsets

The size is decided according to practical utilities of the extracted gene subsets: A small number of genes are desirable for further biological study because it is very expensive or even impractical to conduct biological experiments on a large number of genes; On the other hand, the prediction is not accurate if too few genes are selected. In previous research works, the size is usually decided arbitrarily by a biologist. However, we notice that it is difficult for a biologist to decide such a value precisely. It is more reasonable for him or her to decide a field instead of a value. In this work, the lower-bound of the field is always assumed to be 1. So the biologist only needs to decide the upper-bound. For the prostate cancer dataset, performance on the 10 stratified subsets with 10 genes to 1 gene is reported. For the AML/ALL dataset, performance on the 20 stratified subsets with 20 genes to 1 gene is reported.

6.6.3 RI pre-filtering

Another group of experiments show that performance of GSVM-RFE is deteriorated without RI pre-filtering (not shown here). This result verifies our assumption that RI pre-filtering not only can speed up the running of GSVM-RFE, but also can decrease noise between irrelevant genes and noisy genes. As a result, the following stages of GSVM-RFE work more effectively and more efficiently to extract informative cancer-related gene subsets.

6.6.4 Number of clusters and membership of clusters

For complex gene selection problems, selection from multiple granules is better than selection from one single granule without clustering. By explicitly clustering genes into different function

granules based on their expression strength patterns, redundant genes can be identified and removed. Furthermore, genes that regulate cancers in different ways can be extracted in balance. It should be even better if a gene can be grouped into more than one granule. By being clustered into multiple function granules, an important cancer-related gene may be ranked low in some granules but ranked high in other granules, and hence gains more opportunities to be extracted. FCM has the inherent advantage to be applied for gene selection compared to other crisp clustering algorithms such as self-organizing map, k-means, or hierarchical clustering.

On the other hand, too many clusters make clustering trivial. In the second stage of GSVM-RFE, 5 clusters are generated at each step. That is, genes are approximately grouped into "strong negative-related cluster", "weak negative-related cluster", "neutral cluster", "weak positive-related cluster", and "strong positive-related cluster". Notice that at the second stage, the clustering is recursively executed on the smaller and smaller survived gene subset at each step. The number 5 may be not the best. However, the contribution here is to prove multiple granules can improve the performance by decreasing noise and selecting genes in balance.

6.6.5 Extract gene subsets in balance

TABLE 6.26
UNBIASED PERFORMANCE COMPARISON ON THE PROSTATE CANCER DATASET

models	accuracy	auc	sensitivity	specificity
No selection	0.3235	0.5400	1.0000	0.0800
S2N	0.7941	0.6111	1.0000	0.7872
SVM-RFE	0.8177	0.6804	0.9720	0.8282
GSVM-RFE	0.9018	0.8394	0.9720	0.9125

TABLE 6.27
UNBIASED PERFORMANCE COMPARISON ON THE AML/ALL DATASET

models	accuracy	auc	sensitivity	specificity
No selection	0.9118	0.8929	1.0000	0.8696
S2N	0.8883	0.8654	0.9950	0.8466
SVM-RFE	0.8029	0.7629	0.9900	0.7575
GSVM-RFE	0.9175	0.9064	0.9695	0.9017

Without gene selection, classification performance of linear SVMs is not reliable, as shown in the first row of Table 6.26 for the prostate cancer dataset and the first row of Table 6.27 for the AML/ALL dataset. Obviously, the classifiers are prone to the negative class with low AUC values (or high sensitivity and low specificity).

S2N and SVM-RFE do not alleviate the imbalance too much. Linear SVMs modeled on the gene subsets extracted by both of them are still prone to the negative class (the second rows and the third rows in the Tables, the average performance is reported).

Due to the explicit granulation with clustering, the minor genes may form a "pure granule" (a cluster) so that at least some of them can be extracted. The balance ability is demonstrated to be critical for the superior performance of GSVM-RFE (the fourth rows in the Tables, the average performance is reported).

The results also show that the prostate cancer dataset is much more noisy and imbalanced than the AML/ALL dataset. The fact that the performance improvement gained by GSVM-RFE is more significant in the first dataset demonstrates that GSVM-RFE improves the cancer classification mainly due to noise elimination and balanced gene selection.

6.6.6 Selection bias

Ambroise et al demonstrated that selection bias is introduced because the testing dataset or the validation dataset is involved in gene selection [3]. As a result, the testing or validation accuracy can not be confidently generalized to new samples. In our experiments, the best accuracy is in this case. However, in practice we can expect that the size of a gene subset is randomly selected in the field $[N_{fl}, N_{fu}]$ and one special run of the Fuzzy C-Mean clustering for GSVM-RFE is randomly picked up. In this way, the testing dataset is leaved out from gene selection and classifier modeling. As a result, the average accuracy is an unbiased estimation of generalized

prediction accuracy. Actually, the average accuracy is even a pessimistic estimation because a biologist does not expect to select too few genes with too much information loss. Consequently, the improvement of GSVM-RFE is reliable. The unbiased average testing accuracy on the two datasets are reported in the first columns of Table XII and Table XIII. Notice that in Table XIII, the unbiased accuracy of S2N or SVM-RFE is even lower than the accuracy without gene selection so that the extracted gene subsets are not reliable.

Furthermore, due to existence of selection bias, it is helpful or even necessary to provide multiple gene subsets other than just one single "most informative" gene subset to avoid information loss. Therefore, the flexibility of clustering makes GSVM-RFE more suitable for gene selection than traditional algorithms.

6.6.7 Time Complexity

Correlation-based algorithms are straightforward to understand and work efficiently. If there are d genes originally, the ranking process takes $O(dlgd)$ time.

Because the ranking process dominates the SVM-RFE algorithm, the SVM-RFE (which remove 50% lower-ranked genes in each step) works in $O(dlgd)$ time.

GSVM-RFE is a $O(d^2)$ algorithm because the clustering process dominates. In practice, because the pre-filtering stage by RI metrics eliminates most of genes, the expected time is much shorter.

6.7 Summary on GSVM-RFE simulation

To find most informative gene subsets for reliable cancer classification, the GSVM-RFE algorithm is proposed in this chapter. Firstly, GSVM-RFE utilizes Relevance Index metric for gene pre-filtering to improve the algorithm efficiency and effectiveness at the same time. Secondly, GSVM-RFE explicitly groups genes with similar expression patterns into clusters. Therefore, the lower-ranked genes in each cluster can be safely removed because the more

significant genes with similar functions will survive. Finally, GSVM-RFE deals with complex correlation between genes by assigning a gene into several clusters with different membership values so that a really informative gene is more possible to survive.

GSVM-RFE is more reliable to predict unseen testing samples, as demonstrated in the experiments on the two microarray gene expression datasets. More importantly, GSVM-RFE can find multiple compact cancer-related gene subsets on each of which a SVM with 100% prediction accuracy can be modeled. Due to the selection bias in each single gene subset, the multiple "perfect gene subsets" are believed to be more helpful for biologists to uncover the inherent cancer-resulting mechanism.

In summary, GSVM-RFE has advantages over traditional algorithms. Of course, the newly identified genes by this algorithm need to be further confirmed biologically, which may generate more insights for cancer mechanism, treatment and study. Nevertheless, the extraction of these cancer-related gene subsets may help to stimulate and guide detailed cancer studies on the gene functions.

As a general feature selection algorithm, because of the inherent advantage to eliminate irrelevant or redundant features while select really informative features, we expect that this superior performance can also be true in processing other similar datasets with extreme sparseness such as biomedical text mining, biomedical image pattern recognition, and other bioinformatics problems. This is an interesting future work.

CHAPTER 7

GSVM-DC

7.1 Algorithm

Usually, grid search heuristic [46] is adopted for SVM modeling. The basic idea is to try different parameter grids to find which one is the best. It is time-consuming due to large parameter space, especially for very large datasets.

To improve efficiency, it is natural to try to decrease the size of the training dataset. The elimination of some samples from the training dataset may have two results: 1 information loss by eliminating informative or useful samples to deteriorate the performance of a classifier; and 2 data cleaning by eliminating the irrelevant or redundant or noisy samples to improve the performance of a classifier. Our goal is to minimize the first negative effect (information loss) while to maximize the second positive effect (data cleaning). If the original training dataset is viewed as a single granule, the problem is, how to adjust the size of the granule to achieve the optimal classification performance.

For a biomedical problem, usually much more than need data is accumulated. There are many redundant or even noisy data. In this case, too many training data is unnecessary or even harmful. The SVM algorithm tells us that only SVs are needed and other samples can be safely removed without affecting classification (Fig. 7.1). The advantage motivates us to explore the potentiality to utilize SVM as a data cleaning tool.

However, different kernels and/or different parameter values may extract different SVs. Which one should be extracted?

SVM tries to find the optimal decision boundary by trade-off between the margin width and the training accuracy. There is a regulation parameter C for misclassification errors penalty. For

complex datasets, if *C* is increased, the SVM algorithm is forced to decrease misclassification errors by decreasing the margin width. As a result, less SVs are extracted. It motivates us to build a linear SVM with a very small *C* value and the extracted SVs can be used as the new compressed training dataset, which is expected to include most, if not all, important and useful samples.

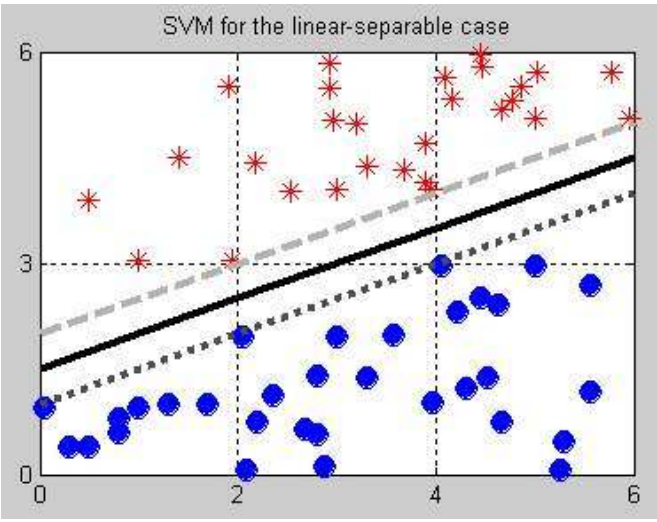


Figure. 7.1. A SVM with maximal margin. Except Support Vectors, the other samples can be safely removed

However, information loss may still happen because a linear SVM is still possible to loss some important samples. As a result, it is not a good idea to extract SVs directly from the whole training dataset. The other disadvantage by extracting the "global" SVs is overfitting: when we use grid search and cross validation heuristics to optimize the parameters, overfitting may happen because the validation samples were already used to extract the "global" SVs.

A natural way is to split the training samples into different granules. The samples in one granule could be leaved for validation and other samples could be used to extract SVs. In [36], authors won the ACM KDDCUP04 protein homology task mainly by taking advantage of the natural

granules splitting (blocks decided by native protein sequences) to undersample the training dataset with SVMs in each block. However, it is still unexplored how to do undersampling (SVs extraction) for a general dataset without any prior knowledge for granulation.

Our solution is to build multiple granules in a bagging-similar way. Fig. 7.2 sketches the algorithm, named Granular Support Vector Machines with Data Cleaning (GSVM-DC). Each granule is composed of the training samples in each fold of the *k*-fold cross validation. There are altogether k granules, where k is the number of the folds of cross validation.

The granulation is highly overlapping (each training sample appears in *k-1* granules) so that each training sample gets *k-1* opportunities to be extracted as a SV. In each granule, a SVM with a very small *C* value is modeled and the corresponding SVs are extracted to form the "Local Support Vector set" (LSVs) to "clean" the granule. With the very small C value and highly overlapping granulation, it is expected to minimize information loss caused by missing some important samples.

After LSVs are extracted, *k*-fold cross validation is executed to optimize SVM parameters. In the i^{th} fold, the training dataset is LSV_i, the validation dataset is the $(k\text{-}i+1)^{th}$ subset of the original training dataset.

Finally, LSVs are disjunctively combined to form a compressed training dataset, on which a SVM with the optimal parameters is modeled for classification.

Suppose there are $g \in N$ groups of parameters for grid search with *k*-fold cross validation. In each fold, there are *n* training samples, and *m* LSVs, averaged on all *k* folds. If SVM modeling takes $O(g * k * n^2)$ time, GSVM-DC needs $O(k * n^2 + g * k * m^2)$. Because usually the size of LSVs is much less than the size of the original training dataset ($m << n$) and $g >> 1$, the tuning by grid search can be greatly speed up.

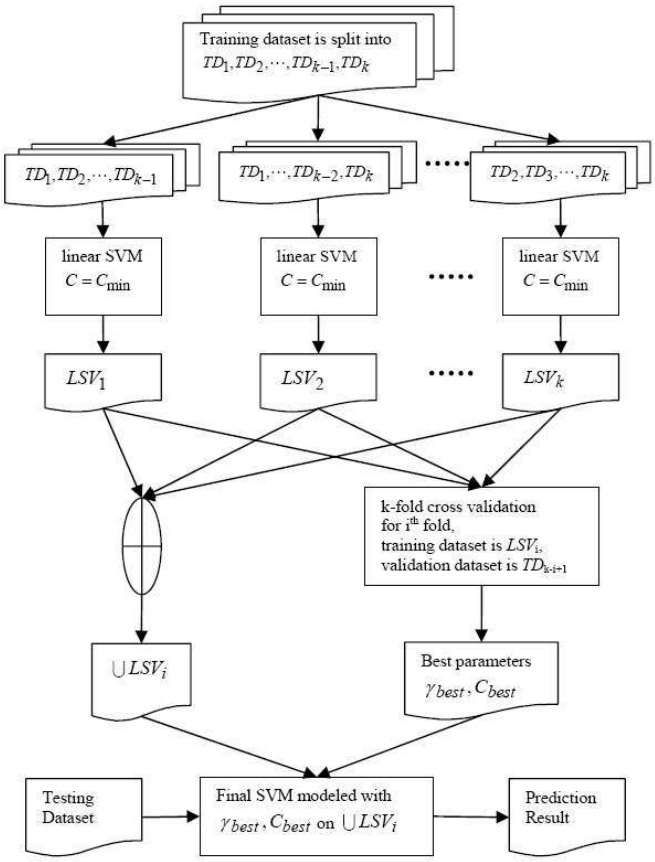


Figure. 7.2. GSVM-DC algorithm

7.2 Simulation

GSVM-DC proposed in this chapter is compared with the original SVM algorithm. We try linear kernel and RBF kernel for SVM. Correspondingly, the final SVM in GSVM-DC also adopts linear kernel and RBF kernel, respectively, for comparison. Notice that the SVMs used to select LSVs in GSVM-DC are always linear SVMs.

7.2.1 Datasets

Six life science binary classification data from UCI data mining repository [68] are used for comparison. The detailed characteristics of datasets are listed in Table 7.1.

TABLE 7.1
CHARACTERISTICS OF DATASETS USED FOR EXPERIMENTS

Dataset	Size	Attr	Ratio
1 Pima Indians Diabetes	768	8	500:268
1 Wisconsin Breast Cancer	683	9	444: 239
1 Cleveland heart-disease	297	13	160:137
1 Postoperative Patient	90	8	66:24
2 Abalone	4177	8	4145:32
2 Protein Localization Sites	1484	8	1433:51

Note 1: Size = # of cases after removing cases with missing data, Attr = # of input features, Ratio = # of negative cases : # of positive cases.
Note 2: 16 cases in Wisconsin Breast Cancer dataset and 6 cases in Cleveland heart-disease dataset with missing values were removed.
Note 3: Class "S" in Postoperative Patient dataset was defined as positive. Class "19" in Abalone dataset was defined as positive. Class "ME2" in Protein Localization Sites dataset was defined as positive.

7.2.2 Metrics

Two sets of experiments, on balanced datasets and imbalanced datasets, separately, are designed and performed.

For balanced datasets, misclassification error, defined in Eq. 1.1, is used to evaluate performance of the two algorithms. The top 4 datasets in Table 7.1 are used in the first set of experiments.

$$g-means=\sqrt{\frac{TN}{TN+FP}\times\frac{TP}{TP+FN}}. \tag{7.1}$$

For imbalanced datasets, misclassification error is virtually useless to evaluate a classifier's performance. [56] proposed g-means as defined in Eq. 7.1. This metric has been broadly used by many researchers to evaluate performance of classifiers on imbalanced datasets. We also adopt g-means here. The last 2 datasets in Table 7.1 are used in the second set of experiments.

7.2.3 Modeling

Each dataset is split into 5 equal size subsets. Each subset is leaved for testing in turn and other 4 subsets are combined and used for training. The split is executed in a stratified way so that

$$S(training):S(testing)=4:1,$$

$$S(positive\_training):S(positive\_testing)=4:1,$$

$$S(negative\_training) : S(negative\_testing) = 4:1,$$

where $S(x)$ means the size of the dataset x.

For each training/testing process, firstly the data is normalized so that each input feature has 0 mean and 1 standard deviation on the training dataset; then two models are built: the first model is a general SVM, whose parameters (γ, C) are optimized by grid search and 4-fold inner cross validation. (For linear kernel, only C is optimized.) The second model is a GSVM-DC: Local Support Vectors are extracted by a linear SVM with $C = 2^{-10}$ in each granule that consists of the training samples in each fold. Because there are 4 granules, 4 sets of LSVs are extracted. And then the parameters are optimized in the similar way to SVM but on the smaller training dataset that is only composed of the LSVs. Finally performance on the 5 training/testing processes is aggregated (for misclassification errors) or averaged (for g-means values) for comparison.

The above mentioned 5-fold outer cross validation is executed 5 times for 5 different random splitting on each dataset.

For linear kernel, the grid search scope is limited in

$$C \in \{2^{-10}, 2^{-9}, 2^{-8}, 2^{-7}, 2^{-6}, 2^{-5}, 2^{-4}, 2^{-3}, 2^{-2}, 2^{-1}, 2^{0}, 2^{1}, 2^{2}, 2^{3}, 2^{4}, 2^{5}\}.$$

For RBF kernel, the grid search scope is limited in

$$\gamma \in \{2^{-16}, 2^{-14}, 2^{-12}, 2^{-10}, 2^{-8}, 2^{-6}, 2^{-4}, 2^{-2}, 2^{0}, 2^{2}, 2^{4}\},$$

$$C \in \{2^{-6}, 2^{-4}, 2^{-2}, 2^{0}, 2^{2}, 2^{4}, 2^{6}, 2^{8}, 2^{10}\}.$$

7.2.4 Results analysis on balanced datasets

Tables 7.2-7.4 show the results on the first set of experiments. For linear kernel, performance of the two algorithms is almost the same. However, as we analyzed above, GSVM-DC is much

faster than SVM. That means GSVM-DC works both efficiently and effectively. Due to the small size of these datasets, the time differences are not significant so we don't report them here.

For RBF kernel, GSVM-DC is even a little more accurate than SVM with fewer errors. The results demonstrate that GSVM-DC can effectively clean the training dataset and avoid information loss at the same time, at least on the 4 datasets. For example, Table 7.2 shows that GSVM-DC (average test errors=175.2 with RBF kernel) is more accurate than SVM (average test errors=178.6 with RBF kernel) on Pima Indians Diabetes dataset.

TABLE 7.2
VALIDATION/TEST ERRORS ON PIMA INDIANS DIABETES DATASET

Trial	linear SVM	linear GSVM	RBF SVM	RBF GSVM
1	173/176	173/172	170/178	170/175
2	175/178	174/176	172/177	170/172
3	176/177	176/176	172/182	174/180
4	173/176	172/176	169/175	170/170
5	176/178	176/177	173/181	173/179
Mean	**174.6/177**	**174.2/175.4**	**171.2/178.6**	**171.4/175.2**
Std	**1.52/1**	**1.79/1.95**	**1.64/2.88**	**1.95/4.32**

TABLE 7.3
VALIDATION/TEST ERRORS ON WISCONSIN BREAST CANCER DATASET

Trial	linear SVM	linear GSVM	RBF SVM	RBF GSVM
1	22/23	22/23	21/26	21/24
2	20/20	20/20	18/21	18/22
3	21/21	21/21	20/24	20/21
4	21/22	21/22	18/28	18/28
5	20/22	20/22	20/23	20/21
Mean	**20.8/21.6**	**20.8/21.6**	**19.4/24.4**	**19.4/23.2**
Std	**0.84/1.14**	**0.84/1.14**	**1.34/2.70**	**1.34/2.95**

TABLE 7.4
VALIDATION/TEST ERRORS ON CLEVELAND HEART-DISEASE DATASET

Trial	linear SVM	linear GSVM	RBF SVM	RBF GSVM
1	48/50	48/50	45/49	45/49
2	46/45	46/45	46/45	46/45
3	45/47	45/47	45/46	45/46
4	47/47	47/47	46/49	46/48
5	46/45	46/45	44/45	44/45
Mean	**46.4/46.8**	**46.4/46.8**	**45.2/46.8**	**45.2/46.6**
Std	**1.14/2.05**	**1.14/2.05**	**0.84/2.05**	**0.84/1.82**

TABLE 7.5
VALIDATION/TEST ERRORS ON POSTOPERATIVE PATIENT DATASET

Trial	linear SVM	linear GSVM	RBF SVM	RBF GSVM
1	25/24	25/24	24/27	25/26
2	25/24	25/24	24/25	25/24
3	25/24	25/24	25/24	25/24
4	25/24	25/24	25/26	24/24
5	25/24	25/24	25/25	25/24
Mean	**25/24**	**25/24**	**24.6/25.4**	**24.8/24.4**
Std	**0/0**	**0/0**	**0.55/1.14**	**0.45/0.89**

It seems that GSVM-DC is more possible to improve performance on RBF kernel than on linear kernel. One possible reason is that RBF kernel is more complex than linear kernel. Therefore, SVM with RBF kernel is more sensitive to redundant or noisy samples. Modeling only on LSVs eliminates the negative effect and hence improves performance of GSVM-DC.

7.2.5 Results analysis on imbalanced datasets

Tables 7.6 and 7.8 compare performance of two algorithms on the second group of datasets. Linear SVM and linear GSVM-DC are totally ineffective because both of them have 0 g-means values. The reason is that they classify every sample as negative so that TP is 0.

For RBF kernel, GSVM-DC greatly improves performance on the imbalanced datasets. For Abalone dataset, the g-means value of GSVM-DC is 0.6030, averaged on 5 trials, which is much higher than SVM whose average g-means value is 0.0804. For Protein Localization Sites dataset, GSVM-DC (with g-means value=0.6138, average on 5 trials) is also significantly better than SVM (with average g-means=0.5393). Interestingly, SVM has higher validation g-means value but lower test g-means value than GSVM-DC. Once again, it shows that modeling only on LSVs eliminates the negative effect of redundant or noisy samples and hence improves classification performance.

TABLE 7.6
VALIDATION/TEST G-MEANS ON ABALONE DATASET

Trial	linear SVM (%)	linear GSVM (%)	RBF SVM (%)	RBF GSVM (%)
1	0/0	21.59/0	20.87/16.43	61.54/64.06
2	0/0	19.63/0	9.86/8.14	62.53/59.94
3	0/0	23.87/0	15.64/8.13	62.78/58.01
4	0/0	31.87/0	16.29/7.50	61.13/59.48
5	0/0	21.63/0	10.02/0	62.84/60.02
Mean	**0/0**	**23.72/0**	**14.54/8.04**	**62.16/60.30**
Std	**0/0**	**4.80/0**	**4.65/5.82**	**0.78/2.25**

TABLE 7.7
TIME(S)/AVGTRAINSIZE G-MEANS ON ABALONE DATASET

Trial	RBF SVM	RBF GSVM
1	1909/3342	51/69
2	1902/3342	50/70
3	2005/3342	55/68
4	2012/3342	53/70
5	1985/3342	54/65
Mean	**1963/3342**	**52.6/68.4**
Std	**53.12/0**	**2.07/2.07**

TABLE 7.8
VALIDATION/TEST G-MEANS ON PROTEIN LOCALIZATION SITES DATASET

Trial	linear SVM (%)	linear GSVM (%)	RBF SVM (%)	RBF GSVM (%)
1	0/0	35.30/0	58.77/53.29	62.19/59.04
2	1.58/0	25.38/0	61.33/56.56	59.24/59.23
3	0/0	23.63/0	59.37/54.65	58.04/60.11
4	0/0	22.72/0	65.22/49.03	61.26/62.72
5	2.13/0	17.73/0	61.77/56.10	59.33/65.82
Mean	**0.74/0**	**24.95/0**	**61.29/53.93**	**60.01/61.38**
Std	**1.03/0**	**6.45/0**	**2.54/3.02**	**1.68/2.88**

TABLE 7.9
TIME(S)/AVGTRAINSIZE G-MEANS ON PROTEIN LOCALIZATION SITES DATASET

Trial	RBF SVM	RBF GSVM
1	430/1187	27/91
2	478/1187	30/91
3	472/1187	29/92
4	458/1187	27/90
5	457/1187	29/91
Mean	**459/1187**	**28.4/91**
Std	**18.55/0**	**1.34/0.71**

Moreover, Tables 7.7 and 7.9 show that GSVM-DC with RBF kernel is also much faster than SVM with RBF kernel. For Abalone dataset, GSVM-DC averagely runs 52.6 seconds, while SVM needs 1963 seconds. The reason is that GSVM-DC greatly decreases the size of the training dataset from 3342 to 68.4, averagely. For Protein Localization Sites dataset, GSVM-DC averagely takes 28.4 seconds with the average training size 91, while SVM takes 459 seconds with the training size 1187.

7.3 Discussion

In this chapter, a new GSVM modeling algorithm, named GSVM-DC, is presented. It works by building a sequence of information granules and then extracting informative samples as Local Support Vectors while eliminating redundant samples in each granule. Finally, the LSVs are disjunctively combined to model a final SVM. In this way, the local significance of each granule and global correlation among different granules are elegantly trade-off. As a result, an accurate and fast classifier can be modeled.

GSVM-DC is inherently an undersampling technology. The improvement on effectiveness for imbalanced datasets is expected to be more significant if we combine GSVM-DC with some oversampling technologies, such as SMOTE [24]. Algorithm design and simulations on larger and more complex datasets are currently in processing.

CHAPTER 8

GSVM-RU

Highly imbalanced classification is important and increasingly common with emergence of new machine intelligence application domains including biomedical informatics. In order to solve this challenging class imbalance problem, a novel Granular Support Vector Machines - Repetitive Undersampling algorithm (GSVM-RU) is designed in this work. GSVM-RU creatively utilizes Support Vector Machines (SVM) themselves for undersampling to minimize the negative effect of information loss while maximizing the positive effect of data cleaning in the undersampling process. Consequently, an accurate and fast classifier can be modeled. The empirical study on four benchmark imbalanced datasets demonstrates that GSVM-RU is both effective and efficient. Specifically, for the extremely imbalanced abalone dataset, GSVM-RU achieves 73.4±1.6% g-means value, which is much higher than the best known result 57.8±5.4%. Another encouraging result is that GSVM-RU leads the extremely imbalanced KDDCUP 2004 protein homology prediction competition as of July/19/2005.

8.1 Introduction

8.1.1 Class Imbalance Problem

Highly skewed data distribution induces the class imbalance problem that happens, in its simplest form, when there are significantly more negative samples than positive samples for a binary classification problem. (In this chapter, the majority class is always assumed to be negative and the minority class is positive.) The class imbalance problem is ubiquitous in machine intelligence applications, such as protein homology prediction, diagnosing medical diseases, credit card fraud detection, intrusion detection for national security, etc. Usually, for an imbalanced classification problem, it is of primary interest to find rare samples.

8.1.2 Traditional Algorithms

Many methods have been proposed for imbalanced classification and some good results have been reported [25]. These methods can be categorized into three different classes: weight-adjusting, boundary alignment, and sampling. Sampling can be further categorized into two subclasses: oversampling the minority class, or undersampling the majority class. Interested readers may refer to [106] for a good survey. However, the class imbalance problem is still not thoroughly investigated yet:

- Most of current methods use Decision Trees (DT) as the basic classifier [48]. Although there are already some works on SVM for imbalanced classification [1,85,107], the application of SVM for undersampling is still unexplored. Because SVM decides the class label of a sample only based on Support Vectors (SVs), which are the training samples close to the decision boundary, the modeling effectiveness and efficiency may be improved for imbalanced classification in a SVs-based undersampling way.
- From the viewpoint of granular computing, most of current methods, including cost-sensitive boosting, bagging, undersampling the majority class, or oversampling the minority class, actually can be viewed as granular-computing-based because all of them try to form multiple information granules with suitable numbers and suitable sizes (See Section 2 for a brief introduction of granular computing). It is interesting to try to solve the class imbalance problem systematically in the framework of granular computing.

8.1.3 SVM for Imbalanced Classification

SVM embodies the Structural Risk Minimization (SRM) principle to minimize an upper bound on the expected risk [102,19]. Because structural risk is a reasonable trade-off between the training error and the modeling complication, SVM has a great generalization capability.

Geometrically, the SVM modeling algorithm works by constructing a separating hyperplane with the maximal margin.

Compared with other standard classifiers, SVM performs better on moderately imbalanced datasets. The reason is that only SVs are used for classification and many negative samples which are far from the decision boundary can be removed without affecting classification [1]. However, performance of SVM is significantly deteriorated on the highly imbalanced datasets. For this kind of datasets, the SRM principle becomes unsuitable because it is prone to find the simplest model that best fits the training dataset. Unfortunately, the simplest model is exactly the naive classifier that classifies all samples as negative.

Another disadvantage of SVM is that it is an expensive $O(n^2)$ algorithm. Usually, the grid search heuristic [46] is used to find optimal SVM parameters. For large datasets, it is very time-consuming, especially when some non-linear kernel (for example, Radial Basis Function kernel) is applied.

There are already some works that are targeted at improving effectiveness of SVM for highly imbalanced classification:

Akbani et al proposed the SMOTE with Different Costs algorithm (SDC) [1]. SDC oversamples the minority class by applying Synthetic Minority Over-sampling TEchnique (SMOTE) [24], a popular oversampling algorithm, with different error costs. As a result, the boundary of the learned SVM can be better defined and far away from the minority class.

Raskutti et al explored effects of different imbalanced compensation techniques on SVM [85]. They demonstrated that a one-class SVM learned only from the minority class sometimes can perform better than a SVM modeled from two classes.

Wu et al proposed the Kernel Boundary Alignment algorithm (KBA) which adjusts the boundary toward the majority class by directly modifying the kernel matrix [107]. In this way, the boundary is expected to be closer to the "ideal" boundary.

8.1.4 GSVM-RU for Imbalanced Classification

In this chapter, we creatively utilize the advantage of SVs-based classification to design a novel Granular Support Vector Machines–Repetitive Undersampling algorithm (GSVM-RU) under the principle of granular computing.

As above-mentioned, only SVs of a SVM classifier are related to classification. So the intuitive idea is to extract SVs as the new training dataset while eliminating the non-SVs samples for undersampling. However, information loss may happen by extracting SVs only once. Firstly, whether a sample is extracted to be a SV is sensitive to SVM parameters. Secondly, the highly skewed data distribution pushes the boundary toward the minority class [107]. As a result, some informative samples may be lost.

GSVM-RU extracts SVs multiple times to build multiple information granules, and then the information in these granules is fused in a data-dependant way for classification. Our theoretical and empirical studies below show that GSVM-RU can achieve better data distribution with much fewer training samples than the original dataset. Consequently classification performance can be improved in terms of both effectiveness and efficiency.

The rest of the chapter is organized as follows. In Section 8.2, GSVM-RU is presented in detail. Section 8.3 evaluates performance of GSVM-RU on three highly imbalanced life science datasets. Section 8.4 reports performance of GSVM-RU on the KDDCUP04 protein homology prediction task. Finally, Section 8.5 concludes the chapter.

8.2 GSVM-RU algorithm

Under the framework of GSVM, GSVM-RU is designed to attack highly imbalanced classification problems.

8.2.1 GSVM-RU

Usually, the grid search heuristic [46] is adopted for SVM modeling: different parameter grids are tried to find which one has the best training (or validation) performance. It is time-consuming due to usually large datasets and the large parameter space.

To improve efficiency, it is natural to decrease the size of the training dataset. In this sense, undersampling is by nature more suitable to model a SVM for imbalanced classification than other approaches.

However, elimination of some samples from the training dataset may have two effects:

- information loss: due to elimination of informative or useful samples, classification performance is deteriorated;
- data cleaning: because of elimination of irrelevant or redundant or noisy samples, classification performance is improved.

Our goal is to minimize the negative effect of information loss and to maximize the positive effect of data cleaning.

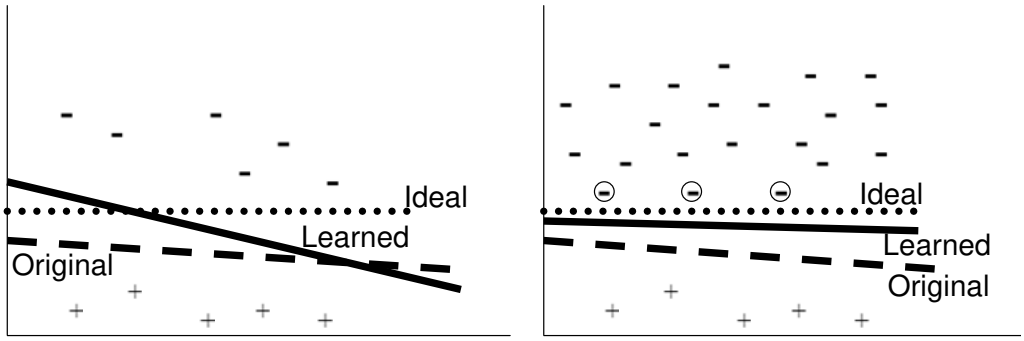
For a highly imbalanced dataset, it is expected that there are many redundant or even noisy negative samples. Random undersampling is the most common undersampling approach for rebalancing the dataset to achieve better data distribution. However, random undersampling suffers from information loss. As Fig. 8.1(a) shows, although random undersampling makes the distance of the learned boundary close to the distance of the ideal boundary, the cues about the orientation of the ideal boundary may be lost [1].

In this work, we creatively utilize SVM itself for undersampling. The idea is based on the well-known fact about SVM: only SVs are necessary and other samples can be safely removed without affecting classification. This fact motivates us to explore the potentiality to utilize SVM for data cleaning/undersampling.

Unfortunately, due to the highly skewed data distribution, the SVM modeled on the original training dataset is prone to classify every sample to be negative. As a result, a single SVM cannot guarantee to extract all informative samples as SVs, no matter which kernel and which parameters are used.

We believe that GSVM is a promising way to reduce information loss. The assumption is that although a single SVM is not enough to extract all informative samples, it does be able to extract a part of them. Under this assumption, multiple information granules with different informative samples can be formed by the following granulation operations: Firstly, we assume that all positive samples are informative to form a positive information granule. Secondly, negative samples extracted by a SVM as SVs are also possibly informative so that they form a negative information granule. Here we call these negative samples as Negative Local Support Vectors (NLSVs). And then the NLSVs are removed from the original training dataset to produce a smaller training dataset, on which a new SVM is modeled to extract another group of NLSVs. This process is repeated several times to form multiple negative information granules.

After that, an aggregation operation is executed to selectively aggregate the samples in these negative information granules with all positive samples to complete the undersampling process. Finally, a SVM is modeled on the aggregated dataset for classification. Fig. 8.2 sketches the GSVM-RU algorithm.



(a) randomly undersampling (b) GSVM-RU undersampling

the dot line - the ideal boundary
the dash line - the boundary learned from the original dataset
the solid line - the boundary learned from the undersampling dataset

Figure. 8.1. GSVM-RU can still give good cues on the orientation of the ideal boundary while make the distance close to the ideal one.

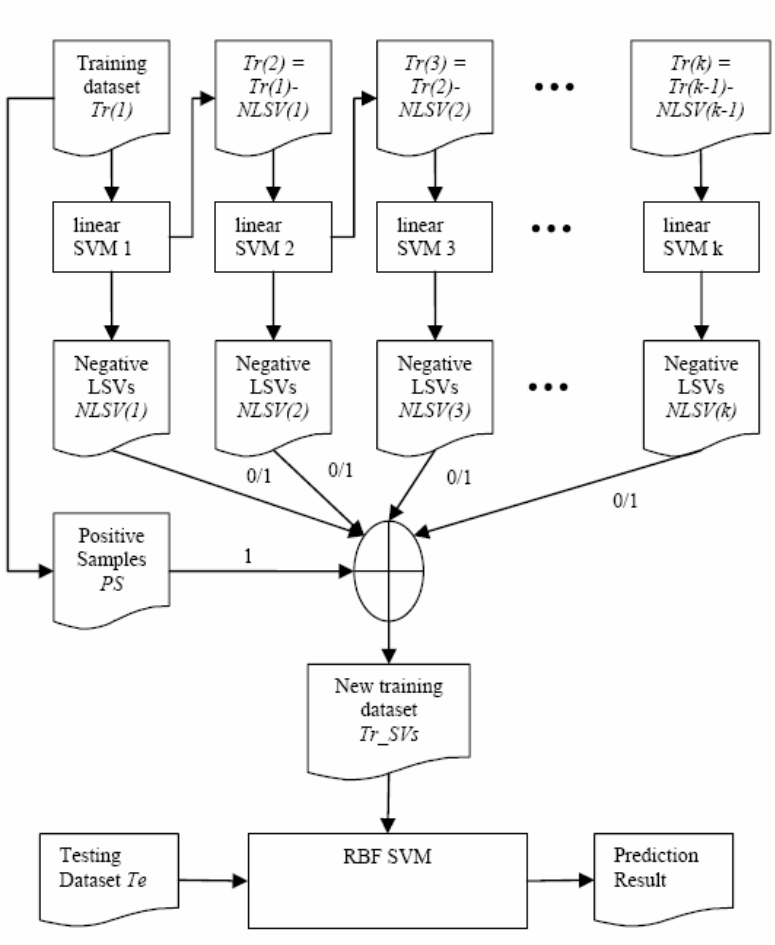


Figure. 8.2. GSVM-RU algorithm

As demonstrated in Fig. 8.1(b), because only the circled negative samples are removed from the original dataset, GSVM-RU undersampling can still give good cues about the orientation of the ideal boundary, and hence can overcome the shortcoming of random undersampling as mentioned earlier.

To make GSVM-RU an utilizable algorithm, there are still three problems: (1) how to select kernels and parameters to extract NLSVs? (2) how many negative information granules should be formed? and (3) how to aggregate the samples in these information granules?

In this work, linear SVMs are adopted for undersampling. The obvious advantage by using linear SVMs is the efficiency reason: for a linear SVM, only the regulation parameter C needs to be tuned so that the parameter space for grid search is much smaller. (As a comparison, two parameters (γ, C) need to be tuned for a SVM with the RBF kernel). As a result, a linear SVM is easier and faster to be modeled than a non-linear kernel SVM. Furthermore, although a single linear SVM may lose information, running it multiple times based on the above-mentioned granulation idea can extract most, if not all, informative samples. (Actually, the RBF SVM is also used for undersampling in our simulations. However, no obvious performance gain is observed.)

As we know, SVM tries to find the optimal decision boundary by trade-off between the margin width and the training accuracy. The regulation parameter C is used for misclassification errors penalty. Different C values result in different SVs. So we can adjust C to control the information extracted for one granule. The optimal C value is data-dependant and can be searched by cross validation.

For the similar reason to reduce information loss, more negative information granules are preferred than less negative information granules. However, some non-informative samples may

also be extracted as NLSVs. As a result, some granules which are not really informative should be eliminated from the final aggregated dataset. In GSVM-RU, the granulation operation and the aggregation operation are executed iteratively. The undersampling process is stopped if classification performance cannot be further improved when a new negative granule is extracted and the corresponding NLSVs are aggregated into the final dataset.

In general, if $g \in N$ granules are extracted, there are $2g$ possible aggregation ways. For simplicity and efficiency, two special aggregation operations are adopted in this work:

- The first aggregation operation is called "discard": when a new negative granule is extracted, only negative samples in this granule are aggregated into the new aggregation dataset and all samples in old negative granules are discarded. This operation is based on an assumption mentioned in [1,107]: for a highly imbalanced classification problem, the majority class pushes the "ideal" decision boundary toward the minority class. So if old NLSVs are discarded, the decision boundary is expected to be closer to the ideal one. The repetitive undersampling process is stopped when the new extracted granule alone cannot further improve classification performance.
- The second aggregation operation is called "combine": when a new granule is extracted, it is combined with all old granules to form a new aggregation dataset. The assumption is that not all informative samples can be extracted as NLSVs in one granule. As a result, it is expected to reduce information loss by extracting NLSVs multiple times. The repetitive undersampling process is stopped when the new extracted granule cannot further improve classification performance if joint with old granules.

8.2.2 Time Complexity Analysis

In grid search, suppose there are $d_1 \in N$ groups of parameters on the SVM for undersampling and $d_2 \in N$ groups of parameters on the SVM for classification; Suppose also there are $n \in N$ training samples in the original training dataset and $m \in N$ informative samples in the aggregated dataset after g times undersampling. If SVM modeling needs $O(d_2 * n^2)$ time, GSVM-RU approximately takes $O(d_1 * g * n^2 + d_2 * m^2)$. If the number of informative samples is much less than the size of the original training dataset $(m << n)$ and the parameter space for GSVM-RU is smaller than the parameter space for the classification SVM $(d_1 * g < d_2)$ (which is true here because the linear SVM is used for undersampling and the RBF SVM is used for classification), the modeling time can be reduced compared with directly modeling a SVM on the original training dataset for classification.

8.3 Simulations on the First Group of Datasets

The hardware used in the simulations is a laptop with centrino-1.6MHz CPU and 1024M memory. The software is based on OSU SVM Classifier Matlab Toolbox [67], which implements a Matlab interface to LIBSVM [23].

8.3.1 Evaluation Metric and Datasets

For highly imbalanced datasets, accuracy is virtually useless to evaluate a classifier's performance. Kubat et al [56] proposed g-means as defined in Eq. 1 and Fig. 3, which is the geometric mean of classification accuracy on negative samples and classification accuracy on positive samples. This metric has been broadly used by many researchers to evaluate classification performance on imbalanced datasets. We also adopt g-means here. The three datasets in Table I are used in our simulations.

$$g-means=\sqrt{\frac{TN}{TN+FP}\times\frac{TP}{TP+FN}}\ . \tag{8.1}$$

8.3.2 Data Modeling

We run GSVM-RU on each dataset for 7 times.

For the yeast dataset and the abalone dataset, in each run, the original dataset is randomly split into 7 equal-sized subsets in a stratified way so that the majority/minority ratio is the same for each subset.

After that, each subset is left for testing in turn and other 6 subsets are combined for training. For each training/testing process, firstly the data is normalized so that each input feature has 0 mean and 1 standard deviation on the training dataset; then GSVM-RU is executed on the normalized training dataset: the model parameters are optimized by grid search with 6-fold inner cross validation. (Notice that SVMs used for undersampling are with the linear kernel, and SVMs used for classification are with the RBF kernel.) Finally each sample is used for testing one and only one time, and thus the testing performance is calculated and reported. The validation performance averaged on the 7 training/testing processes is also reported.

For the mammography dataset, the original dataset is randomly split into 10 equal-sized subsets in the stratified way and 9-fold inner cross validation is used to optimize the model parameters in each training/testing process.

The GSVM-RU undersampling process is repeated until classification performance cannot be improved by aggregating the latest NLSVs.

As mention in Section 2, to some extent, the regulation parameter C of a linear SVM can be adjusted to control how many informative samples are extracted to form an information granule. In our preliminary simulations, we validate this idea: for the yeast dataset and the abalone

dataset, more SVs can be extracted with the increase of the *C* value (Fig. 8.4 and Fig. 8.5); while for the mammography dataset, less SVs are extracted with the increase of the C value (Fig. 8.6).

8.3.3 Result Analysis

For the yeast dataset, the best validation performance is observed when the "discard" aggregation operation is adopted and the 7th granule is used as the final aggregation dataset. The result indicates that the first assumption (the decision boundary is pushed toward the minority class) is reasonable on the yeast dataset. When the NLSVs in the old granules are discarded, the decision boundary gradually goes back to the "ideal" one and thus classification performance is improved (Fig. 8.7). After the 8th granule is extracted, too many informative samples are discarded so that classification performance is deteriorated. And hence the repetitive undersampling process is stopped.

For the abalone dataset, the best validation performance is observed when the "combine" aggregation operation is adopted and the first 5 granules are combined to form the final aggregation dataset. The result indicates that the second assumption (a part but not all of informative samples can be extracted in one granule) is reasonable on the abalone dataset. When more and more informative samples are combined into the aggregated dataset, information loss is less and less so that better performance can be achieved (Fig. 8.8). However, when the 6th granule is extracted and combined into the aggregation dataset, the validation performance can not be improved. The reason is that the new extracted samples are too far from the "ideal" boundary so that they are prone to be redundant or irrelevant other than informative. And hence the repetitive undersampling process is stopped.

For the mammography dataset, the best validation performance is observed when the first granule is used as the final aggregation dataset. Both the "discard" operation and the "combine"

operation are not effective to improve classification performance further (the results are not shown here). One possible reason is that enough informative samples have been extracted in the first granule. Another possible reason is that the aggregation operations used here are not general enough. It is an interesting future work to try more general aggregation operations. For example, maybe classification performance can be improved by discarding the 2nd granule and combining the 1st one and the 3rd one.

	real negatives	real positives
predicted negatives	TN	FN
predicted positives	FP	TP

Figure. 8.3. the confusion matrix

TABLE 8.1
CHARACTERISTICS OF DATASETS USED FOR SIMULATIONS

Dataset	Attr	Size	#positive (positive%)
Yeast 0	8	1484	51 (3.44%)
Abalone 0	8	4177	32 (0.77%)
Mammography	6	11183	260 (2.32%)
Error! Reference source not found.			

Note 1: Attr = # of input features, Size = # of cases.
Note 2: Class "ME2" in the yeast dataset is defined as

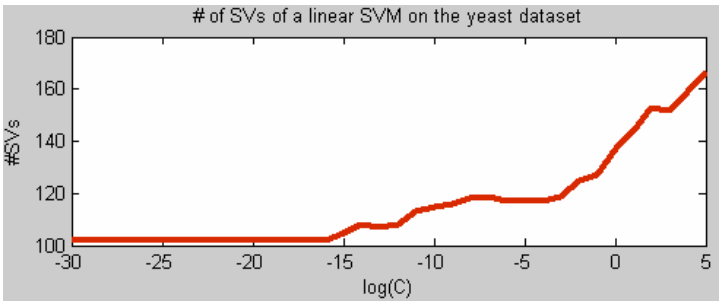


Figure. 8.4. a larger C value results in more Support Vectors on the yeast dataset

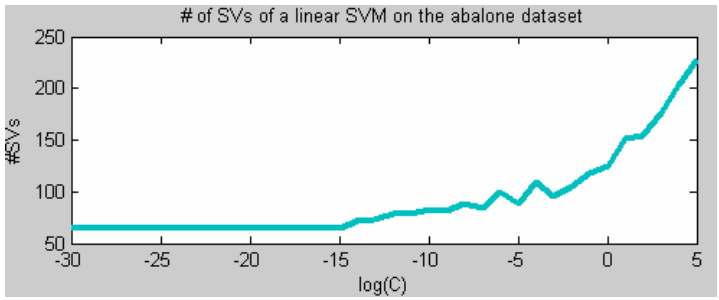


Figure. 8.5. a larger C value results in more Support Vectors on the abalone dataset

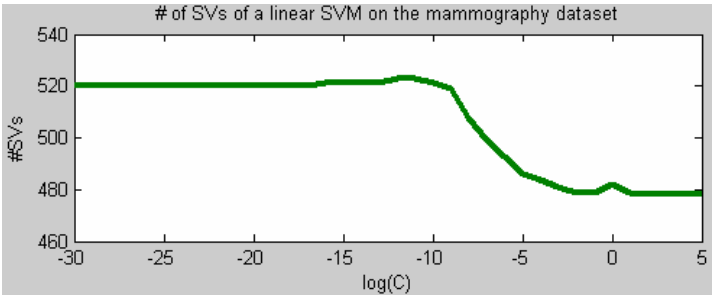


Figure. 8.6. a larger C value results in less Support Vectors on the mammography dataset

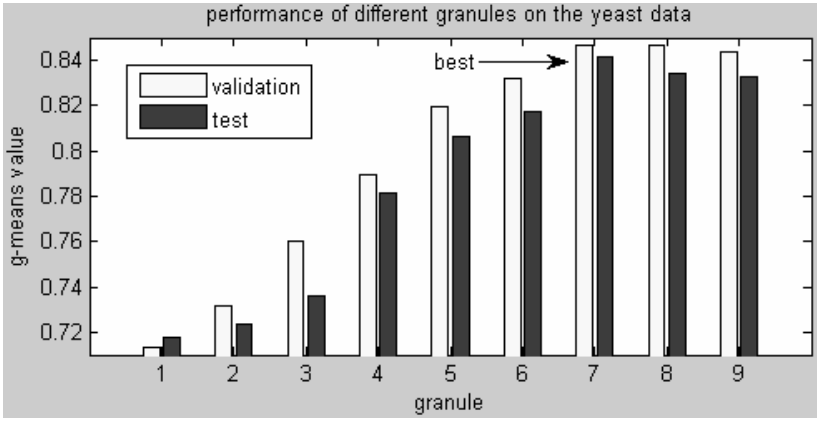


Figure. 8.7. results of different granules for the yeast dataset with the "discard" operation averaged on the 7 runs

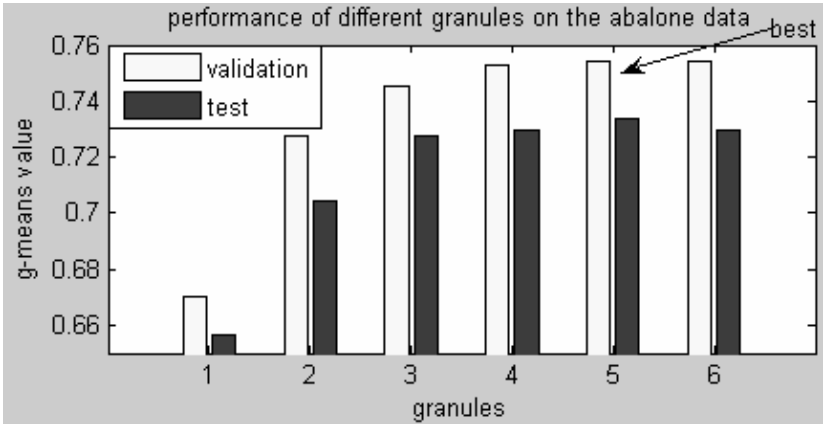


Figure. 8.8. results of different granules for the abalone dataset with the "combine" operation averaged on the 7 runs

TABLE 8.2
VALIDATION/TEST G-MEANS ON YEAST DATASET (THE 7TH GRANULE, 7-6-FOLDS DOUBLE CV)

Trial	Validation	Test
1	84.37	83.94
2	84.28	83.11
3	84.77	84.93
4	84.20	84.04
5	85.18	84.01
6	84.43	84.04
7	85.48	85.06
GSVM-RU	84.7±0.5	**84.2±0.7**
KBA 0	N/A	82.2±7.1

TABLE 8.3
VALIDATION/TEST G-MEANS ON ABALONE DATASET (THE FIRST 5 GRANULES, 7-6-FOLDS DOUBLE CV)

Trial	Validation	Test
1	76.04	75.01
2	75.76	73.55
3	74.97	70.15
4	75.45	74.81
5	74.40	73.23
6	74.74	73.57
7	76.57	73.50
GSVM-RU	75.4±0.8	**73.4±1.6**
KBA 0	N/A	57.8±5.4

TABLE 8.4
VALIDATION/TEST G-MEANS ON MAMMOGRAPHY DATASET (THE FIRST GRANULE, 10-9-FOLDS DOUBLE CV)

Trial	Validation (%)	Test (%)
1	83.16	83.83
2	83.93	83.63
3	83.20	83.63
4	82.27	83.33
5	84.05	84.75
6	83.91	84.50
7	83.98	83.42
GSVM-RU	83.5±0.7	**83.9±0.6**

Table 8.2 and Table 8.3 compare GSVM-RU with KBA [107] in terms of the g-means metric on the yeast dataset and the abalone dataset, respectively. The g-means value is increased from 82.2±7.1% to 84.2±0.7% for the highly imbalanced yeast dataset, and significantly increased from 57.8±5.4% to 73.4±1.6% for the extremely highly imbalanced abalone dataset. Notice that in Wu's work [107], for each random splitting, only one subset is used for testing and other six subsets are combined for training; in our work, every subset is used for testing in turn. So the performance improvement is statistically more reliable.

To our best knowledge, no results on the g-means metric have been reported for the mammography dataset in the literature. Table 8.4 reports performance of GSVM-RU and the result (83.9±0.6%) is quite promising.

Moreover, Table 8.5 compares the modeling time between GSVM-RU and a RBF SVM optimized by grid search. For the yeast dataset, GSVM-RU averagely takes 37 seconds, while SVM takes 643 seconds. For the abalone dataset, GSVM-RU averagely runs 447 seconds, while SVM needs 2748 seconds. The higher speed of GSVM-RU is due to the significantly smaller training size by undersampling the majority class. As a comparison, KBA even needs more time than SVM [107].

8.4 Simulations on the KDDCUP 2004 Protein Homology Prediction Dataset

The same hardware and software as the first group of simulations are adopted.

8.4.1 Granular Computing and GSVM Dataset and Evaluation Metrics

The KDDCUP 2004 protein homology prediction task at http://kodiak.cs.cornell.edu/kddcup/index.html is used in the second group of simulations. The detailed characteristics of the dataset are listed in Table 8.6. The task

can be modeled as a binary classification problem: Given a protein sequence, the task is to predict whether it is homologous to the corresponding native sequence or not. There are 153 native sequences in the training dataset and 150 native sequences in the testing dataset. For each native sequence, there is a block of approximately 1000 protein sequences with class label (1 means homologous and 0 means non-homologous). The class labels of protein sequences in the testing dataset are kept secret. 74 features are provided to describe the match (e.g. the score of a sequence alignment) between the native protein sequence and the sequence that is tested for homology. The problem is extremely highly imbalanced: there are only 1296 homologous protein sequences from altogether 145751 ones in the training dataset.

Four metrics are used for performance measures:

- TOP1: fraction of blocks with a homologous sequence ranked top 1 (maximize)
- RMSE: root mean squared error averaged on blocks (minimize)
- RKL: average rank of the lowest ranked homologous sequence (minimize)
- APR: average of the average precision in each block. For a single block, APR could be approximately described as the area of precision-recall curves. (maximize)

RMSE is a metric for accuracy evaluation and is easier to show the differences between models than directly using accuracy. The other 3 metrics are rank-based, which means that the 3 metrics' values are decided by the order of ranking list, and the absolute values of predictions do not affect the performances. The four metrics are precisely defined in perf [20]. In the simulation, we use the corresponding code to calculate the four metrics.

8.4.2 Data Modeling

Firstly the data is normalized so that each input feature has 0 mean and 1 standard deviation in each block. The reason for normalizing in each different block separately is that protein sequences in different blocks are in different protein families, which may be remote so that similar absolute feature vectors are not necessary to mean similar homology behaviors.

Similar to [36], GSVM-RU is tuned by grid search with 153-folds cross validation: each block is left for validation in turn and other 152 blocks are used for training. In each training/validation process, linear SVMs are modeled on each training block for undersampling and then all extracted informative samples are aggregated to model a RBF SVM for prediction.

GSVM-RU is tuned for each of the four metrics separately. Finally, the model parameters with the best validation performance are used to retrain a GSVM on all of the 153 blocks. The GSVM is used to make prediction on the testing dataset.

The GSVM-RU undersampling process is repeated until the validation performance cannot be improved by aggregating the latest NLSVs. Due to the efficiency issue, the regulation parameter C of linear SVMs for undersampling is fixed to be 1, and only the "discard" operation is adopted for aggregation.

TABLE 8.5
MODELING TIME COMPARISON AVERAGED ON 7 RUNS BETWEEN SVM AND GSVM-RU

	RBF-SVM (seconds)	GSVM-RU (seconds)
Yeast	643±26	37±3
Abalone	2748±74	447±18

TABLE 8.6
CHARACTERISTICS OF KDDCUP04 PROTEIN HOMOLOGY PREDICTION DATASETS

Dataset	Block	Size	Attr	#positive (positive%)
Training	153	145751	74	1296 (0.89%)
Testing	150	139658	74	N/A

Note: Block = # of blocks, Size = # of protein sequences, Attr = # of input features.

TABLE 8.7
VALIDATION/TEST PERFORMANCE ON KDDCUP04 PROTEIN HOMOLOGY PREDICTION TASK (153-FOLDS CV) AS OF 07/19/2005

	TOP1 (maximize)	RMSE (minimize)	RKL (minimize)	APR (maximize)	average
validation	0.9020	0.03553	40.54	0.84723	N/A
test	0.9000	0.03529	45.88	0.84184	N/A
rank	6.5	2	2	2	**3.125 (the best)**

8.4.3 Result Analysis

Our solution has the best average rank over the four metrics in the ongoing KDDCUP 2004 protein homology prediction contest as of 07/19/2005 at http://kodiak.cs.cornell.edu/cgi-bin/newtable.pl?prob=bio. Table 8.7 summarizes the result.

For the TOP1 metric, the best validation performance 0.9020 is achieved when the 1st granule is used as the final aggregation dataset. The testing performance is 0.9000, which ranks 6.5th.

For the RMSE metric, the best validation performance 0.03553 is achieved when the 1st granule is used as the final aggregation dataset. The testing performance is 0.03529, which ranks 2nd.

For the RKL metric, the best validation performance 40.54 is achieved when the 7th granule is used as the final aggregation dataset. The testing performance is 45.88, which ranks 2nd.

For the APR metric, the best validation performance 0.84723 is achieved when the 10th granule is used as the final aggregation dataset. The testing performance is 0.84184, which ranks 2nd.

The results demonstrate that the first assumption (the decision boundary is pushed toward the minority class) is reasonable for RKL and APR metrics. When the NLSVs in the old granules are discarded, the decision boundary gradually goes back to the "ideal" one and thus prediction performance is improved.

For TOP1 and RMSE metrics, the "discard" operation is not effective to improve prediction performance further. One possible reason is that enough informative samples have been extracted from the first granule. Another possible reason is that the "discard" operation is not the most suitable aggregation operation. We have not tried other more expensive operations due to the huge size of the KDDCUP 2004 protein homology prediction dataset.

8.5 Summary

A new learning model called Granular Support Vector Machines is proposed in this work. GSVM systematically and formally combines the methodologies from statistical learning theory and granular computing theory. In this work, a new GSVM modeling algorithm, named GSVM-RU, is designed specifically for highly imbalanced classification problems. GSVM-RU builds a sequence of information granules by repetitively extracting informative samples as SVs. Finally, the samples in these granules are selectively aggregated to model a SVM for final classification. In this way, the local significance of each granule and global correlation among different granules are elegantly trade-off. GSVM-RU is efficient because of usually much smaller size of the after-

undersampling aggregation dataset compared to the original training dataset. It is also effective due to

- reservation of informative samples which are essential for classification and
- elimination of large quantities of redundant or even noisy samples which may confuse a classifier to find the optimal decision boundary.

GSVM-RU is inherently an undersampling algorithm. The improvement on effectiveness seems more significant if the imbalance degree is higher. As two benchmarks, GSVM-RU greatly improves the g-means value on extremely imbalanced abalone dataset (the positive ratio is 0.77%) and leads the ongoing KDDCUP 2004 protein homology prediction contest (the positive ratio is 0.89%).

The performance is expected to be improved further if we combine GSVM-RU with some oversampling approaches, such as SMOTE [24], or boosting meta-learning techniques, such as SMOTEBoost [26]. Algorithm design and simulations on larger and more complex datasets are currently in processing. In the future, parallel GSVM-RU will also be investigated to speed up learning significantly.

CHAPTER 9

CONCLUSIONS AND FUTURE WORKS

9.1 Conclusion

A classification problem is a predictive data mining problem where the unknown variable is categorical. Samples of different classes are accumulated, on which a classifier is trained to predict future samples. With emergence of E-business, Web intelligence and biomedical informatics, new challenges are coming. Among them, noise, non i.i.d., high dimensionality and imbalance are four especially interesting ones due to their pervasiveness in datasets from these application domains.

In this work, a framework named Granular Support Vector Machines (GSVM) is proposed to systematically and formally combine statistical learning theory, granular computing theory and soft computing theory to enhance effectiveness, efficiency and/or interpretability of classification on complex datasets [93-100]. In general, GSVM works by building a sequence of information granules and then modeling Support Vector Machines (SVM) in some of these information granules when necessary. A good granulation method to find suitable granules is crucial for modeling a GSVM with good performance.

Under this framework, many algorithms have been proposed to build a GSVM model for classification problems with different characteristics. Specifically, GSVM-RFE (recursive feature elimination) algorithm was proposed for high-dimensional cancer classification. The empirical study demonstrates that GSVM-RFE can make much more reliable prediction on microarray expression data compared to previous approaches. Another GSVM-RU (repetitive undersampling) algorithm was proposed for highly

imbalanced classification. GSVM-RU ranks as one of the best solutions in ACM KDDCUP04 competition for protein homology prediction and ranks #1 in the US in DMC05 competition for customers' online shopping behavior prediction.

9.2 Long vision

A lot of data mining algorithms have been proposed by scientists from different research communities such as database, machine learning, statistics, soft computing, etc. Each algorithm has its own advantages and also its own disadvantages. For a specific data mining task, which algorithm is the best is highly data dependant. That means, before we touch the data, we can not know which algorithm is the most suitable for the problem at hand. As a result, it is desirable to design a hybrid and adaptive data mining system.

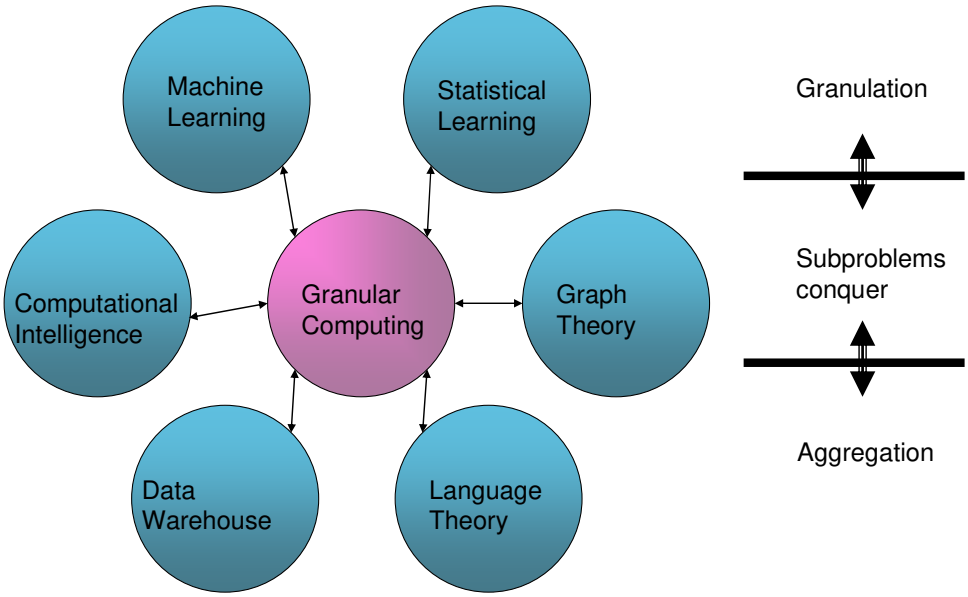


Figure. 9.1. GrC-PDM

Our long term research goal is to build a hybrid intelligent predictive data modeling framework with the ideas of granular computing (named GrC-PDM). With GrC-PDM

framework, we can build adaptive knowledge discovery and data mining systems to provide effective and efficient decision support for drug design, disease diagnosis, credit card fraudulent detection, spam filtering, and many other applications. This dissertation work can be viewed as one preliminary step toward the goal.

BIBLIOGRAPHY

[1] R. Akbani, S. Kwek, and N. Japkowicz, "Applying Support Vector Machines to Imbalanced Datasets," Proceedings of the 2004 European Conference on Machine Learning (ECML2004), pp.39-50, 2004.

[2] U. Alon, N. Barkai,D. A. Notterman, K. Gish, S. Ybarra, D. Mack, and A. J. Levine, "Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays," Proc. of Natl Acad Sci USA 96: 6745–6750, 1999.

[3] C. Ambroise and G.J. McLachlan, "Selection bias in gene extraction on the basis of microarray gene expression data," Proc. Natl. Acad. Sci. USA, 99:6562-6566, 2002.

[4] J. Archambault, G. Pan, G. K. Dahmus, M. Cartier, N. Marshall, S. Zhang, M. E. Dahmus, J. Greenblatt, "FCP1, the RAP74-interacting subunit of a human protein phosphatase that dephosphorylates the carboxyl-terminal domain of RNA polymerase IIO," J Biol Chem. 273:27593-27601, 1998.

[5] N. Arshadi and I. Jurisica, "Data Mining for Case-Based Reasoning in High-Dimensional Biological Domains," IEEE Transactions on Knowledge and Data Engineering, pp.1127-1137, Vol.17, No.8, 2005.

[6] N. D. Avent, "Human erythrocyte antigen expression: its molecular bases," Br J Biomed Sci. 54:16-37, 1997.

[7] E. Bair and R. Tibshirani, "Machine learning methods applied to DNA microarray data can improve the diagnosis of cancer," SIGKDD Explorations, vol. 5(2), pp. 48-55, 2003.

[8] A. Bargiela, Granular Computing: An Introduction, Kluwer Academic Pub, Kluwer, 2002.

[9] D. M. Barrett, K. S. Gustafson, J. Wang, S. Z. Wang, G. D. Ginder, "A GATA factor mediates cell type-restricted induction of HLA-E gene transcription by gamma interferon," Mol Cell Biol, 24(14):6194-204, 2004.

[10] K. P. Bennett and J. Blue, "A support vector machine approach to decision trees," R.P.I math report no. 97-100, Rensselaer Polytechnic Institute, Troy, NY, 1997.

[11] A. Ben-Dor, L. Bruhn, N. Friedman, I. Nachman, M. Schummer, and Z. Yakhini, "Tissue Classification with Gene Expression Profiles," Journal of Computational Biology, vol. 7, pp. 559-583, 2000.

[12] J. C. Bezdek, Pattern Recognition with Fuzzy Objective Function Algorithms. Plenum Press, New York, 1981.

[13] G. Bortolan and W. Pedrycz, "Reconstruction problem and information granularity," IEEE Transactions on Fuzzy Systems, 2, 1997, 234 - 248.

[14] A. P. Bradley, "The use of the area under the roc curve in the evaluation of machine learning algorithms," Pattern Recognition, 30(7):1145–1159, 1997.

[15] U. Braga-Neto and E. R. Dougherty, "Is cross-validation valid for small-sample microarray classification?," Bioinformatics, 2004, 20 : pp. 374-380.

[16] L. Breiman, J. C. Friedman, R. A. Olshen, and C. J. Stone, Classification and Regression Trees, Belmont, Calif.: Wadsworth, 1984.

[17] P. Broberg, "Statistical methods for ranking differentially expressed genes," Genome Biology 4: R41, 2003.

[18] L. J. Bruce, S. Ghosh, M. J. King, D. M. Layton, W. J. Mawby, G. W. Stewart, P. A. Oldenborg, J. Delaunay, M. J. Tanner, "Absence of CD47 in protein 4.2-deficient hereditary spherocytosis in man: an interaction between the Rh complex and the band 3 complex," Blood. 100:1878-85, 2002.

[19] C. J. C. Burges, "A tutorial on support vector machines for pattern recognition," Data Mining and Knowledge Discovery, 2(2), pp. 121-167, 1998.

[20] R.Caruana, The PERF Performance Evaluation Code, http://kodiak.cs.cornell.edu/kddcup/software.html.

[21] N. Cercone, A. An and C. Chan, "Rule Induction and Case-based Reasoning: Hybrid Architectures appear Advantageous," IEEE Transactions on Knowledge and Data Engineering, pp.166-174, Vol.11, No.1, 1999.

[22] F. Cetani, E. Pardi, P. Viacava, G. D. Pollina, G. Fanelli, A. Picone, S. Borsari, E. Gazzerro, P. Miccoli, P. Berti, A. Pinchera, C. A. Marcocci, "reappraisal of the Rb1 gene abnormalities in the diagnosis of parathyroid cancer," Clin Endocrinol (Oxf), 60(1):99-106, 2004.

[23] C. -C. Chang and C. -J. Lin, LIBSVM: a library for support vector machines, 2001. Available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[24] N. V. Chawla, K.W. Bowyer, L.O. Hall and W.P. Kegelmeyer, "SMOTE: Synthetic Minority Over-sampling TEchnique," Journal of Artificial Intelligence Research, vol. 16, pp. 321-357, 2002.

[25] N. V. Chawla, N. Japkowicz and A. Kolcz, "Editorial: Special Issue on Learning from Imbalanced Data Sets," SIGKDD Explorations 6(1): 1-6 (2004).

[26] N. V. Chawla, A. Lazarevic, L. O. Hall and K. W. Bowyer, "SMOTEBoost: Improving Prediction of the Minority Class in Boosting," in 7th European Conference on Principles and Practice of Knowledge Discovery in Databases, pp. 107-119, Cavtat-Dubrovnik, Croatia, Sep. 22-26, 2003.

[27] M. Chernick, Bootstrap Methods: A Practitioner's Guide, Wiley, New York, NY, 1999.

[28] K. K. Chin, "Support vector machines applied to speech pattern classification," Master's thesis, Engineering Department, Cambridge University, 1999.

[29] S. Chiu, "Fuzzy Model Identification Based on Cluster Estimation," Journal of Intelligent and Fuzzy Systems, Vol. 2, No. 3, pp. 267–268, 1994.

[30] N. Cristianini and J. Shawe-Taylor, An introduction to support Vector Machines and other kernel-based learning methods, Cambridge University Press, NY, 1999.

[31] C. Davidson, R. Tirouvanziam, L. Herzenberg, J. Lipsick, "Functional Evolution of the Vertebrate Myb Gene Family: B-Myb, but neither A-Myb nor c-Myb, complements Drosophila Myb in Hemocytes," Genetics, published online, 2004.

[32] C. A. Dinarello, “Interleukin-18,” Methods. 19:121-132, 1999.

[33] K. Duan and J. C. Rajapakse, “A Variant of SVM-RFE for Gene Selection in Cancer Classification with Expression Data,” Proc. of IEEE CIBIB 2004, pp. 49-55, San Diego, 2004.

[34] S. Fears, C. Mathieu, N. Zeleznik-Le, S. Huang, J. D. Rowley, and G. Nucifora, “Intergenic splicing of MDS1 and EVI1 occurs in normal tissues as well as in myeloid leukemia and produces a new member of the PR domain family,” Proc. Natl. Acad. Sci. U.S.A. 93:1642-1647, 1996.

[35] R.A. Fisher, “The Use of Multiple Measurements in Taxonomic Problems,” nnals of Eugenics, vol. 7, part II, pp. 179-188, 1936.

[36] Y. Fu, R. Sun, Q. Yang, S. He, C. Wang, H. Wang, S. Shan, J. Liu and W. Gao, “A block-based support vector machine approach to the protein homology prediction task in KDD Cup 2004,” ACM SIGKDD Explorations Newsletter, vol. 6, no. 2, pp. 120-124, 2004.

[37] T. Furey, N. Cristianini, N.Duffy, D. Bednarski, M. Schummer and D. Haussler, “Support Vector Machine Classification and Validation of Cancer Tissue Samples Using Microarray Expression Data,” Bioinformatics, vol. 16 pp. 906-914, 2000.

[38] C. Furlanello, M. Serafini, S. Merler, and G. Jurman, “Entropy-Based Gene Ranking without Selection Bias for the Predictive Classification of Microarray Data,” BMC Bioinformatics, (4):54, 2003.

[39] C. Fyfe and J. Corchado, “A comparison of Kernel methods for instantiating case based reasoning systems,” Advanced Engineering Informatics, pp. 165-178, 16 (2002).

[40] J. Golab, “Interleukin 18--interferon gamma inducing factor--a novel player in tumor immunotherapy?,” Cytokine. 12:332-338, 2000.

[41] T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caligiuri, et al., “Molecular classification of cancer: class discovery and class prediction by gene expression monitoring,” Science, 286, pp. 531-537, 1999.

[42] S. Gunn, "Support vector machines for classification and regression," ISIS technical report, Image Speech & Intelligent Systems Group, University of Southampton, 1998.

[43] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik, "Gene selection for cancer classification using support vector machines," Machine Learning, vol. 46, pp. 389–422, 2002.

[44] D. Hand, H. Mannila and P. Smyth, Principle of Data Mining, MIT Press, Cambridge, London, 2001.

[45] Y.C. He, Y.C. Tang, Y.-Q. Zhang and R. Sunderraman, "Adaptive Fuzzy Association Rule Mining for Effective Decision Support in Biomedical Applications," International Journal of Data Mining and Bioinformatics, 2006, (accepted).

[46] C.-W. Hsu, C.-C. Chang, C.-J. Lin, A practical guide to support vector classification, Available at http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf.

[47] X. Hu, I. Yoo, I.-Y. Song, M. Song, J. Han and M. Lechner, "Extracting and Mining Protein-Protein Interaction Network from Biomedical Literature," Proc. of the 2004 IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB2004), pp.244-251, San Diego, 2004.

[48] N. Japkowicz and S. Stephen, "The Class Imbalance Problem: A Systematic Study," Intelligent Data Analysis Journal, Volume 6, Number 5, November 2002.

[49] J. -S. R. Jang, "ANFIS: Adaptive-Network-based Fuzzy Inference Systems," IEEE Transactions on Systems, Man, and Cybernetics, Vol. 23, No. 3, pp. 665-685, 1993.

[50] D. Jiang, C. Tang and A. Zhang, "Cluster analysis for gene expression data: A survey," IEEE Transactions on Knowledge and Data Engineering, vol. 16, no. 11, pp. 1370-1386, 2004.

[51] H. Kiss, D. Kedra, Y. Yang, M. Kost-Alimova, C. Kiss, K. P. O'Brien, I. Fransson, G. Klein, S. Imreh, J. P. Dumanski, "A novel gene containing LIM domains (LIMD1) is located within the common eliminated region 1 (C3CER1) in 3p21.3," Hum Genet, 105(6):552-9, 1999.

[52] T. Klaavuniemi, A. Kelloniemi, J. Ylanne, "The ZASP-like motif in actinin-associated LIM protein is required for interaction with the alpha-actinin rod and for targeting to the muscle Z-line," J Biol Chem, 279(25):26402-10, 2004.

[53] A. Kollau, A. Hofer, M. Russwurm, D. Koesling, W. M. Keung, K. Schmidt, F. Brunner, B. Mayer, "Contribution of aldehyde dehydrogenase to mitochondrial bioactivation of nitroglycerin. Evidence for activation of purified soluble guanylyl cyclase via direct formation of nitric oxide," Biochem J., published online, 2004.

[54] S. Kraft, S. Rana, M. H. Jouvin, J. P. Kinet, "The role of the FcepsilonRI beta-chain in allergic diseases," Int Arch Allergy Immunol. 135:62-72, 2004.

[55] B. Krishnapuram, L. Carin, and A. Hartemink, "Joint Classifier and Feature Optimization for Cancer Diagnosis Using Gene Expression Data," Research in Computational Molecular Biology 2003 (RECOMB03), 2003.

[56] M. Kubat and S. Matwin, "Addressing the curse of imbalanced training sets: One sided selection," Proc. of the 14th International Conference on Machine Learning (ICML1997), pp.179-186, 1997.

[57] Y. LeCun, J. Denker, S. Solla, R. Howard and L. D. Jackel, "Optimal brain damage," Advances in Neural Information Processing Systems II, D. S. Touretzky, Ed. Mateo, CA: Morgan Kaufmann, 1990.

[58] J. Li and H. Liu, "Kent ridge bio-medical data set repository," Available at http://sdmc.lit.org.sg/GEDatasets/Datasets.html.

[59] E. C. Liao, B. H. Paw, L. L. Peters, A. Zapata, S. J. Pratt, C. P. Do, G. Lieschke, L. I. Zon, "Hereditary spherocytosis in zebrafish riesling illustrates evolution of erythroid beta-spectrin structure, and function in red cell morphogenesis and membrane stability," Development, 127(23):5123-32, 2000.

[60] T. Y. Lin, "Data Mining and Machine Oriented Modeling: A Granular Computing Approach," Journal of Applied Intelligence, Kluwer, Vol 13, No 2, pp. 113-124, 2000.

[61] T. Y. Lin, "Granular Computing: Fuzzy Logic and Rough Sets," Computing with words in information/intelligent systems, L.A. Zadeh and J. Kacprzyk (eds), Physica-Verlag (A Springer-Verlag Company), 183-200, 1999.

[62] T. Y. Lin, "Granular Computing on Binary Relations I: Data Mining and Neighborhood Systems." Rough Sets in Knowledge Discovery, A. Skowron and L. Polkowski (eds), Physica-Verlag, 1998, 107-121.

[63] T. Y. Lin, "Granular Computing on Binary Relations II: Rough Set Representations and Belief Functions," Rough Sets In Knowledge Discovery, A. Skowron and L. Polkowski (eds), Physica -Verlag, 1998, 121-140.

[64] A. M. Linden, M. Baez, M. Bergeron, D. D. Schoepp, "Increased c-Fos expression in the centromedial nucleus of the thalamus in metabotropic glutamate 8 receptor knockout mice following the elevated plus maze test," Neuroscience 121:167-78, 2003.

[65] D. X. Liu, S. C. Biswas, L. A. Greene, "B-myb and C-myb play required roles in neuronal apoptosis evoked by nerve growth factor deprivation and DNA damage," J Neurosci. 24:8720-8725, 2004.

[66] D. R. Lohmann, B. L. Gallie, "Retinoblastoma: revisiting the model prototype of inherited cancer," Am J Med Genet, 129C(1):23-8, 2004.

[67] J. Ma, Y. Zhao, and S. Ahalt, OSU SVM Classifier Matlab Toolbox, Available at http://www.ece.osu.edu/~maj/osu_svm/.

[68] C.J. Merz and P.M. Murphy. UCI repository of machine learning databases, 1998. Available at http://www.ics.uci.edu/~mlearn/MLRepository.html.

[69] P. Misiano, B. B. Scott, M. A. Scheideler, M. Garnier, "PTH2 receptor-mediated inhibitory effect of parathyroid hormone and TIP39 on cell proliferation," Eur J Pharmacol, 468(3):159-66, 2003.

[70] E. J. Moler, M. L. Chow, and I. S. Mian, "Analysis of molecular profile data using generative and discriminative methods," Physiol. Genomics, vol. 4, pp. 109-126, 2000.

[71] F. Model, P. Adorjan, A. Olek and C. Piepenbrock, "Feature selection for DNA methylation based cancer classification," Bioinformatics, vol. 17 Suppl. 1, pp. S157-S164, 2001.

[72] R. Nakamura, Y. Sato, K. Takagi, N. Sasaki, J. Sawada, S. Kitani, R. Teshima, "Presence and primary sequence of a high-affinity IgG receptor on canine mastocytoma (CM-MC) cells.Immunogenetics," 55:271-4, 2003.

[73] A. Narayanan, E. Keedwell and T. Tayfun, "Analysing prostate cancer gene expression data with single layer neural networks and classical symbolic machine learning techniques," (unpublished).

[74] W. S. Noble, "Support vector machine applications in computational biology," Kernel Methods in Computational Biology. B. Schoelkopf, K. Tsuda and J.-P. Vert, ed. MIT Press, pp. 71-92, 2004.

[75] J. F. Novak and F. Trnka, "Proenzyme therapy of cancer," Anticancer Res. 2005, 25(2A), pp. 1157-77.

[76] M. P. Oyarzo, P. Lin, A. Glassman, C. E. Bueso-Ramos, R.Luthra and L. J. Medeiros, "Acute myeloid leukemia with t(6;9)(p23;q34) is associated with dysplasia and a high frequency of flt3 gene mutations," Am J Clin Pathol. 122:348-58, 2004.

[77] K. T. Patton, M. S. Tretiakova, J. L. Yao, V. Papavero, L. Huo, B. P. Adley, G. Wu, J. Huang, M. R. Pins, B. T. Teh, X. J. Yang, "Expression of RON Proto-oncogene in Renal Oncocytoma and Chromophobe Renal Cell Carcinoma," Am J Surg Pathol, 28(8):1045-50, 2004.

[78] F. Pastorino, C. Brignole, D. Marimpietri, G. Pagnan, A. Morando, D. Ribatti, S. C. Semple, C. Gambini, T. M. Allen, M. Ponzoni, "Targeted liposomal c-myc antisense oligodeoxynucleotides induce apoptosis and inhibit tumor growth and metastases in human melanoma models," Clin Cancer Res. 9:4595-605, 2003.

[79] P. Pavlidis, J. Weston, J. Cai, et al., "Gene functional analysis from heterogeneous data," Proc. RECOMB, New York: ACM Press, pp. 249-255, 2001.

[80] B. E. Peace, K. J. Hill, S. J. Degen, S. E. Waltz, "Cross-talk between the receptor tyrosine kinases Ron and epidermal growth factor receptor," Exp Cell Res, 289(2):317-25, 2003.

[81] W. Pedrycz, Granular Computing: An Emerging Paradigm, Physica-Verlag, 2001.

[82] W. Pedrycz and G. Vukovich, "Granular computing in pattern recognition," Neuro-Fuzzy Pattern Recognition, (H. Bunke and A. Kandel, eds.), World Scientific, 2002.

[83] W. Pedrycz, "Granular computing in Data Mining," M.Last and A. Kandel (eds.), Data Mining & Computational Intelligence, Springer-Verlag, 2001.

[84] L. Ragolia, T. Palaia, T. B. Koutrouby, J. K. Maesaka, "Inhibition of cell cycle progression and migration of vascular smooth muscle cells by prostaglandin D2 synthase: resistance in diabetic Goto-Kakizaki rats," Am J Physiol Cell Physiol, 287(5):C1273-81, 2004.

[85] B. Raskutti and A. Kowalczyk, "Extreme re-balancing for SVMs: a case study," SIGKDD Explorations 6(1): 60-69 (2004).

[86] V. Roth, "The Generalized LASSO: a wrapper approach to gene selection for microarray data," University of Bonn, Dep. Computer Science III, Tech. Report IAI-TR-2002-8, 2002.

[87] J. E. Rundhaug, K. A. Hawkins, A. Pavone, S. Gaddis, H. Kil, R. D. Klein, T. R. Berton, E. McCauley, D. G. Johnson, R. A. Lubet, S. M. Fischer, C. M. Aldaz, "SAGE profiling of UV-induced mouse skin squamous cell carcinomas, comparison with acute UV irradiation effects," Mol Carcinog, published online, 2004.

[88] B. Schölkopf, I. Guyon, and J. Weston, "Statistical Learning and Kernel Methods in Bioinformatics," Artificial Intelligence and Heuristic Methods in Bioinformatics 183, (Eds.) P. Frasconi und R. Shamir, IOS Press, Amsterdam, The Netherlands, pp. 1-21, 2003.

[89] G. P.-Shapiro and P. Tamayo, "Microarray data mining: facing the challenges," SIGKDD Explorations, vol. 5(2), pp. 1-5, 2003.

[90] R. She and F. Chen, Frequent-subsequence-based prediction of outer membrane proteins, in: Proceedings of SIGKDD03, Lise Getoor, Ted E. Senator, Pedro Domingos, Christos Faloutsos (Eds.), 2003, p.436-445, ACM press, Washington, DC, USA.

[91] D. Singh, P. G. Febbo, K. Ross, et al, "Gene expression correlates of clinical prostate cancer behavior," Cancer Cell 1:203-209, 2002.

[92] W. Takahashi, K. Sasaki, N. Kvomatsu and K. Mitani, "TEL/ETV6 accelerates erythroid differentiation and inhibits megakaryocytic maturation in a human leukemia cell line UT-7/GM," Cancer Sci. 2005, 96, pp. 340-8.

[93] Y.C. Tang, Y.C. He, Y.-Q. Zhang, Z. Huang, X. Hu and R. Sunderraman, "A Hybrid CI-Based Knowledge Discovery System on Microarray Gene Expression Data," Proc. of 2005 IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB2005), San Diego, pp.25-30, 2005.

[94] Y.C. Tang, Y.C. He, Y.-Q. Zhang, Z. Huang, X. Hu and R. Sunderraman, "Computational Intelligence-Based Knowledge Discovery on Microarray Gene Expression Data," Special Issue on Biomedical Informatics: Research and Applications, IEEE Transactions on Information Technology in Biomedicine, (under review).

[95-94] Y.C. Tang, B. Jin, Y. Sun and Y.-Q. Zhang, "Granular Support Vector Machines for Medical Binary Classification Problems," Proc. of the 2004 IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB2004), pp.73-78, San Diego, 2004.

[96-95] Y.C. Tang, B. Jin and Y.-Q. Zhang, "Granular Support Vector Machines with Association Rules Mining for Protein Homology Prediction," Special Issue on Computational Intelligence Techniques in Bioinformatics, Artificial Intelligence in Medicine, Vol. 35, No. 1-2, pp. 121-134, 2005.

[97-96] Y.C. Tang, B. Jin, Y.-Q. Zhang, H. Fang and B. Wang, "Granular Support Vector Machines Using Linear Decision Hyperplanes for Fast Medical Binary Classification," Proc. of the 14th annual IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 2005), pp.138-142, Reno, 2005.

[98-97] Y.C. Tang and Y.-Q. Zhang, "Granular Support Vector Machines with Data Cleaning for Fast and Accurate Biomedical Binary Classification," International Conference on Granular Computing (GrC-IEEE 2005), Beijing, 2005.

[99-98] Y.C. Tang and Y.-Q. Zhang, "Granular SVM with Repetitive Undersampling for Highly Imbalanced Protein Homology Prediction," International Conference on Granular Computing (GrC-IEEE 2006), Atlanta, 2006.

[100] Y.C. Tang and Y.-Q. Zhang, "Highly Imbalanced Classification by Granular Support Vector Machines with Repetitive Undersampling," Special Issue on Granular Computing, IEEE Transactions on Fuzzy Systems, (under review).

[101-99] Y.C. Tang, Y.-Q. Zhang and Z. Huang, "FCM-SVM-RFE Gene Feature Selection Algorithm for Leukemia Classification from Microarray Gene Expression Data," Proc. of the 14th annual IEEE International Conference on Fuzzy Systems (FUZZ-IEEE 2005), Reno, pp.97-101, 2005.

[102] Y.C. Tang, Y.-Q. Zhang and Z. Huang, "Two-Stage SVM-RFE to Extract Support Vectors-Based Rules for Reliable Cancer Classification on Microarray Gene Expression Data," IEEE/ACM Transactions on Computational Biology and Bioinformatics, 2006, (under 2nd round review).

[103-100] Y.C. Tang, Y.-Q. Zhang, Z. Huang and X. Hu, "Granular SVM-RFE Gene Selection Algorithm for Reliable Prostate Cancer Classification on Microarray Expression Data," Proc. of The Fifth IEEE Symposium on Bioinformatics & Bioengineering (BIBE 2005), Minneapolis, pp.290-293, 2005.

[104] Y.C. Tang, Y.-Q. Zhang, Z. Huang, X. Hu and Y. Zhao, "Granular SVM-RFE Feature Selection Algorithm for Reliable Cancer-Related Gene Subsets Extraction on Microarray Gene Expression Data," Special Issue on Bioinformatics, Pattern Recognition, (accepted).

[105-101] P. L. Tazzari, A. Cappellini, T. Grafone, I. Mantovani, F. Ricci, A. M. Billi, E. Ottaviani, R. Conte, G. Martinelli, A. M. Martelli, "Detection of serine 473 phosphorylated Akt in acute myeloid leukaemia blasts by flow cytometry," Br J Haematol. 126:675-81, 2004.

[106-102] V. Vapnik, Statistical Learning Theory, John Wiley and Sons, New York, 1998.

[107-103] K. Vareli, M. Frangou-Lazaridis, "Prothymosin alpha is localized in mitotic spindle during mitosis," Biol Cell, 96(6):421-8, 2004.

[108-104] R. Wadgaonkar, L. Linz-McGillem, A. L. Zaiman, J. G. Garcia, "Endothelial cell myosin light chain kinase (MLCK) regulates TNFalpha-induced NFkappaB activity", J. Cell Biochem., published online, 2004.

[109-105] X. C. Wang, K. I. Strauss, Q. N. Ha, S. Nagula, M. E. Wolpoe, D. M. Jacobowitz., "Chymotrypsin gene expression in rat peripheral organs," Cell Tissue Res. 292:345-354, 1998.

[110-106] G. M. Weiss, "Mining with Rarity: A Unifying Framework," SIGKDD Explorations 6(1): 7-19 (2004).

[111-107] G. Wu and E. Y. Chang, "KBA: Kernel Boundary Alignment Considering Imbalanced Data Distribution," IEEE Transactions on Knowledge and Data Engineering, pp. 786-795, Vol. 17, No. 6, June 2005.

[112-108] Y.Y. Yao, "On Modeling data mining with granular computing," Proc. of COMPSAC 2001, Chicago, pp. 638-643, 2001.

[113-109] J. T. Yao and Y. Y. Yao, "A granular computing approach to machine learning," Proc. of FSKD'02, Singapore, pp.732-736, 2002.

[114-110] Y. Yap, X. Zhang, M. Ling, X. Wang, Y. Wong and A. Danchin, "Classification between normal and tumor tissues based on the pair-wise gene expression ratio," BMC Cancer, 4, 72, 2004.

[115-111] X. Yin and J. Han, "CPAR: Classification Based on Predictive Association Rules," Proc. of SIAM International Conference on Data Mining, San Francisco, pp. 331-335, 2003.

[116-112] K. Y. Yip, D. W. Cheung and M. K. Ng, "HARP: A Practical Projected Clustering Algorithm," IEEE Transactions on Knowledge and Data Engineering, vol. 16, no. 11, pp. 1387-1397, 2004.

[117-113] H. Yu, J. Yang and J. Han, "Classifying large data sets using SVMs with hierarchical clusters," Proc. of SIGKDD03, Washington, DC, pp. 306-315, 2003.

[118-114] L. Zadeh, "Fuzzy Sets," Journal of Information and Control, Volume 8, pp 338--353, 1965.

[119-115] L. Zadeh, "Outline of a New Approach to the Analysis of Complex Systems and Decisions Processes," IEEE Transactions on Systems, Man and Cybernetics, SMC-3(1), Jan. 1973.

[120-116] Y. –Q. Zhang, M. D. Fraser, R. A. Gagliano, A. Kandel, "Granular neural networks for numerical-linguistic data fusion and knowledge discovery," IEEE Transactions on Neural Networks, vol.11, no. 3, pp. 658-667, 2000.

[121-117] H. Zhang, C. Y. Yu, B. Singer, M. Xiong, "Recursive partitioning for tumor classification with gene expression microarray data," Proc Natl. Acad. Sci. U.S.A. 98(12):6730-5, 2001.

[122-118] R. A. Zimmerman, J. J. Tomasek, J. McRae, C. J. Haaksma, R. J. Schwartz, H. K. Lin, R. L. Cowan, A. N. Jones, B. P. Kropp, "Decreased expression of smooth muscle alpha-actin results in decreased contractile function of the mouse bladder," J Urol, 172(4 Pt 2):1667-72, 2004.