RELEVANT GENE SUBSET SELECTION: THE MAXIMUM MARGIN CRITERION IN SVM AND GENETIC ALGORITHM

XIAO BING HUANG

Relevant Gene Subset Selection: The Maximum Margin Criterion in SVM and Genetic Algorithm

by

© Xiao Bing Huang

A thesis submitted to the

School of Graduate Studies

in partial fulfilment of the

requirements for the degree of

Master of Science

Department of Computer Science

Memorial University of Newfoundland

July 2006

St. John's Newfoundland

Your file Votre référence
ISBN: 978-0-494-30473-0
Our file Notre référence
ISBN: 978-0-494-30473-0

Abstract

The data collected from a typical *microarray* experiment usually consist of tens of samples and thousands of genes (i.e., features). Usually only a small subset of the features is relevant to the differentiation of the samples. The problem of identifying an optimal subset of features for the differentiation is called *Feature Subset Selection* (FSS). The main purpose of the thesis is to develop a method for relevant gene subset selection using microarray gene expression data. Specifically, this thesis extends the classic *Support Vector Machine* (SVM) algorithm to present a new hill-climbing method *Relevant Subset Selection Using The Maximum Margin Criterion* (RSSMMC) and using its *Genetic Algorithm (GA)* version RSSMMC-GA for feature selection. This method identifies that there are two factors, one *biological* and the other *mathematical*, that can affect the SVM *margin* value. Through an analytic process, we neutralize the mathematical factor, which has no contribution to the relevant gene selection, and utilize the biological factor to select genes which contribute to the increase of the SVM margin. The result subset with a fixed number of features is determined when the maximum accumulative margin value is achieved.

This method is shown experimentally to yield better performance than previous attempts which select features with correlation techniques and *Recursive Feature Elimination* (RFE), to generate biologically relevant genes. In contrast to the former methods, RSSMMC creates a unique and more compact gene subset. Moreover, since the RSSMMC method starts from an empty set to construct the subset whose size is usually small, it consumes less computation time than the comparing methods. This improvement is especially evident in large data sets.

Acknowledgements

I would like to thank my supervisor, Dr. Jian Tang, for his continuous support over the past two years. His enlightening discussions and suggestions helped improve the thesis quality greatly.

I would also like to thank Dr. Guang Sun of the Medical School of Memorial University of Newfoundland. His guidance in bioinformatics and first-hand data resources were very important in the success of this thesis and experimental analyses.

Thanks also to several faculty members in the Department of Computer Science, including Dr. Harold Todd Wareham, who gave helpful suggestions on improving the thesis, and Ms. Elaine Boone, who offered various professional assistances that have made completing this thesis much easier.

Portions of the data used in this thesis were obtained from the study of "Global gene expression profiles of subcutaneous adipose tissue in obese and non-obese young men", supported by Canadian Institute of Health Research Grant 200209MOP-107450-NUT-CJAA-56379.

Finally, I would like to express my gratitude to the thousands of individuals responsible for the development and distribution of robust and freely available software tools such as Linux, LaTeX Eclipse, and related tools that contributed significantly to the successful completion of this thesis.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

In recent years, the production of information has increased to the extent of an information explosion. This explosion has implications to the environment in which we live, to the workplace, the academic world, and our own peace of mind. Most research agrees that, as a result of this explosion of information, we are experiencing a state called *information overload.* For instance, along with the appearances of new biological technologies, huge quantities of biological records are being generated everyday and each of these records usually contains many features. The problem of focusing on the most relevant information in a potentially overwhelming quantity of data has become increasingly important.

In Machine Learning area, the most relevant knowledge is critical for future predication. Given a set of training data and a set of candidate functions known as the hypothesis space $\mathcal{H}$, a supervised learning algorithm takes the training data as input and selects a hypothesis from $\mathcal{H}$ as a target function, where the target function reflects the functional relationship mapping inputs to outputs. The ability of a hypothesis

to correctly classify future data, i.e., those not in the training set, is known as its generalization. Usually, good generalization ability needs a lot of training data, which is often impractical in real-world applications.

Generally, a large number of candidate features may be involved in the learning procedure. The majority of these features are often irrelevant to the target function. This is especially evident in the application domain in this thesis, microarray data analysis. In a typical microarray data set, only tens of samples are available altogether for training and testing while each sample has thousands of genes as the features [27]. Most of the features are irrelevant to determining the target function of classifying the samples. These irrelevant features may lower the learning accuracy, increase learning time and complicate data description. Moreover, in some applications, it is not economic to collect input features that are irrelevant or redundant. Thus, searching for the optimal feature subset, i.e., Feature Subset Selection (FSS), is usually critical.

A typical microarray data set [1] is usually represented by a matrix. The rows are the measurements associated with individual genes while the columns are the measurements associated with the samples. Each entry represents the expression level of one gene of a sample. Typically, an asymmetric relationship exists between genes and samples, i.e., the number of genes (in thousands) is much larger than the number of samples (in tens). The data set can be analyzed in two ways, either by explaining the genes across different samples or by explaining the samples under which the mutually functioned genes vary in the expression levels. This thesis will focus on

[1] Interested readers are referred to Appendix A for the introduction of basic molecular biology and the microarray experiment technology.

the former one. By monitoring gene expression levels in different clinical samples, information can be extracted to understand special gene functions, diagnose disease conditions and test effects of medical treatments.

The main purpose of the thesis is to develop a method for relevant gene subset selection using microarray gene expression data. Since the genes in the resulting subset are most relevant to classify different types of samples, they may have important contributions to the functions of relevant diseases and deserve further medical research.

A SVM-based hill-climbing algorithm *Relevant Subset Selection Using the Maximum Margin Criterion* RSSMMC is proposed and applied to the classification in this thesis. The relevance of each gene is ranked according to its contribution to the classification. The SVM margin, the distance between the SVM hyperplane and the support vectors which are closest to the hyperplane, is adopted as the measurement of relevance (Details of SVM and its margin will be introduced in Chapter 2). Because each relevant gene has its own contribution to the classification, they can be searched stepwise. That is, one relevant gene can be added iteratively to the subset to achieve a higher margin value until the maximum margin value is attained. This thesis identifies that two types of factors, mathematical factor and biological factor, affect the margin. An analytic method is provided in this thesis to neutralize the influence of the mathematical factor, which has nothing to do with the differentiation between samples, to let the biological factor dominate the margin increase. Through such a procedure, these genes are ranked solely by their respective biological contributions to the margin increase.

To test the applicability of RSSMMC both theoretically and practically, an ex-

periment was first conducted on a simulated data set to exhibit its ability on locating the relevant features. Experiments were then performed on a leukemia data set which is publicly available and a newly created obesity data set. In the experiments on the leukemia data set, RSSMMC is shown to yield better performance than the other two comparing methods, RFE and the baseline method (both the methods will be explained in details in Chapter 2 and Chapter 4), in identifying biologically relevant genes. Specifically, in contrast to the comparing methods, the RSSMMC algorithm generates a unique and more compact gene subset. Moreover, since the RSSMMC method starts from an empty set in constructing the subset and the number of relevant features is usually small, it consumes less computation time than another SVM-based algorithm, the sequential version of RFE method does. This improvement is especially evident in large data sets.

In the experiments on the obesity data set, RSSMMC exhibits better ability of locating the obesity-relevant genes than the p-value method (Chapter 4 will provide details for this method), SVM (without using maximum margin as the criterion), and randomized selection. Due to the heuristic essence of the hill-climbing method RSSMMC, the best solution may be hidden by the local optimum in terms of the maximum SVM margin. Therefore, a Genetic Algorithm version of the RSSMMC method, RSSMMC-GA, is also applied to search the optimal feature subset in the whole space. The results of RSSMMC and RSSMMC-GA are shown experimentally to locate the same subset of obesity-relevant genes, although some members are different in their resulting subsets.

This thesis is organized as follows:

Chapter 2 reviews recent work on FSS technologies. This includes a formulated

definition of FSS and a review on main FSS methods. Chapter 2 also provides detailed descriptions of SVM and GA. This chapter summarizes recent applications of SVM and GA in FSS problems lastly.

Chapter 3 describes the hill-climbing method RSSMMC and its GA version RSSMMC-GA. Chapter 3 also presents an analytic method to neutralize the influence of the mathematical factor which has no contribution to the relevant feature selection. This chapter concludes with a discussion of the differences between RSSMMC and RSSMMC-GA.

Chapter 4 reports implementation and experiments results of RSSMMC and RSSMMC-GA on one simulated and two real-world data sets. For the simulated data set, an experiment was conducted to exhibit RSSMMC's ability in locating the most relevant features where the SVM margin is maximized. The experiments on the two real-world data set are divided into two parts. Experiments of RSSMMC on the leukemia data set are first described and the results are compared with two other algorithms. The results from implementations on both RSSMMC and RSSMMC-GA on the obesity data set are then discussed. Lastly, the gene ranking results are investigated against the results from p-value, SVM, and randomized selection.

Chapter 5 concludes the thesis with a summary of the RSSMMC and RSSMMC-GA methods and indicates several future research directions.

Chapter 2

Related Work

In this chapter, a formal description of FSS is provided. Recent work on FSS is reviewed and the SVM concepts are introduced. Since the GA version of RSSMMC is proposed in Chapter 3, the principle of GA is also briefly described. This chapter concludes with a discussion of recent efforts on FSS problems using SVM and GA.

2.1 Feature Subset Selection Problem

2.1.1 The FSS problem

As a classic problem in machine learning, FSS has been defined from various angles. Since the FSS algorithm proposed in this thesis is within the context of classification, the definition described by M. Dash et al is adopted [14]. It is summarized as follows:

FSS attempts to select the minimally sized subset of features while the classification accuracy does not significantly decrease. Specifically, let γ be the original set of features, with cardinality n. Let d represents the desired number of features in the

selected subset X, $X \subseteq \gamma$. Let the FSS criterion for the set X be represented by $f(X)$. Without loss of generality, a higher value of f is assumed to indicate a better feature subset. The problem of FSS is to find a subset $X \subseteq \gamma$ such that $\mid X \mid = d$ and

$$f(X) = \max_{Z \subseteq \gamma,\ |Z|=d} f(Z) \tag{2.1}$$

where Z represents the possible subsets with cardinality d.

The algorithms that tackle the FSS problem can be classified into two main categories, i.e., the *filter* method and the *wrapper* method [31][35]. We discuss Filters and Wrappers in Section 2.1.2 and 2.1.3. The proposed algorithms in this thesis, RSSMMC and RSSMMC-GA, fall into the class of the wrapper method. They use SVM as the learning algorithm and the SVM margin to evaluate the performance of the possible feature subsets.

2.1.2 Filter Algorithms

Filter methods filter out irrelevant features before the learning occurs and thus are independent of the learning algorithms. This preprocessing step uses general characteristics of the training set to select some features. These selected features are then used in the learning algorithm. This approach is computationally more efficient but ignores the relationship between the learning algorithm and the optimal feature subset. Since the learning algorithm is not integrated into the filter algorithm. Instead, the selected subsets are evaluated by other techniques.

The RELIEF [31] algorithm is a filter method that assigns relevance weight to each feature. This algorithm adopts random samples to find the relevance of features. Specifically, it utilizes the difference between the selected samples and the two nearest

samples of the same and the opposite class, called *near-hit*, *near-miss*, respectively. The *ID3* decision tree algorithm is then applied to the training data using only the selected features to induce a decision tree.

The FOCUS algorithm described in [3] searches the minimal combinations of features that perfectly discriminate the classes. This algorithm starts from evaluating each feature in isolation, then turns to pairs of features, triples, and so forth. It stops when a combination that generates perfect partitions of the training set, i.e., in which no samples have different classes (in other words, in each partition all samples have the same class label). The original training examples described using only the selected features are then passed to an algorithm for conducting decision tree classification.

Besides the Decision Tree algorithm, other classifier learning methods, such as *Naive Bayesian classifier*, *Nearest Neighbor Retrieval*, *Cross-Entropy*, and *Principal Components Analysis (PCA)*, have also appeared in recent literature. Blum and Langley summarized these methods in [6].

Filtering methods based on Information theory such as Markov blanket algorithms constitute another broad family. Let μ and σ are two distributions over some probability space Ω, the cross-entropy of μ and σ is defined as $D(\mu, \sigma) = \sum_{x \in \Omega} \log \frac{\mu(x)}{\sigma(x)}$, where μ is the "real distribution" and σ is the approximation to μ. Suppose all the features in feature set F construct a vector f and f_G is used to represent the projection of f onto the variables in G (G is the desired feature subset of F), G will minimize $\Delta_G(\Delta = \sum_f Pr(f)D(Pr(C|f), Pr(C|f_G)))$. Given F_i a feature in G and $G' = G - F_i$, F_i is conditionally independent of the classification function C if and only if $\Delta_{G'} = \Delta_G$. Let M be some set of features which does not contain F_i, M is said to be a Markov blanket for F_i if F_i is conditionally independent of $F - M - F_i$.

Therefore, once a Markov blanket is found, F_i can be safely eliminated. Futhermore, in a backward elimination procedure, F_i will continue to be unnecessary at later stages [36]. Koller et al [36] applied algorithms which iteratively select one candidate set M_i for each feature F_i, and uses a heuristic to estimate how close M_i is to being a Markov blanket for F_i; the feature F_i for which M_i is closest to being a Markov blanket is eliminated.

2.1.3 Wrapper Algorithms

The wrapper methodology, proposed by Kohavi and John [35], offers a straightforward and powerful way to address the FSS problem. In its most general formulation, the wrapper method utilizes a learning algorithm as the black box to score subsets of features according to their predictive power. A typical wrapper algorithm searches for the optimal feature subset by running some learning algorithm on the training data and using some predefined criterion (e.g. the estimated accuracy of the resulting classifier) as its metric. A search strategy is used to search all the candidate feature subsets. This process is implemented across all the candidate subsets and the subset that has the highest predication performance is the resulting feature subset.

In practice, three questions in the wrapper approach need to be answered: (1) how to search the space of all possible feature subsets; (2) which learning algorithm to use; and (3) how to assess the prediction performance of a learning algorithm to guide the search and halt it.

Ideally, an exhaustive search can be performed given a small number of features. Unfortunately, the exhaustive search of optimal feature subset becomes computa-

tionally intractable when a large number of features are processed [1]. To make the wrapper method a feasible technology, a wide range of heuristic search strategies can be used, including *Best-First*, *Sequential Forward Selection*, *Sequential Backward Selection*, *Branch And Bound*, *Simulated Annealing*, and *Genetic Algorithms*, to solve this problem [35]. These search strategies are discussed in Section 2.1.4.

Many learning methods, including Decision Trees, Naive Bayesian classifier, *Least-Square linear classifiers*, and *SVMs*, are used as a subroutine in the subset search procedure.

The performance of a classifier is usually assessed using a validation set or by cross-validation [27]. The objective function (the evaluation standard) often consists of two terms that compete with each other. On one hand, the *goodness-of-fit* needs to be maximized; on the other hand, the number of features needs to be minimized. This characteristics of multi-criteria optimization make the FSS a challenging problem.

2.1.4 Search Strategies

2.1.4.1 Exponential Search

An exponential search method searches the best feature subset exhaustively. Two representative search algorithms in this category are *Branch And Bound* (BAB) and *Beam Search.*

For a feature subset S of size $m(\geq d)$ (d is a constant) under search, BAB [48] uses the best criterion value obtained so far at size m for cutting the branches below S. Its improved version BAB(g) [65] uses the best criterion value obtained so far at

[1] In fact, this problem is shown to be NP-hard by Amaldi and Kann [4].

size $m-g$ for the cutting, where g is a *look-forward* parameter. One limitation of the BAB algorithm is that it requires feature selection algorithm to be monotonic, i.e., the addition of new features to a subset does not decrease the evaluation value for that subset.

In beam search [19], an initial feature subset is stored in a beam. The beam is then expanded to multiple subset by adding untested features to the initial one. These expanded sets are called states. These states are evaluated and assigned scores. The states with the lowest scores are dropped from the beam. Other newly tested states are put in proper places in the beam according to their evaluation scores. The beam expansion proceeds iteratively until no more untested states remain.

The exhaustive essence of the exponential search algorithms makes them computationally expensive and less effective for real-world applications.

2.1.4.2 Sequential Forward Selection and Sequential Backward Selection

The *Greedy Search* strategy (i.e., the former decision is never revisited to include or exclude features in light of new decisions) comes in two flavors: *forward selection* and *backward elimination.* Forward selection methods progressively incorporate features into the subset whereas backward elimination methods start with the set of all features and progressively eliminates the least promising ones. Both methods produce *nested subsets* of features with each super set (of its child set) includes one or more added/removed feature(s) plus all the elements of the child set set.

Sequential Forward Selection (SFS) [16] [47] search starts with an empty set, evaluation is conducted against each feature and the best feature f^* is selected. The combinations of f^* with the other features are then tested and the best subset is

selected. The process continues by putting one more feature into the subset until no more performance improvement for the system can be achieved. *Sequential Backward Selection* (SBS) starts the search from the complete feature set. Let the cardinality of the full set be n, all subsets with $n-1$ features are evaluated and the best subset is chosen, noted as S^*. In the next step, all the subsets of S^* with $n-2$ features are evaluated. This process is run iteratively until the deletion of a feature does not improve performance any more [11]. There are some variations of SFS and SBS to speed up the search process. Instead of processing one at a time, *GSFS(g)* and *GSBS(g)*, the generalized versions of SFS and SBS, respectively, evaluate g features at the same time and the best g-feature subset is chosen for addition or deletion [16] [37]. *PTA(l, r)*, noted as Plus-l take-away-r algorithm, goes forward l stages by adding l features (obtained by SFS) first and then go backward r stages by deleting r features (obtained by SBS) [37]. In its generalized version *GPTA(l, r)*, GSFS(g) and GSBS(g) are used to be the strategies for addition and deletion.

2.1.4.3 Stochastic Search

The Stochastic search algorithm is another main type of FSS technology which includes Simulated Annealing search methods and Genetic Algorithms (GA). Simulated Annealing is a stochastic optimization method that derives its name from the annealing process used to re-crystallize metals. In annealing, temperature is initially set to high in the beginning and then is cooled down to an equilibrium for optimization, i.e., the system reaches a configuration of minimum energy. Namely, at high temperature the algorithm is only searching the gross features (search in a large solution space) of the optimum, while at low temperatures, the finer details (search in a small solution

space) of the optimum start to appear. In FSS problem, if the initial annealing schedule, i.e., the difference of features in the solutions of two neighboring annealing stages is too large, the temperature will decrease very slowly (since the next stages are generated from these neighboring stages and are supposed to still have large differences), allowing the "moves" to higher energy states to occur more frequently. This results in slow convergence. On the other hand, if the annealing schedule is too small and the temperature decreases very fast, the algorithm is more likely to converge to a local minimum [56].

Since RSSMMC-GA integrates GA with RSSMMC in this thesis, the principle of GA is presented in more details in Section 2.2.

2.2 Genetic Algorithm

First proposed by John H. Holland in 1962 [29], GA has been successfully applied in many real-world problems, e.g., optimization and planning, decision making, and feature selection. GA is a particular class of evolutionary algorithms that use techniques inspired by evolutionary biology such as inheritance, mutation, natural selection, and recombination (or crossover). One nice statement of GA is cited as follows:

> "Genetic algorithms are based on a biological metaphor: They view learning as a competition among a population of evolving candidate problem solutions. A *fitness function* evaluates each solution to decide whether it will contribute to the next generation of solutions. Then, through operations analogous to gene transfer in sexual reproduction, the algorithm creates a new population of candidate solutions." [41].

GA is an iterative procedure that searches for an optimal solution in a solution space. Since the solution space is usually huge, GA adopts a heuristic approach. In each iteration. A fixed-size set of candidate solutions, called *population*, are examined. Each member of this population is encoded as a finite string of symbols, e.g. a sequence of zeros and ones. These members are called *chromosomes* and all possible chromosomes form the set of possible solutions in a given problem space. The standard GA procedure imitates the biological evolution. First, a random or heuristic process is conducted to produce the initial population. Second, each member in this population is evaluated according to some predefined quality criterion, referred to as the fitness function. Finally, those which score higher in the fitness function values are assigned higher probabilities to be selected for the creation of the next generation. Thus, individuals with high fitness are more likely to be reserved for reproduction while those with low fitness values are more likely to disappear as the evolution proceeds. This procedure is called *selection.* Basically selection prepares the population for the later reproduction. Reproductions are implemented by special operations. The two best known operations are *crossover* and *mutation.* Crossover is a process between two individuals, named *parents*, in a population. Crossover occurs when the parents exchange parts of their chromosomes to form two new individuals, called *offspring.* This operation, as shown in Figure 2.1(Note that in this figure, the crossover point indicates where the crossover operation occurs. Also note that this demonstrates only the one-point crossover and there are other types of crossover such as multi-point crossover through which the crossover occurs at multiple points at the same time), tends to enable the evolutionary process to move towards more promising regions of the search space. Another important operation, mutation, as illustrated in Figure 2.2,

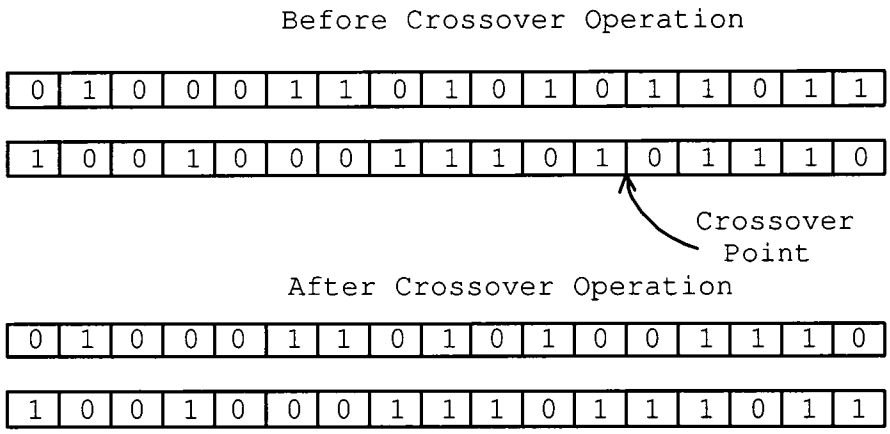

Figure 2.1: Crossover Occurs at the Crossover Point

is applied to randomly sample new points in the search space to prevent premature

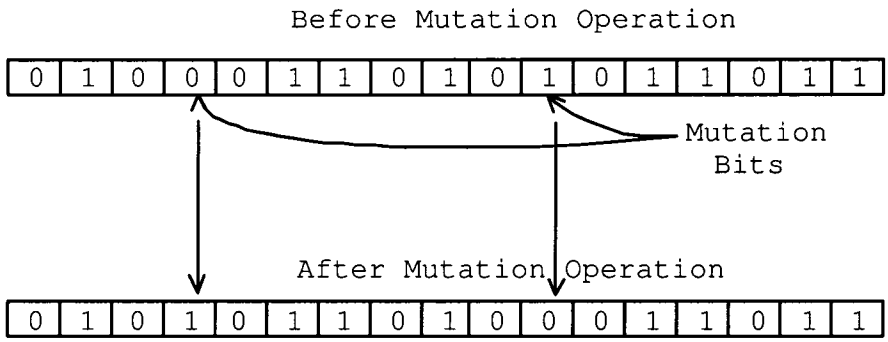

Figure 2.2: Mutation Occurs at the Mutation Points

convergence. The mutation operation flips bits of individual chromosomes at random with some small probability. Generally, GA is a stochastic search process and is not guaranteed to converge. Thus, some termination condition should be specified to stop the iteration. For example, stop after some fixed number of generations or when some

acceptable fitness level is attained. The GA evolution process is briefly summarized in Figure 2.3.

Several points need to be noted. First, the encoding strategies are problem specific, i.e., it can be binary encoding or in other forms, such as permutation, value, tree encoding [45]. Second, the crossover operation can be performed in single point/two points/uniform points (i.e. many random points), as shown in the Figure 2.4. Third, depending on the mutation probability, the mutation operation can occur in many bits in one individual of the population. Fourth, in many real-world applications, various elitism mechanisms are adopted. That is, instead of conducting crossover and mutation for each individual in a generation, the best chromosome (or a few best chromosomes) is (are) directly copied to the next generation. By reserving the best solution(s) in each generation, the performance of GA can often be improved because the loss of the best found solutions is more likely prevented. In a domain which has many dimensions, each dimension denotes a trait, feature, or attribute, the population can be envisioned as a $n+1$-dimensional space with n features and the height corresponding to fitness. The values of the n features of the population construct a n-dimensional *hyperplane.* This hyperplane is usually called a fitness *landscape.* Each individual represents a single point on the landscape and the population is therefore a cloud of points. The GA search moves across the landscape over time as evolution proceeds, this is called *adaptation.* Figure 2.5 shows a landscape formed by two features. The selection procedure "pushes" population upwards in the landscape while genetic operation, e.g. crossover and mutation, can cause the population to skip across hills, thus crossing valleys and leaving local optima.

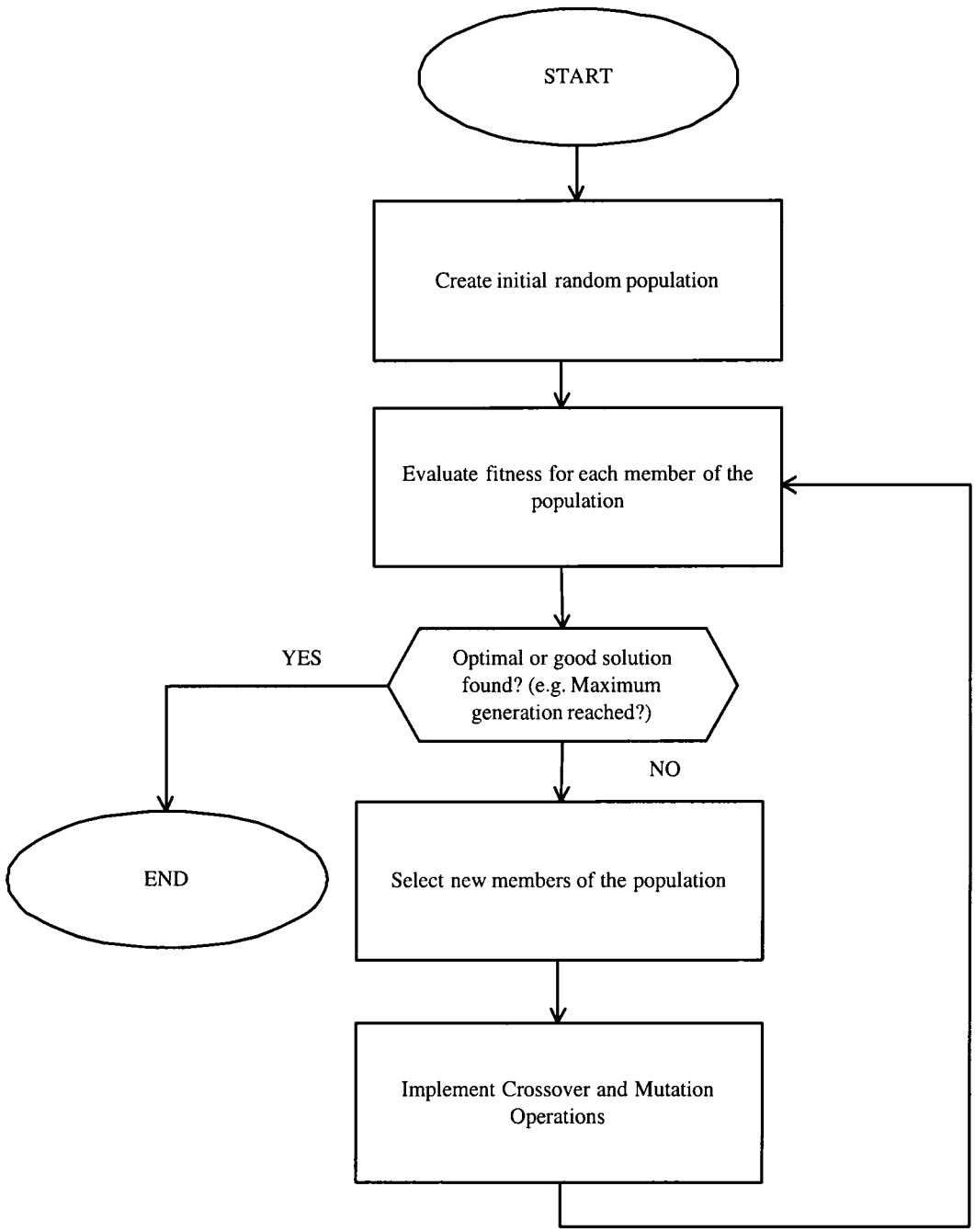

Figure 2.3: A Demonstration of the Basic GA Procedure

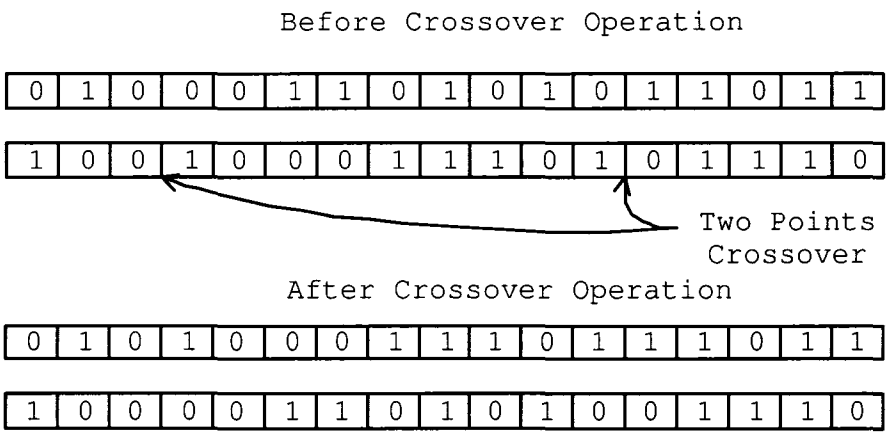

Figure 2.4: Crossover Occurs at Multiple Crossover Points

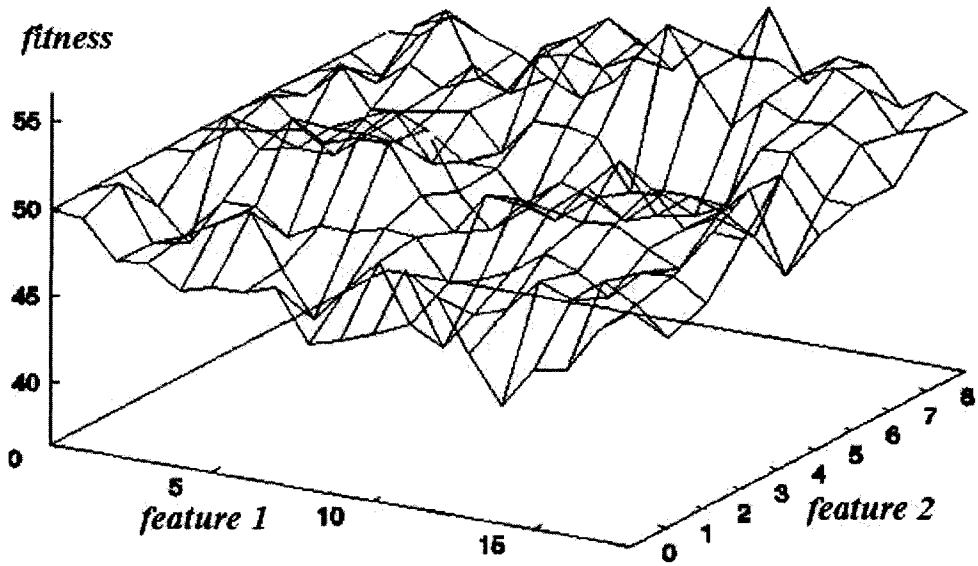

Figure 2.5: Example of a Landscape with Two Features

2.3 Support Vector Machine (SVM)

Although somewhat complex in its mathematical deduction, the basic idea of SVM is straightforward. That is, to find the best hyperplane to separate two classes. In this section, the necessary Algebra knowledge is first introduced. Based on the basic separable case, a brief but complete description of SVM is then presented. The separable case is also extended to the non-separable case and the nonlinear decision function using the *kernel function* (kernel will be explained in full details in Section 2.3.4), which gives SVM the real power on solving real-world problems, is explained. This section concludes with a summary of the SVM implementation adopted in this thesis.

2.3.1 Basic Algebraic Properties of a Hyperplane

Let $\alpha_1, \alpha_2, \ldots, \alpha_n, \alpha_{n+1}$ be scalars that are not all equal to 0. The set S consisting of all vectors $V = (v_1, v_2, \ldots, v_n)^T$ in $\mathcal{R}^n$ such that $\alpha_1 v_1 + \alpha_2 v_2 + \ldots + \alpha_n v_n + \alpha_{n+1} = 0$ is a subspace of $\mathcal{R}^n$, called a hyperplane.

Figure 2.6 illustrates the hyperplane L defined by $f(x) = \beta_0 + \beta^T x = 0$ in $\mathcal{R}^2$. Some important properties of this hyperplane are listed as follows:

1. For any two data points p_1 and p_2 lying in L, $\beta^T(p_1 - p_2) = 0$, i.e., β is orthogonal to $p_1 - p_2$ and thus $\beta^* = \beta/\|\beta\|$ is the normal vector to L;

2. For any point x in L, $f(x)$ can be transformed and $\beta^T x = -\beta_0$ holds;

3. The signed distance of any point x to L is given by $\beta^{*T}(x - x_0) = \frac{1}{\|\beta\|}(\beta^T x + \beta_0)$.

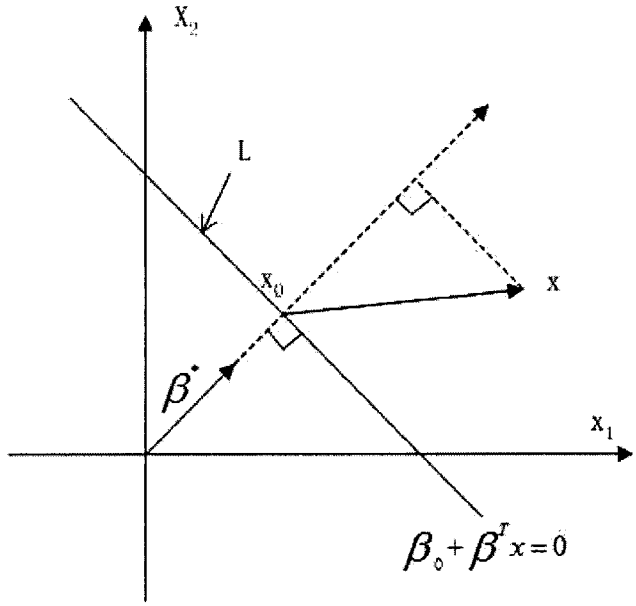

Figure 2.6: Linear Algebra for a 2D Hyperplane

2.3.2 Separable Case

The simplest case of binary separation, the separation of two classes, is accomplished via linear classifiers trained on separable data. In fact, the analysis for the more general case, nonlinear classifiers trained on non-separable data, can be deducted using a quadratic programming method. In Figure 2.7, three hyperplanes can all separate the given binary-labeled training data correctly. Thus, the problem to find the "optimal" separating hyperplane needs to be solved.

Given the training data points $\{x_i, y_i\}, i = 1, \ldots, N, x_i \in \mathcal{R}^d, y_i = \{-1, 1\}$. Suppose there exists some hyperplane which separates the positive from the negative samples (See Figure 2.8). The data points that lie on the hyperplane satisfy $x^T\beta + \beta_0 = 0$, where β is normal to the hyperplane. $\frac{|\beta_0|}{\|\beta\|}$ is the perpendicular distance from the hy-

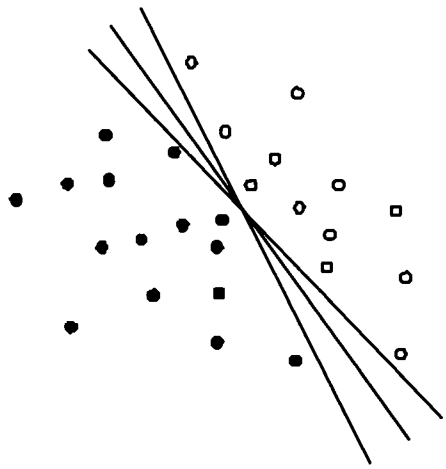

Figure 2.7: Multiple Hyperplanes all Produce the Correct Separation

perplane to the origin, and $\|\beta\|$ is the Euclidean norm of β. Let d_+ (d_-) be the shortest distance from the separating hyperplane to the closest positive (negative) sample, Vapnik [60] defined the "margin" of a separating hyperplane to be $d_+ + d_-$. For the linear separable case, a SVM searches for the separating hyperplane with the largest margin. The formulation of this process is shown as follows:

Suppose that all the training data satisfy the following constraints:

$$x_i^T\beta + \beta_0 \geq +1 \; for \; y_i = +1 \tag{2.2}$$

$$x_i^T\beta + \beta_0 \leq -1 \; for \; y_i = -1. \tag{2.3}$$

The two constraints can be combined into one set of inequalities:

$$y_i(x_i^T\beta + \beta_0) - 1 \geq 0 \; \forall i. \tag{2.4}$$

Let us now consider the points for which the equality in Equation 2.2 holds. These points lie on the hyperplane $H_1 : x_i^T\beta + \beta_0 = 1$ with normal β and perpendicular distance from the origin $\frac{|1-\beta_0|}{\|\beta\|}$. Similarly, the points for which the equality in

Equation 2.3 holds lie on the hyperplane $H_2 : x_i^T\beta + \beta_0 = -1$, with normal β, and perpendicular distance from the origin $\frac{|-1-\beta_0|}{\|\beta\|}$. Thus $d_+ = d_- = \frac{1}{\|\beta\|}$ and the margin is $\frac{2}{\|\beta\|}$. Note that H_1 and H_2 are parallel since they have the same normal and that no training points fall between them. Thus the pair of hyperplanes which gives the maximum margin and separates the two classes, is obtained by solving the following optimization problem:

$$\min_{\beta,\beta_0} \|\beta\|^2 \qquad \text{subject to} \quad y_i(x_i^T\beta + \beta_0) \geq 1, i = 1, \ldots, N \tag{2.5}$$

. Those training data points for which the equality in Equation 2.4 holds are called *support vectors*. This is illustrated in Figure 2.8.

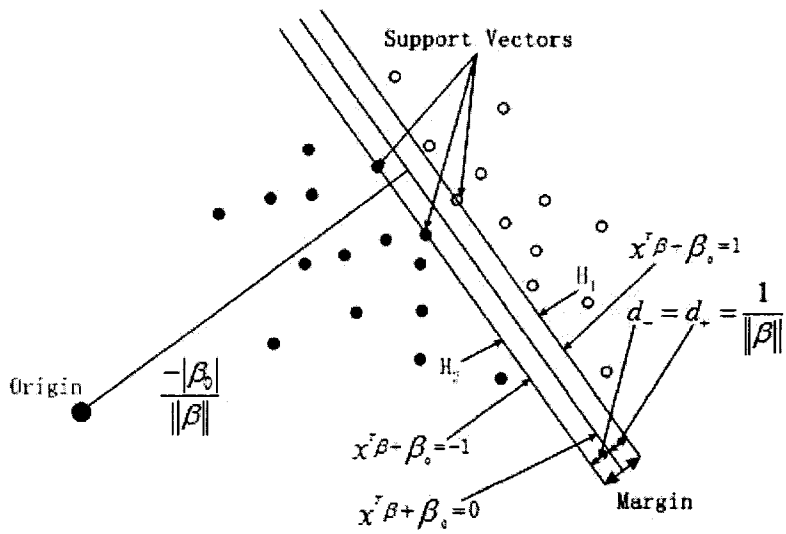

Figure 2.8: The Linear Separating Hyperplane with Three Support Vectors

2.3.2.1 How to Solve the SVM Problem

Expression 2.5 is a standard convex optimization problem. Fletcher provided detailed discussion on solving this problem in [23]. This problem can be formulated using Lagrange multiplier theory. There are two reasons to do this [10]: (1) the constraints $y_i(x_i^T\beta+\beta_0) \geq 1$ can be replaced by constraints on the Lagrange multipliers, which is much easier to handle; and (2) only dot products of vectors of the training data points appear in the new formulation. The second property is crucial for later generalization of the procedure to the nonlinear case.

Positive multipliers $\alpha_i,\ i = 1,\ldots,N$ are introduced (these multipliers are called Lagrange multipliers), one for each of the inequality constraints $y_i(x_i^T\beta + \beta_0) \geq 1$. The Lagrange rule for constraints of the form $C_i \geq 0$ is that the constraint equations are multiplied by positive Lagrange multipliers and subtracted from the objective function to form the Lagrangian. For equality constraints, the Lagrange multipliers are unconstrained. This gives the Lagrangian

$$L_p = \frac{1}{2}\|\beta\|^2 - \sum_{i=1}^{N}(\alpha_i \left[y_i(x_i^T\beta + \beta_0) - 1\right]) \tag{2.6}$$

Setting the derivatives of β to zero, two equations

$$\beta = \sum_{i=1}^{N} \alpha_i y_i x_i \tag{2.7}$$

$$0 = \sum_{i=1}^{N} \alpha_i y_i \tag{2.8}$$

are obtained. Substituting Equations 2.7 and 2.8 into 2.6, the Wolfe dual [23]

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{k=1} N\alpha_i\alpha_k y_i y_k x_i^T x_k \qquad \text{subject to} \quad \alpha_i \geq 0 \tag{2.9}$$

is achieved. According to one important Wolfe dual property, minimizing L_P is equivalent to maximizing L_D. In addition, the Karush-Kuhn-Tucker(KKT) conditions, which include Equations 2.7, 2.8, 2.9, and

$$\alpha_i \left[y_i(x_i^T \beta + \beta_0) - 1 \right] = 0 \; \forall i, \tag{2.10}$$

must be satisfied. In this solution, those data points for which $\alpha_i > 0$ are called support vectors and lie on one of the boundaries H_1 *and* H_2 in Figure 2.8. All other training data points satisfy $\alpha_i = 0$ and lie on the boundary H_1/H_2 such that the equality part holds or on the side of H_1/H_2 (but not on the boundary) such that the strict inequality part holds. As the support vectors are the only constructing elements to form the hyperplane, removing any other data points does not change the separating hyperplane. Once the SVM is trained, a function

$$\hat{f(x)} = x^T \hat{\beta} + \hat{\beta_0} \tag{2.11}$$

is obtained. This function is also noted as the *SVM discriminative function* [44]. Thus, which class a test data point x_k belongs to can be determined by simply checking the sign of this function using the Equation

$$G(x) = sign(x_k^T \beta + \beta_0). \tag{2.12}$$

2.3.3 Non-separable Case

The aforementioned technique for separable data does not apply to non-separable data. When class overlapping occurs, one way to accommodate the new case is to relax the strict constraint which requires no class members can appear on the wrong side to allow some data points to appear on the wrong side of the boundaries. By

defining the slack variables to be $\xi = (\xi_1, \xi_2, \ldots, \xi_N)$, the constraint in Equation 2.4 is rewritten as

$$y_i(x_i^T\beta + \beta_0) \geq 1 - \xi_i, \ \forall i, \ \xi_i \geq 0, \ \sum_{i=1}^{N} \xi_i \leq constant. \tag{2.13}$$

ξ_i in the constraints is the proportional amount by which the prediction $f(x_i) = x_i^T\beta + \beta_0$ can be on the wrong side of its boundaries. Thus the total proportional amount can be bounded through bounding $\sum \xi_i$. From Equation 2.13, misclassifications are known to occur when $\xi_i > 1$. Thus $\sum \xi_i$ is the upper bound of total training misclassifications. Therefore, the optimization problem in non-separable case is

$$\min \|\beta\| \qquad \text{subject to} \quad \begin{cases} y_i(x_i^T\beta + \beta_0) \geq 1 - \xi_i \ \ \forall i, \\ \xi_i \geq 0, \sum \xi_i \leq \ \gamma \end{cases} \tag{2.14}$$

where γ is a constant. The non-separable case is illustrated in Figure 2.9. Similar to

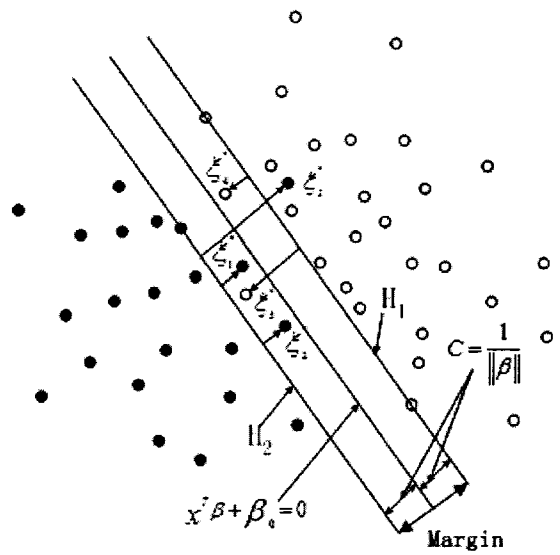

Figure 2.9: Data Points ξ_i^* Appear on the Wrong Side of the Boundaries

the separable case but with the new slack factor ξ_i, Equation 2.5 can be rewritten as

$$\min_{\beta,\beta_0} \frac{1}{2}\|\beta\|^2 + \gamma \sum_{i=1}^{N} \xi_i \qquad \text{subject to} \quad \xi_i \geq 0,\ y_i(x_i^T\beta + \beta_0) \geq 1 - \xi_i \ \ \forall i. \tag{2.15}$$

The separable case now corresponds to $\gamma = \infty$. Like the solution for Equation 2.5, the Lagrangian multipliers can be used to solve Equation 2.15[2].

2.3.4 Nonlinear Decision Functions

When the decision function is not a linear function of the training data, by applying an old kernel method [1] (the dot product in high dimensional space can be mapped to a kernel function of dot product in low dimensional space), the classification process stated in Section 2.3.3 can still be performed. Note that data points in Equation B.5 have the form of dot product, e.g. $x_i \cdot x_j$. These data can be mapped to some high dimensional Euclidean space $\mathcal{H}$ (also noted as *feature space*) using function

$$\Phi : R^d \mapsto \mathcal{H}. \tag{2.16}$$

Note that the training algorithm works only with the dot product $\Phi(x_i) \cdot \Phi(x_j)$ in $\mathcal{H}$. Thus, if some kernel function K is found such that $K(x_i, x_j) = \Phi(x_i) \cdot \Phi(x_j)$, only K will appear in the training algorithm without even knowing what Φ is. Another important advantage of using a kernel in real-world applications is that the algorithm will take almost the same amount of time it would take to train the original data(in a low dimensional space $\mathcal{L}$). In other words, exactly the same linear classification is performed in a different space.

[2]Interested readers are referred to Appendix B for the details on solving Equation 2.15.

There exists a pair $\{\mathcal{H}, \Phi\}$ for all kernels that satisfy Mercer's condition [13], i.e. there exists a mapping Φ and an expansion

$$K(\mathbf{x}, \mathbf{y}) = \sum_i \Phi(\mathbf{x})_i \Phi(\mathbf{y})_i \tag{2.17}$$

if and only if, for any $g(\mathbf{x})$

$$\int g(\mathbf{x})^2 d\mathbf{x} \tag{2.18}$$

is finite. Thus, from Mercer's condition,

$$\int K(\mathbf{x}, \mathbf{y}) g(\mathbf{x}) g(\mathbf{y}) d\mathbf{x} d\mathbf{y} \geq 0 \tag{2.19}$$

is achieved. More detailed discussions about the kernel function can be found in [13] and [60]. In recent literature, three popular kernels are:

$$\text{dth Degree Polynomial Kernel} \qquad K(\mathbf{x}, \mathbf{y}) = (\mathbf{x} \cdot \mathbf{y} + 1)^d \tag{2.20}$$

$$\text{Radial Basis Kernel} \qquad e^{\frac{-\|\mathbf{x}-\mathbf{y}\|^2}{2\sigma^2}} \tag{2.21}$$

$$\text{Neural Network Kernel} \qquad tanh(k\mathbf{x} \cdot \mathbf{y} - \delta) \tag{2.22}$$

Since the SVM kernel evaluations on large size data are very time-consuming, the implementation efficiency of SVM on the microarray data sets, which usually include thousands of genes, is critical. In this thesis, the modified version of *Sequential Minimal Optimization* (SMO) [33], a rapid SVM implementation, is adopted to perform SVM kernel calculations. Since only two samples are evaluated each time in this asymptotical method, this algorithm has rapid implementation speed and good scalability. Its mechanism is described in Appendix C and the Java software implementation for all experiments in this thesis can be obtained by sending a paper request to the thesis author.

2.4 SVM and GA in FSS

2.4.1 SVM in FSS

Guyon et al [28] presented the *Recursive Feature Elimination* (RFE) algorithm to select gene subset for cancer sample classification using SVM. RFE is a Sequential Backward Selection method. Starting from the full set of genes (the initial working set), RFE eliminates the gene with the smallest discriminative power (Recall in Equation 2.5, β is the vector normal to the hyperplane. For $\beta = (\beta_1, \cdots, \beta_n)$, β_i indicates the discriminative power of the *ith* gene) in each iteration. This eliminated gene is removed from the working set (Note that RFE only works with the working set in each loop, i.e., the removed genes are no longer considered in future calculation. This is different from the proposed method RSSMMC in this thesis and will be discussed in Chapter 3). To improve the elimination speed, the actual implementation of RFE removes chunks of genes at a time. After the first iteration, the number of genes that is closest power of 2 is reached (the working gene set is the subset obtained so far and half of the genes with less discriminative power are removed each time). After each subsequent iterations, half of the remaining genes are eliminated. This elimination process results in nested subsets of genes of increasing informative density.

Mao described a *discriminative function pruning analysis* (DFPA) FSS method in [44]. Although the SVM discrimination function (See Equation 2.11) is a non-linear function of the input variables, it has a linear relationship with the kernels. This structure is noted as linear-in-the-parameter structure and its parameters can be estimated using the linear least square estimation algorithm. The goodness of the feature subset is evaluated based on the squared error, which is calculated after

the parameter estimations are obtained. To summarize, DFPA combines a forward selection process with the linear least square algorithm to achieve the pruning step and returns a reduced feature subset.

Suppose that the size of the maximal margin of a SVM is M and $\phi(x_1), ..., \phi(x_l)$ ($\phi(x)$ is the feature space projection of an input variable x) are within a sphere of radius R. Vapnik showed that the SVM performance is related not only on the margin M but also on R [61]. Weston et al showed that the optimal feature subset can be searched by minimizing R^2W^2 where W^2 is the Wolfe dual introduced in Equation 2.9, or some other differentiable criterion, by gradient descent [63]. This can be achieved by repeatedly training a SVM a few times, thus can be very fast in implementation.

Some SVM-based methods search the feature subset using *sensitivity analysis*. Sensitivity analysis is a fundamental concept in *neural networks*. It involves evaluating the deviation of the output of the neural networks, which is caused by perturbations in the input and/or weights. Wang et al defined the sensitivity of SVM as the deviation of margin width with respect to the perturbation of given features [62]. The features are ranked with respect to the values of sensitivity measurement.

Sindhwani et al used *mutual information*, a quantity that measures the independence of two variables, between class labels and classifier outputs as an objective function and applied this objective function in the feature subset selection task in *multilayer perceptrons* (MLPs) and multiclass SVMs [57]. A trained classifier can be visualized as transmitting information across multiple layers of components. The transmission starts from the input layer containing features to the output layer containing class indicators, one of which is fired when the classifier is shown a pattern. The class indicators are output neurons in MLPs and individual binary SVMs in

a multiclass SVM. The transmitted information is calculated and back propagated across the layers of components of the classifier using a heuristic to measure how each components contributes to the information flow. Each feature is assigned a credit in the information back propagation process. These credits represent the relevances of the features. The evaluation of the mutual information objective function is computationally inexpensive and scalable, immune to bias in prior distribution. Recently, mutual information has also been applied to measure the correlations between the features [38].

2.4.2 GA in FSS

Since as a stochastic search algorithm, GA has the freedom to explore more feature subsets than greedy algorithms, it has been incorporated with some feature subset algorithms recently. One of the most difficult aspects of GA is the setting of parameters. GA has four main parameters to be set, i.e., the population size, the maximum number of generations, the crossover rate, and the mutation rate. Initial population of chromosomes are normally randomly generated.

The standard GA was directly applied to data with a small or medium number of features (less than 50) in early GA applications [64]. Typically, through a classification algorithm, a given input sample can be assigned to one of a finite set of classes. Each input sample can also be represented by a subset of features. The size of the subset cannot be too small that important information were lost and it also cannot be too large to lower the accuracy by introducing irrelevant features and increasing learning time and cost. Yang et al selected the representative feature subset through a fitness

function produced in a neural network classification algorithm using only the selected subset of features [64]. The fitness is calculated by combining the accuracy of the classification result with the cost of performing the classification. A fixed set of GA parameters were used for all experiments.

Kudo et al divided the *feature subset* (FS) into three categories: small scale FS, medium scale FS, and large scale FS (if the number of features is in $[0, 19]$, $[20, 49]$, $or [50, \infty]$, respectively) and conducted a comprehensive comparative study of large-scale feature selection [37]. The set of parameters in GA is determined on the basis of the results of experiments using artificial data. Instead of selecting the initial population of chromosomes arbitrarily, Kudo et al adopted two different options to create the chromosomes. The first option has $2n$ extreme feature subsets consisting of n distinct 1-feature subsets (each of the sets has 1 feature) and n distinct $(n-1)$-feature subsets (each of the sets has $(n-1)$ feature) . The second one has $2n$ features subsets in which the number of features is in $[m-2, m+2]$ and all features appear as evenly as possible, where m is the desired number of features and n is the original number of features. The goodness of a feature subset is measured by leave-one-out cross-classification rate of One Nearest Neighbor (1-NN) classifier [3]. The results indicate that GA is suitable for large-scale problems and has a high possibility to find better solutions that cannot be found by other heuristic algorithms.

Loughrey et al [40] studied the *overfitting* problem in wrapper-based FSS algorithms and provided a solution using the *early stopping* strategy. Overfitting happens

[3] 1-NN works as follows: if a sample from the test set is presented to the nearest neighbor classifier, the class label of its nearest (in terms of some distance measure) training sample is declared to be the class label this test sample.

when the feature subsets perform well on the training data but do not perform as well on data that are not used in the training process. Reunanen [53] showed that the degree of overfitting is related to the depth of the search (the more iterations of search occur, the more possible that overfitting happens). In [40], the number of subsets explored is used as an indicator of the search depth and thus as a predictor of overfitting. The overfitting is overcome by using early stopping. This is realized by a cross validation process and the early stopping is performed when the generalization performance starts to fall off in terms of the validation accuracy. The length of time that the GA is allowed to run is reduced through early stopping and the number of subsets that will be evaluated is limited. Thus, the search depth is reduced and overfitting is avoided.

Sun et al wrapped four different classifiers, i.e., a Bayes classifier, a Neural Network classifier, a SVM classifier, and a *Linear Discriminant Analysis* (LDA) classifier, in GA, respectively, to select features subset from the eigen-features provided by Principal Component Analysis (PCA) [59]. The number of 1's and 0's for each chromosome in a population is generated randomly. Sun et al adopted the combination of accuracy from the validation data and the number of used features to evaluate the fitness. In case that the dependent features are far apart in the chromosome, the traditional $1-point$ crossover is applied to destroy such a pattern. In other words, the dependent features with a large distance will get close to each other after the crossover. To reserve the possible dependency between eigen-features, the uniform crossover operation is adopted. The results show that the combination of GA and SVM provides the best performance in terms of the error rate.

Sometimes, features are strongly correlated in real-world problems. One example

is the features derived from the *continuous wavelet* transform [18]. Using GA directly for FSS in the continuous wavelet transform will have poor performance, since GA does not take into account the correlation structure of the features. Dijck et al [18] proposed a 2-stage GA based FSS algorithm to incorporate the feature correlations. This algorithm first constructs basic clusters using the *Hierarchical Agglomerative Clustering Algorithm* [54]. One representative feature from each cluster is selected to form the subsets. The first GA is applied on these subsets to select the one with the best cross validation performance. Features in the clusters that contain the features selected from the first GA process are evaluated in the second GA. In the second GA process, each feature is only allowed to be mutated into a feature from the same cluster. This guarantees each cluster generated in the first GA process is preserved within each solution. The second GA uses the same fitness function as the one in the first GA. The results show that compared with the standard GA, the 2-stage GA finds better solutions in fewer generations.

Chapter 3

RSSMMC and RSSMMC-GA

3.1 SVM Margin: the Criterion of RSSMMC and RSSMMC-GA

This chapter shows that, with the inclusion of new relevant genes to the gene subset generated so far, the change on the SVM margin value is positive, if at all. In contrast those irrelevant genes do not contribute to the separation and thus does not change the margin or their effects on the margin increase are not as evident as the relevant genes do. Therefore, a separating point between the relevant and irrelevant genes can be determined by where the SVM margin reaches its maximum. However, the analysis in this chapter and the experiments in Chapter 4 both indicate that any difference on the expression values of the newly added gene across different samples will increase the margin no matter how small it is due to a mathematical factor. This problem is tackled using an analytic method in this chapter. The proposed method is then expanded to the feature space of SVM.

After neutralizing the influence of the mathematical factor, the hill-climbing method RSSMMC adopts the idea of incremental expansion per iteration in the Sequential Forward Selection (SFS) method (as discussed in Chapter 2) to construct the relevant gene subset. RSSMMC returns the relevant gene subset when the SVM margin reaches its maximum.

In the rest of this chapter, the hill-climbing RSSMMC method is described and formulated. This chapter then discusses how two factors, the mathematical factor and the biological factor, affect the SVM margin value while the gene subset is constructed. An analytic method is provided to neutralize the influence of the mathematical factor. Through such a procedure, all genes can be ranked solely by their biological contributions to increase the margin. Moreover, a GA version of RSSMMC is formulated and presented to search the optimal subset in the whole space. This chapter concludes with a discussion of the differences between RSSMMC and RSSMMC-GA.

3.2 Relevant Subset Selection Using the Maximum Margin Criterion

RSSMMC searches the subset of relevant genes based on the degrees to which they differentiate one group of samples from another. A working set of genes initialized to be empty is used and expanded by adding genes to it one at a time. The genes included earlier are deemed to be more differentially expressed than the genes included later. This idea is implemented in a hill-climbing method RSSMMC using the maximum margin attained from SVM, described as follows.

3.2.1 Formulation of RSSMMC

All genes are initially stored in the remaining gene set. Each sample is a p-dimensional vector of genes. Starting from zero, the size of the working set increases by one in each iteration. In every iteration, the SVM algorithm uses the samples to generate an optimal hyperplane, based on which the SVM margin is obtained. A gene is selected from the remaining genes and included in the working set if it maximizes the margin when the minimum classification errors are achieved. The same process is repeated until the maximum margin is achieved (Note that if the ranking order of all genes are desired, this process can also be implemented continuously until all genes are included in the subset). This process is illustrated in Figure 3.1. This method is noted as Relevant Subset Selection with the Maximum Margin (RSSMMC). In the algorithm, the following notation are adopted:

S_1: the set of the positive samples

S_2: the set of the negative samples

G: the entire set of genes in one microarray experiment

R: the working set of genes, expanding by one in each iteration

A: the set of genes

$SVM(S_1, S_2, A)$: the function that returns the maximum margin between S_1 an S_2 when the minimum test errors are obtained

CP: the maximum margin value in the current iteration

MM: the maximum margin value over all previous iterations

Algorithm 1–**RSSMMC**

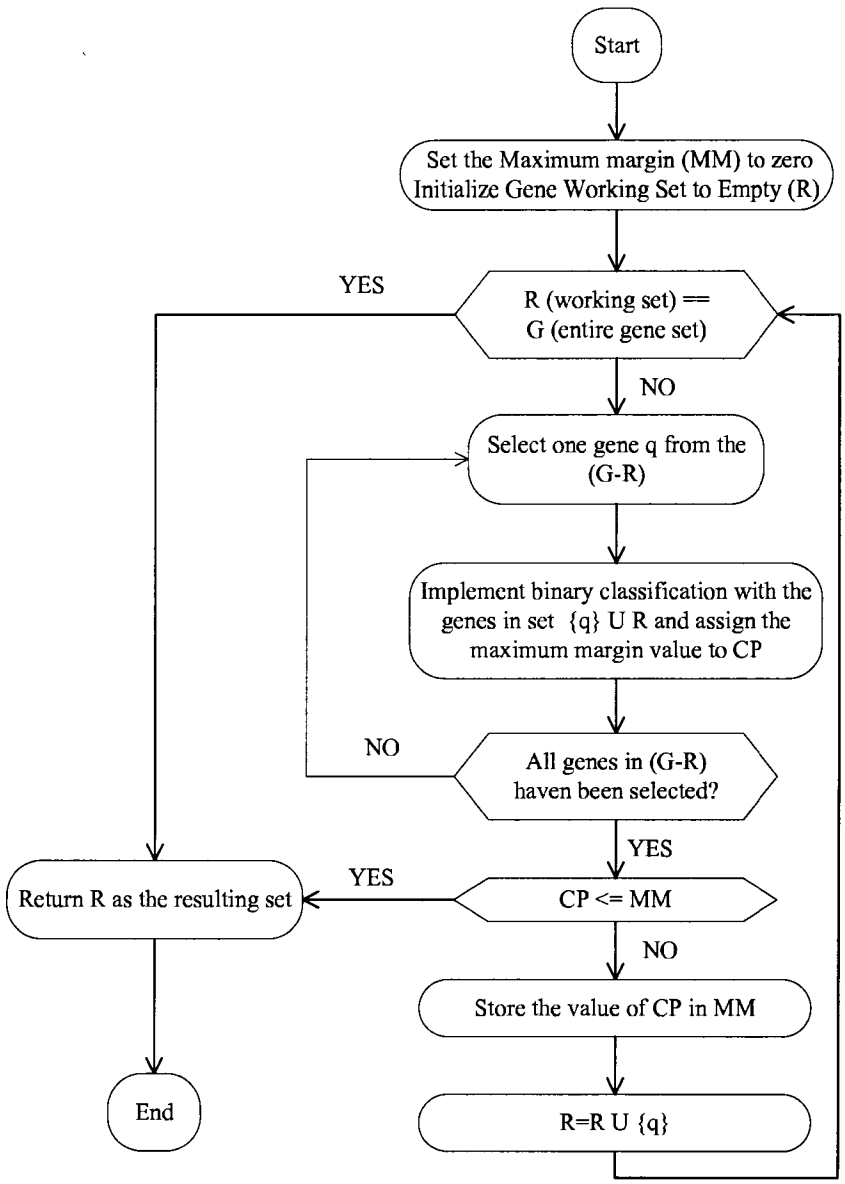

Figure 3.1: The Flowchart of RSSMMC

1. $MM \leftarrow 0$

2. $R \leftarrow \Phi$

3. *while* $R \neq G$ do

4. $\quad CP \leftarrow \max\{SVM(S_1, S_2, \{q\} \cup R) : q \in G - R\}$

5. $\quad if\ CP \leq MM\ return\ R$

6. $\quad MM \leftarrow\ CP$

7. $\quad p = argMax_{q \in G-R}(SVM(S1, S2, q \cup R))$

8. $\quad R \leftarrow R \cup \{p\}$

9. *end while*

The main structure of the algorithm has two loops; the outer loop from lines 3 to 9 expands the working set, R, while the inner loop implicit in line 4 exhausts all the remaining genes in $G - R$ one at a time to find the gene whose contribution together with the genes in the current gene subset R, maximizes the margin in the current outer loop. Recall that RFE [28] does not utilize the removed genes and always calculates its criterion from genes in the remaining gene set after removing the relevant gene(s) in each iteration. This is very different from the principle of RSSMMC. Specifically, RSSMMC evaluates the contributions from genes in the relevant gene subset together with one gene from the remaining gene set in each loop. In this way, the mutual contribution of genes in the relevant gene subset and the new to-be-tested gene is stressed in RSSMMC.

3.2.2 Coping with Increasing Dimensions

The main functionality of the RSSMMC algorithm is realized by the subroutine $SVM(S_1, S_2, A)$. This is implemented with an SVM where the training set is $S_1 \cup S_2$. Each point in the training set has all the genes in A as features. The subroutine returns the margin determined by the optimal hyperplane between S_1 and S_2. However, using the SVM margin directly presents some nontrivial problems. This is explained as follows:

As implied in RSSMMC, the way of judging how relevant a gene is to differentiate one group from another is based on comparing the margins for the same training set of different dimensions (i.e, genes). However, experiments show that, when new dimensions (features) are added to existing dimensions for objects in the training set, the margin always increases, even though the training points differ only by an arbitrarily small non-zero value in the new dimensions (Note here Euclidean distance is applied to obtain the margin). This margin increase seems not to be related to the relevant gene subset construction. One simple demonstration is shown in Figure 3.2 where $Margin2 \geq Margin1$ using the right triangle theorem (experimental data are provided to support this fact in Chapter 4). In other words, the margin increases even when the difference for sample 4 in dimension 2 is small. Therefore, assessing solely whether or not a gene contributes to increasing the margin value is insufficient to judge its capability on differentiating different classes of samples. One approach to address this issue is not only comparing the margins, but also comparing the individual expression levels of the candidate genes. This approach, however, is inappropriate in the context of this thesis. First, the individual gene expression

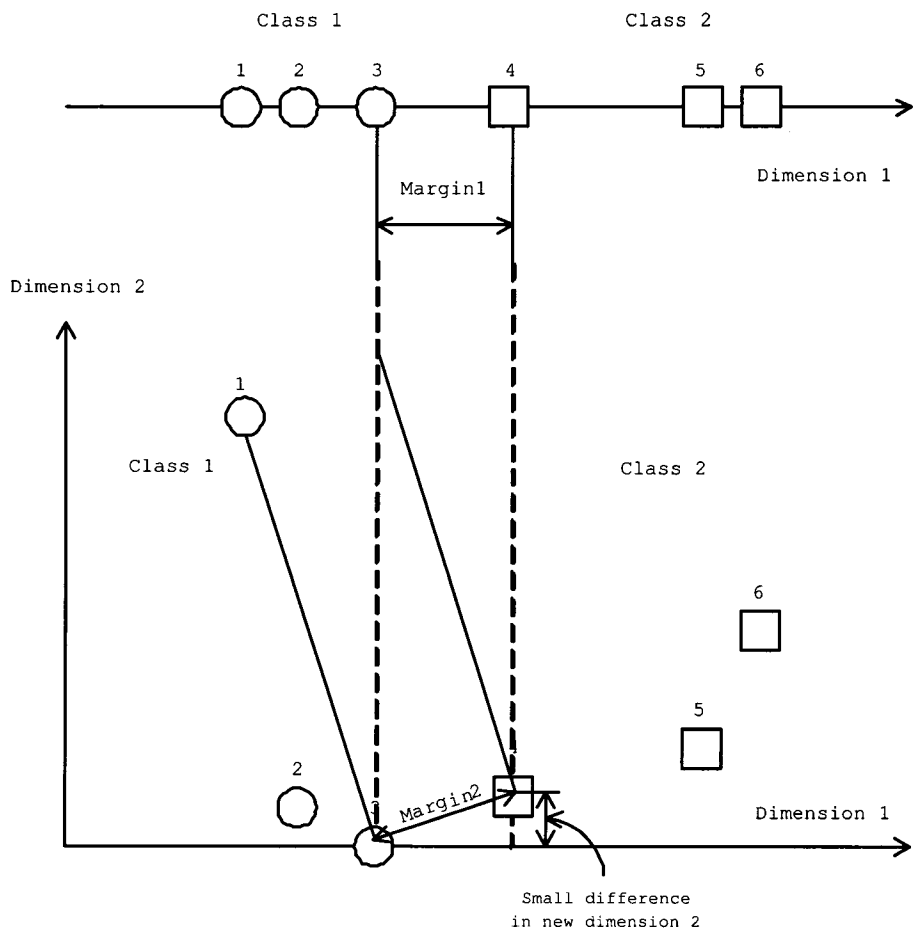

Figure 3.2: The Margin Increase Demonstration from R^1 to R^2

values are measured in the input space, while the margin is calculated in the feature space. Superficially putting them in a pair to form a single measure is awkward in defining a proper metric. For example, how much weight should be given to each of them when pairs are compared? Second, it is difficult, if not impossible, to examine the individual expression values of the genes in the feature space.

The following solution is proposed in this thesis: the individual expression values are incorporated in the margin seamlessly, and then the margin is used as the single measure for comparison (we will explain this in the Section 3.2.3). This solution is based on the observation that, as mentioned before, a small difference in values of the new dimension between classes implies a small difference in margins (but not vice versa). In the next section, how it is applied is discussed.

3.2.3 Normalized Margin

The mathematical effects on the margin increase need to be neutralized so that the biological effects can dominate. This is tackled by multiplying the SVM margin with a fraction, called *normalization factor*, and the product is called a *normalized margin*. A gene must contribute to increasing the normalized margin in order to be selected. The normalization factor should be a decreasing function of the number of current dimensions. This is because more current dimensions result in larger margins between the classes and thus need a smaller normalization factor. To obtain the exact function of the normalization factor is a difficult one since it is not yet known how biologically significant a margin increase is. In the following, a function is deducted with the above property based on a special case, where each of the two classes contains exactly one

training point.

Consider the simplest case where two points are in R^1. Let them be (x_1) and (x_2) (See the cycle and square point in Figure 3.3). Suppose that they are separated by a unit distance in R^1, i.e., $x_2 - x_1 = 1$. 1 is actually the margin value in this case. Now a second dimension is added to both points so they become (x_1, y_1) and (x_2, y_2). Then three possibilities exist which are listed as follows:

Case 1. $\mid y_2 - y_1 \mid > \mid x_1 - x_2 \mid$

Case 2. $\mid y_2 - y_1 \mid = \mid x_1 - x_2 \mid$

Case 3. $\mid y_2 - y_1 \mid < \mid x_1 - x_2 \mid$

In case 3, the newly added dimension y introduces only a smaller difference than the old dimension x. Such a difference is thought to be biologically insignificant. We do not have solid biological evidence yet to support this hypothesis. However, our experimental results give a strong indication that it is biologically reasonable. In other words, if a new dimension is considered in the selecting process, it is selected only if the condition in Case 1 or 2 is true or equivalently only if the $distance((x_1, y_1), (x_2, y_2)) \geq \sqrt{2}(x_2 - x_1)$. Note the distance is the new margin between the two classes, and therefore $\sqrt{2}$ ($d2$ in Figure 3.3) is the minimum required new margin. Similarly, the minimum required new margin is $\sqrt{3}$ ($d3$ in Figure 3.3) when the newly added dimension changes the samples from 2-dimensional space to 3-dimensional space. In general, if the current space is $n-1$-dimensional and the margin is $\sqrt{n-1}$ then a dimension is selected only if the new margin is at least $\sqrt{n}$. Now consider this problem from a different angle. Assume the two points are both d-dimensional, $X = (x_1, \cdots, x_d)$ and $Y = (y_1, \cdots, y_d)$, and 1 additional dimension is added such that X and Y become, respectively, $X' = (x_1, \cdots, x_d, x_{d+1})$ and $Y' = (y_1, \cdots, y_d, y_{d+1})$. In

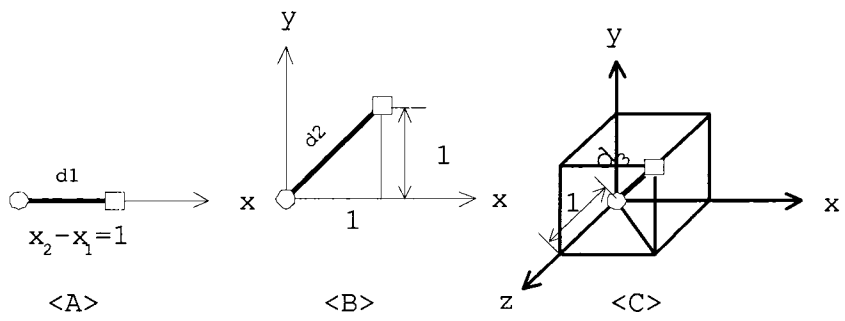

Figure 3.3: The Minimum Requirement on Newly Added Dimensions

order for the margin increase due to additional genes to be remarkable enough, The inequality

$$\frac{\sum_{i=1}^{d+1}(x_i - y_i)^2}{d+1} \geq \frac{\sum_{i=1}^{d}(x_i - y_i)^2}{d} \tag{3.1}$$

is required. The following is the motivation. Note that the value on the right side is the average contribution to the squared margin by each current dimension, and that on the left side is the average contribution to the squared margin by current dimensions together with the margin to be added. The rationale is: if the average contribution combining the new dimension and the current dimensions is less than that by the current dimensions, then the new dimension is considered not contributing enough to differentiate the given sample classes and is not included in the most relevant gene subset. The Inequality 3.1 is equivalent to the following:

$$\frac{\sqrt{(x_1 - y_1)^2 + \cdots (x_{d+1} - y_{d+1})^2}}{\sqrt{d+1}} \geq \frac{\sqrt{(x_1 - y_1)^2 + \cdots (x_d - y_d)^2}}{\sqrt{d}}. \tag{3.2}$$

Which is

$$\frac{Margin\ in\ R_{d+1}}{\sqrt{d+1}} \geq \frac{Margin\ in\ R_d}{\sqrt{d}}. \tag{3.3}$$

Letting $x_i - y_i = 1$ for $1 \leq i \leq d$, Inequality 3.3 becomes $R_{d+1} \geq \sqrt{d+1}$. This

is exactly the requirement that is discussed earlier in this section. Inequality 3.3 is called a *selection condition.* It suggests using $\frac{1}{\sqrt{n}}$ as the normalization factor for n-dimensional training points in the case where each class contains only a single point. To use this function also for the general case, where there are more than two training points, slight modification can be made. In practice, $\frac{1}{n^r}$ is adopted where r, called *normalization suppressor*, is a variant around $\frac{1}{2}$. It can be determined experimentally. A smaller r, and therefore a larger normalization factor, achieves a larger normalized maximum margin value, which in turn lets more relevant genes be included in the relevant subset.

3.2.4 Determining Dimensions in Feature Spaces

In SVM, some complications may arise due to the fact that all input points are mapped to the feature space $\mathcal{H}$ and the optimal hyperplane is computed also in $\mathcal{H}$. Thus, the corresponding number of dimensions of $\mathcal{H}$ for the d dimensional input space need to be calculated to obtain the normalization factor in the feature space. Different kernels produce different $\mathcal{H}$. Therefore, this number is associated with the selected kernel function. In the experiments of this thesis, the cube polynomial kernel

$$(1 + \mathbf{x}_i.\mathbf{y}_i)^3 \tag{3.4}$$

is adopted[1]. Thus, the cube polynomial needs to be analyzed in the general case. Without loss of generality, 1 in Polynomial 3.4 is replaced with $\mathbf{x}_0\mathbf{y}_0$, The polynomial

[1]There are two main reasons to apply the cube polynomial kernel. First, the kernel evaluation is faster with cube polynomial kernel than with other kernels (e.g. Radial Basis Kernel) when multiple kernel calculations are needed. Second, the cube polynomial kernel can achieve zero classification error in one of the data sets in our experiments and this will allow us to focus on investigating the

can be equivalently written as

$$(\mathbf{x}_0\mathbf{y}_0 + \mathbf{x}_1\mathbf{y}_1 + ... + \mathbf{x}_d\mathbf{y}_d)^3 \tag{3.5}$$

where d is the number of dimensions in $\mathcal{H}$. It can be expanded to

$$\begin{gathered}\textstyle\sum_{i=0}^{d} \mathbf{x}_i^3\mathbf{y}_i^3 + 3\sum_{0\le j\le d, j\neq 0} \mathbf{x}_0^2\mathbf{y}_0^2\mathbf{x}_j\mathbf{y}_j + ...+ \\ \\ 3\sum_{0\le j\le d, j\neq d} \mathbf{x}_d^2\mathbf{y}_d^2\mathbf{x}_j\mathbf{y}_j + 6\sum_{0\le i<j<k\le d} \mathbf{x}_i\mathbf{y}_i\mathbf{x}_j\mathbf{y}_j\mathbf{x}_k\mathbf{y}_k\end{gathered} \tag{3.6}$$

This polynomial can be further written as the dot product of the following two vectors (the row vector and the column vector) in feature space

$$\begin{pmatrix} 1, \mathbf{x}_1^3 \cdots \mathbf{x}_d^3, \sqrt{3}\mathbf{x}_1 \cdots \sqrt{3}\mathbf{x}_d, \sqrt{3}\mathbf{x}_1^2, \\ \sqrt{3}\mathbf{x}_1^2\mathbf{x}_2 \cdots, \sqrt{3}\mathbf{x}_1^2\mathbf{x}_d, \sqrt{3}\mathbf{x}_2^2, \sqrt{3}\mathbf{x}_2^2\mathbf{x}_2, \\ \cdots \sqrt{3}\mathbf{x}_2^2\mathbf{x}_d \cdots \sqrt{3}\mathbf{x}_d^2, \\ \sqrt{3}\mathbf{x}_d^2\mathbf{x}_1 \cdots \sqrt{3}\mathbf{x}_d^2\mathbf{x}_{d-1}, \\ \sqrt{6}\mathbf{x}_1\mathbf{x}_2\mathbf{x}_3 \cdots \sqrt{6}\mathbf{x}_{d-2}\mathbf{x}_{d-1}\mathbf{x}_d \end{pmatrix}^T \begin{pmatrix} 1, \mathbf{y}_1^3 \cdots \mathbf{y}_d^3, \sqrt{3}\mathbf{y}_1 \cdots \sqrt{3}\mathbf{y}_d, \sqrt{3}\mathbf{y}_1^2, \\ \sqrt{3}\mathbf{y}_1^2\mathbf{y}_2 \cdots, \sqrt{3}\mathbf{y}_1^2\mathbf{y}_d, \sqrt{3}\mathbf{y}_2^2, \sqrt{3}\mathbf{y}_2^2\mathbf{y}_2, \\ \cdots \sqrt{3}\mathbf{y}_2^2\mathbf{y}_d \cdots \sqrt{3}\mathbf{y}_d^2, \\ \sqrt{3}\mathbf{y}_d^2\mathbf{y}_1 \cdots \sqrt{3}\mathbf{y}_d^2\mathbf{y}_{d-1}, \\ \sqrt{6}\mathbf{y}_1\mathbf{y}_2\mathbf{y}_3 \cdots \sqrt{6}\mathbf{y}_{d-2}\mathbf{y}_{d-1}\mathbf{y}_d \end{pmatrix} . \tag{3.7}$$

The total number of dimensions in the feature space can then be calculated by counting the total number of elements in either of the two vectors in the dot product. The value is

$$(d+1) + d + d^2 + \frac{d!}{3!(d-3)!} \tag{3.8}$$

which is then simplified to

$$(d+1)^2 + \frac{d(d-1)(d-2)}{6}. \tag{3.9}$$

margin increase by the biological difference without having to handle the additional complications generated by the non-separable classes. Note that, in other cases, such as small data sets, Radial Basis Kernel often shows better classification accuracy. Therefore, The decision which kernel should be adopted according to the details of individual experiments.

Thus, the normalization factor for the cube polynomial kernel is:

$$\frac{1}{n^r} = \frac{1}{\left((d+1)^2 + \frac{d(d-1)(d-2)}{6}\right)^r}. \tag{3.10}$$

3.2.5 Considerations on RSSMMC

Like other wrapper-based SFS methods (See Chapter 2), the hill-climbing RSSMMC is a heuristic that assumes each new candidate gene works only with the genes in the working subset. However, it is possible that some other subsets, which include some or all genes not in the working subset, achieve a larger margin value than that of the working subset. In other words, it is possible that the result from the RSSMMC method is a local optimum because hill-climbing methods are local search algorithms. The ideal method of assessing RSSMMC is to compare RSSMMC's resulting feature subset with the global optimum. Unfortunately, it is usually not practical to obtain the global optimum by exhausting all solutions. A straightforward evaluation method is comparing the performance of RSSMMC with that of other methods. The experimental results described in Chapter 4 show that RSSMMC is highly effective and can provide resulting subset that contains biologically relevant genes.

3.3 A GA version of RSSMMC (RSSMMC-GA)

In this section, another heuristic method, Genetic Algorithm, is used to search for the optimum solution in the whole search space. The reason we adopt the GA algorithm is because GA is a global search algorithm contrast to hill-climbing and can search the complete solution space. One of the most important aspects of GA is the definition of the fitness function (See Chapter 2). In general, the higher the fitness value, the better

the solution. To be consistent with the RSSMMC, the value of the maximum margin in the feature space in each generation is adopted as the fitness to implement GA. Note that, GA is not used for assessing the optimality of the RSSMMC method. Rather, possible "good solutions" are investigated through GA evolution process, which might have been overlooked by the hill-climbing RSSMMC. The basic principle of GA has been introduced in Chapter 2. The following section only discuss the aspects that are related to the context in this thesis, the microarray gene data analysis.

3.3.1 GA in Gene Expression Data Analysis

GA is an iterative procedure that searches for an optimal solution in a solution space. In microarray application domain, a candidate solution is a set of genes. A population is a set of candidate solutions. Each set of genes is encoded as a binary string. A gene is in the gene set if and only if the corresponding bit in the candidate solution is 1. RSSMMC-GA implemented in this thesis works as follows: An initial population of members is generated randomly. In each evolutionary step, denoted as a generation, the members in the current population are evaluated according to the fitness function, the normalized margin value in the feature space. Members are selected for reproduction according to their fitness to create the next generation. Specifically, the fitness values of the members in the population are calculated. Then the crossover and mutation operations are applied to all members to obtain the offspring and the fitness values of the offspring are calculated as well. From the union of the two populations, the parent and offspring, the half members with the best fitness values, are selected to form the next generation. This ensures enough number of good blocks in

one generation are reserved for the evolution. Members with high fitness value have a better chance to be selected for reproduction, while low-fitness ones are more likely to disappear. This process is repeated iteratively. In the implementation in this thesis, the iterative process of GA stops if the best fitness value has not increased across a pre-determined number of generations.

3.3.2 Formulation of RSSMMC-GA

The GA version of the RSSMMC method is called RSSMMC-GA and its work flow is illustrated in Figure 3.4. RSSMMC-GA uses the following notation:

S: a solution string representing a subset of genes; a gene is in the subset iff its corresponding bit in the string is '1'

$|S|$: the length of S

GMM: the global maximum fitness value

GS: the global solution

SS: the array of all solutions in each generation

SS': the offspring of SS

PS: the size of SS

PI: the probability that a bit in a string S is set to 1

PC: the probability for selecting a position for crossover operation

PM: the probability for mutation operation

FA: the array to temporarily store the fitness values for all solutions in each generation

C_1: the set of the positive samples

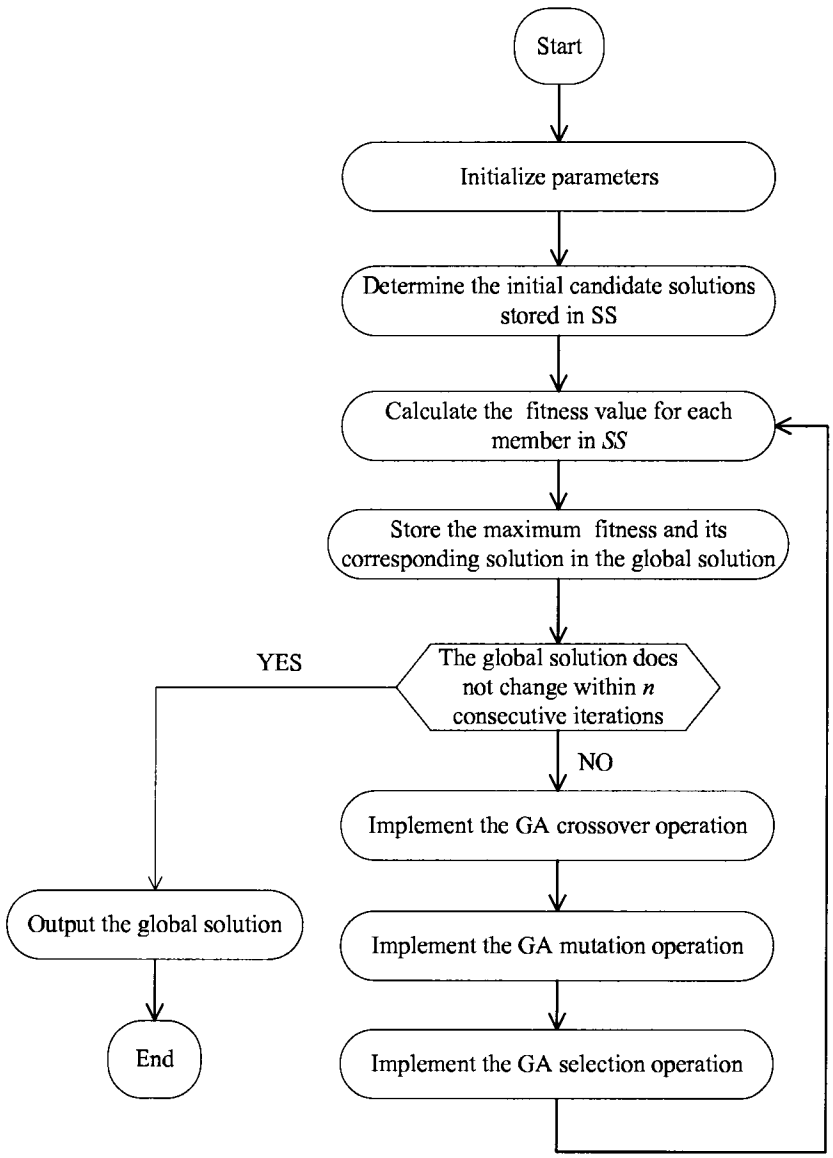

Figure 3.4: The Flowchart of RSSMMC-GA

C_2: the set of the negative samples

A: the set of genes

$SVM(C_1, C_2, A)$: the function that returns the maximum margin between C_1 an C_2 when the minimum classification errors are obtained

$GeneSet(S)$: the set of genes represented by string S

$SubString(S, p1, p2)$: the substring of S from bit $p1$ to bit $p2$

$BitValue(S, p)$: the value of the pth bit in string S

$ConcatenateString(S_1, S_2)$: concatenate the string S_1 and S_2

S_i: substring i, i = $1 \cdots 4$

RSSMMC-GA is described as follows:

1. Initialization.

$SS \leftarrow$ *an array of arbitrary solution strings with size* PS

$GMM \leftarrow 0$

2. Determine the initial candidate solutions with size PS.

For each member S in SS

For $(i = 1; i \leq |S|; i++)$

set BitValue(S, i) to 1 with probability PI

3. Calculate the fitness value for each member in population SS.

For $(i = 1; i \leq |SS|; i++)$

begin

//record fitness function value for each solution

let S_i be the ith string in SS

$FA[i] \leftarrow SVM(C_1, C_2, GeneSet(S_i))$

end

4. Store the maximum fitness and its corresponding solution in global variables.

let S_i be the string in SS

such that $FA[i]$ is the largest in FA

if $FA[i] > GMM$ then

begin

$GMM \leftarrow FA[i]$

$GS \leftarrow S_i$

end

5. Check the stopping criterion.

if GMM's value does not change within n consecutive

iterations

then goto the $9th$ step

6. Implement the GA crossover operation (requires an even number of strings).

For each pair of solutions S_i and S_{i+1} in SS

begin

select j with probability PC

$S_1 \leftarrow SubString(S_i, 0, j-1)$

$S_2 \leftarrow SubString(S_{i+1}, j, |S|-1)$

$S_3 \leftarrow SubString(S_{i+1}, 0, j-1)$

$S_4 \leftarrow SubString(S_i, j, |S|-1)$

$S_i \leftarrow ConcatenateString(S_1, S_2)$

$S_{i+1} \leftarrow ConcatenateString(S_3, S_4)$

store S_i, S_{i+1} *into* SS'

end

7. Implement the GA mutation operation.

```
For each string S in SS'
begin
    For (j = 1; j ≤ |S|; j++)
    begin
        with probability PM do
        begin
            if BitValue(S, j) == 1
                then BitValue(S, j) = 0
            else BitValue(S, j) = 1
        end
    end
end
```

8. Implement the GA selection operation.

```
//reserve the half of strings with the best fitness
//values in both parent and offspring
select the best half (with size |SS|) solutions
from SS ∪ SS'
store the best solutions in SS
repeat the 3rd step
```

9. Output the global solution GS.

3.4 RSSMMC versus RSSMMC-GA

Essentially, The major difference between RSSMMC and RSSMMC-GA is that hill-climbing is a local search algorithm while GA is a global search algorithm. With hill-climbing, genes are processed one at a time to evaluate its impact on classification. In contract, GA processes many different combinations of genes simultaneously. Predictably, RSSMMC always reach the same solution no matter how many times it is tried. On the other hand, the output of GA greatly depends on its settings of parameters, the initial population, the values of crossover and mutation rate, the fitness function, and the random number seeds.

Compared to the stochastic process of RSSMMC-GA, the RSSMMC method is more stable and the number of genes in the resulting feature subset is determined when the SVM margin, the measurement of the distance between the two different classes, attains the maximum value. Contrarily, RSSMMC-GA outputs the subset of genes when the maximum margin is acquired with the current configuration of parameters (this means the number of genes in the subset and the members of the subset vary with different GA setups). On the other hand, unlike RSSMMC, in RSSMMC-GA, the sets of genes selected in different iterations are not necessarily nested and therefore RSSMMC-GA is more flexible in exploring the search space. Thus it has a better chance to reach the optimal solutions.

This chapter provides the formulation of both the hill-climbing method RSSMMC and its GA version RSSMMC-GA. In the next chapter, we will describe the experiments and analyze the results. In the leukemia data set, RSSMMC is shown to achieve a significant improvement over its precedents (e.g. RFE and baseline method)

in terms of the classification accuracy, the size of generated feature subset, and the identification of biologically relevant genes. The gene subsets generated from RSSMMC and RSSMMC-GA on the obesity data set show a large portion of overlapping where both algorithms discover the same obesity-relevant genes. In other words, those genes that have a large impact on the differentiation of different types of samples are captured by both RSSMMC and its GA version. This indicates that the feature subset generated using the methodology in this thesis is capable to approximate the global optimum.

Chapter 4

Empirical Analysis

4.1 Experimental Material and Methods

This chapter first shows results on a simulated data set to exhibit the feature selecting ability of RSSMMC. To investigate the usefulness of RSSMMC in the real-world applications, we present results on two microarray data sets [1]. The first one is the leukemia data set [25] which is available online. The second set, the obesity data set, is a new data source obtained from the study of "Global gene expression profiles of subcutaneous adipose tissue in obese and non-obese young men" [2].

For the leukemia data set, the results of RSSMMC are compared with the baseline method proposed along with the data set [25] and the Recursive Feature Elimination (RFE) method [28]. In the baseline method, each feature (of a pre-selected subset) that is correlated (or anti-correlated) with the class separation is used as a class

[1] All the implementations were written with Java 1.5 on a Linux 2.4.26 kernel and tested on a computer with a X86-64 AMD architecture CPU.

[2] This is a project led by the Discipline of Genetics of Memorial University of Newfoundland.

predictor, albeit an imperfect one. This method is called *Neighborhood Analysis* and works as follows: 1. Find informative genes. One defines an "idealized expression pattern" corresponding to a gene that is uniformly high in one class and uniformly low in the other. One tests whether there is an unusually high density of genes "nearby" (or similar to) this idealized pattern, as compared to equivalent random patterns. Self-Organizing Maps (SOMs) technique is applied for this purpose in [25]. 2. Develop the class predictor. Uses a fixed subset of "informative genes" chosen based on their correlation with class distinction and makes a prediction on the basis of the expression level of these genes in a test sample; each informative gene casts a "weighted vote" for one of the classes, with the magnitude of each vote dependent on the expression level in the test sample and the degree of that gene's correlation with the class distinction; the votes were summed to determine the winning class, as well as a "prediction strength" (PS), which is a measure of the margin of victory that ranges from 0 to 1; the sample was assigned to the winning class if PS exceeded a predetermined threshold, and was otherwise considered uncertain. 3. Test the class predictor. The accuracy of the predictors was first tested by cross-validation on the initial data set and the cumulative error rate is calculated; one then builds a final predictor based on the initial data set and assesses its accuracy on an independent set of samples.

As discussed in Chapter 2, RFE is a sequential backward selection method which eliminates the feature with the smallest discriminative power in each iteration.

The RSSMMC method shows similar results as the RFE method while outperforms the baseline method in terms of the classification accuracy. RSSMMC also generates more diversified subset of genes than RFE. Moreover, the RSSMMC method

provides a fixed number of most relevant genes in the gene subset when the maximum margin is obtained, while RFE only provides nested subsets.

The obesity data set is not yet publicly available. Our experimental result in this thesis is among one of the first analyses of the data set [3]. Since the genes have been prepared pairwise in the significance test in the early statistic processing, the ranking result by p-value is used for comparison. We also compare the results of RSSMMC with that of SVM which does not use maximum margin criterion and the randomized selection process. The RSSMMC method is shown to provide better ranking order of the genes than that of all the comparing methods in terms of the obese gene inclusion.

4.2 RSSMMC Results on Simulated Data

Artificial data from three different distributions were generated to exhibit the ability of RSSMMC to select relevant features. 30 samples with 50 features from three different distributions were created [4]. The first 25 features of the first 15 samples (Class A) were constructed with a $sin()$ based function $y_{i,j} = i * sin(C_1 * j), i = 1, \ldots, 25, j = 1, \ldots, 15$ where C_1 is a constant. The first 25 features of the second 15 samples (Class B) were created with a $cos()$ based function $y_{i,j} = C_2 * i * cos(C_1 * (j - 15)), i = 1, \ldots, 25, j = 16, \ldots, 30$ where C_1 and C_2 are constants. The values of the remaining 25 features across all samples were generated with another $sin()$ based function $y_{i,j} = (i - 25) * sin(C_3 * j), i = 26, \ldots, 50, j = 1, \ldots, 30$ where C_3 is a constant. C_1, C_2 and C_3 are selected such that the two type of samples are more

[3] More information about this data set can be found in http://www.med.mun.ca/genefind/.

[4] For the convenience of description, we use this 30 × 50 matrix for demonstration. Experiments on 100 samples with 1000 features have been conducted and shown the similar results.

differentiated by the first 25 features than by the last 25 features. For all samples, $y_{i,j}$ represents the value of feature j for sample i. After the data were generated, the features were reordered randomly. The reordered data are shown in Figure 4.1. If

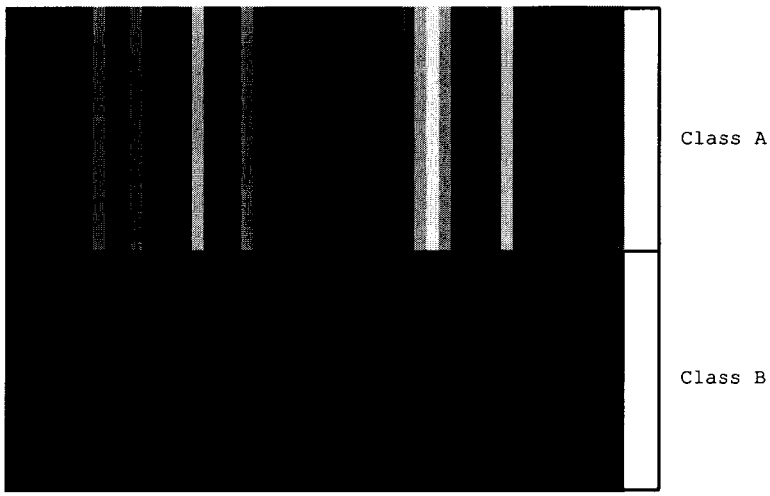

Figure 4.1: The Reordered Simulated Data (The gray shading indicates the feature value of a sample, the lighter the stronger)

RSSMMC works normally, after the implementation, the 25 features with different distribution functions between two classes of samples should be listed first in the output with an order according to the relevance of each feature while the 25 features that are uni-distributed across all the samples should be listed at the end. The experiments results illustrated in Figure 4.2 perfectly matches the prediction. In this figure, samples are represented in rows. Features are represented in columns, listed from left to right according to the descending order of the relevances. The 7 top ranked features are those in the subset when the margin values of the feature subsets reach the maximum. The 7 features show the highest differentiation between two

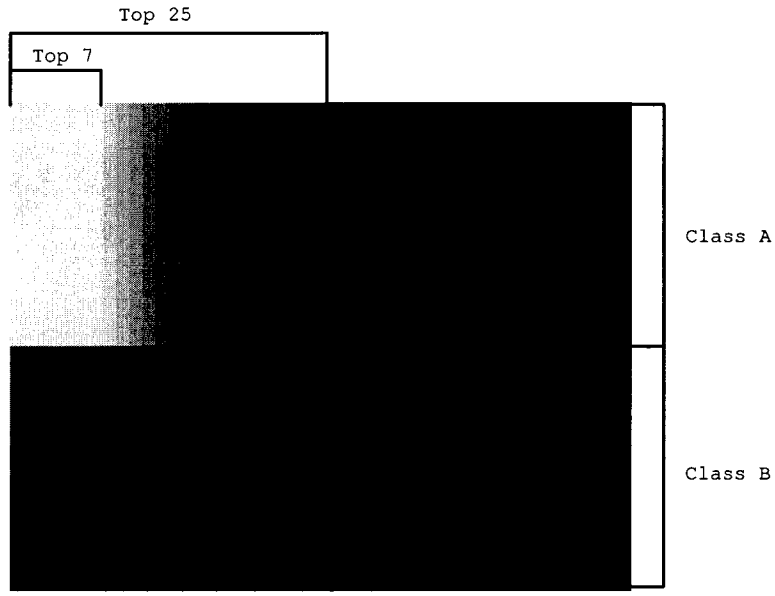

Figure 4.2: RSSMMC's Results on the Reordered Simulated Data

classes in the diagram clearly. In this figure, all the 25 most relevant features have also been selected correctly.

Recall that it has been mentioned in Chapter 3 that the SVM margin value always increases if the mathematical factor is not handled. This is illustrated in Figure 4.3 when the RSSMMC algorithm is implemented without neutralizing the effects of the mathematical factor. After neutralizing the dimensionality effect of the mathematical factor, the final maximum margin distribution is shown in Figure 4.4. The curve in Figure 4.4 matches the prediction, i.e., the margin value reaches the highest when the 7 most relevant features are included (the reason that the peak value does not appear when 25 most relevant features are included is because the margin increase (the numerator) is smaller than the dimension increase (denominator) after 7 genes have been included in the subset).

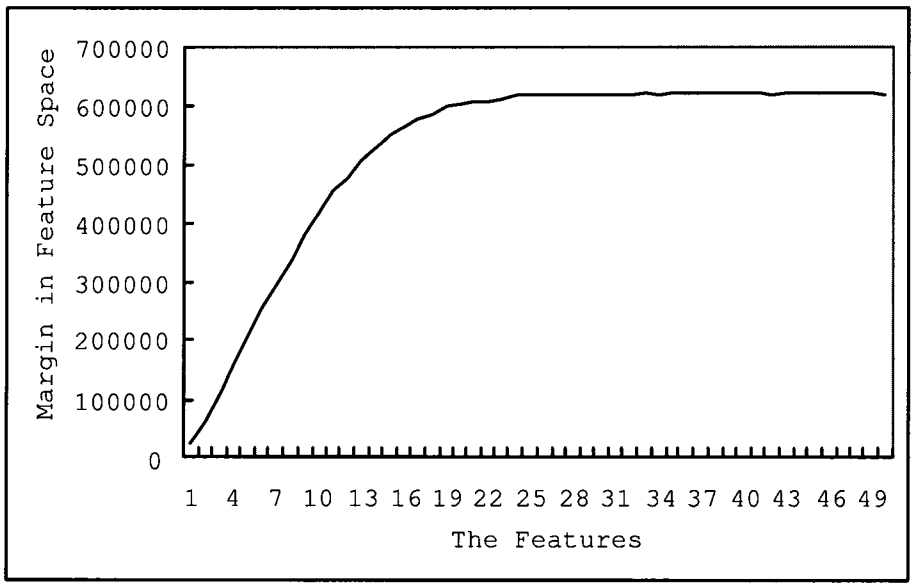

Figure 4.3: The Maximum Margin Distribution across the Features without Neutralizing the Effects of the Mathematical Factor

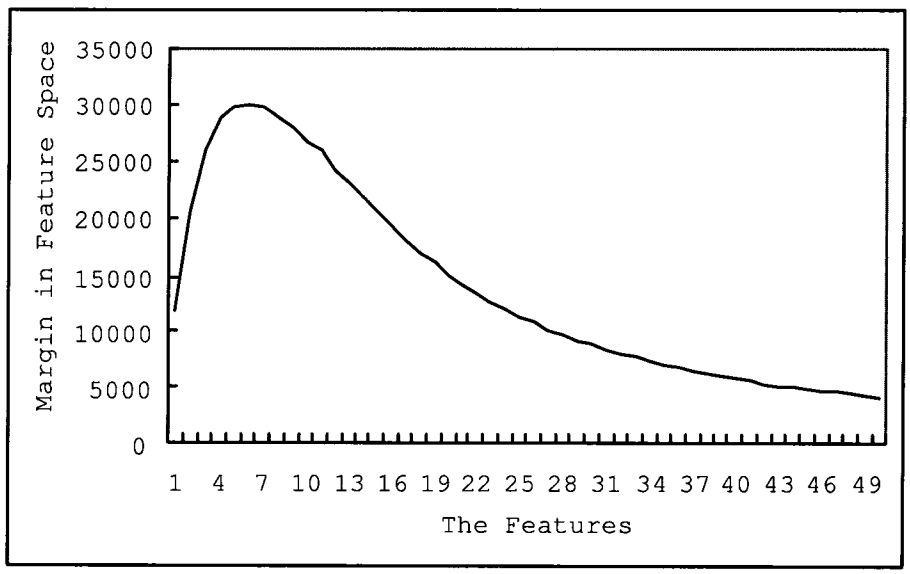

Figure 4.4: The Maximum Margin Distribution across the Features

4.3 Leukemia Data Set

Golub et al [25] presented methods to classify two types of cancer in the leukemia data set, noted as *Acute Lymphocytic Leukemia* (ALL) and *Acute Myelogenous Leukemia* (AML), respectively. This data set is formed by two subsets, i.e., the training and the test set. The training set is used to select a subset of genes and determine the classifiers, and the independent test set is used to estimate the algorithm performance. The training set consists of 38 samples (27 ALL and 11 AML) from bone marrow specimens. The test set has 34 samples (20 ALL and 14 AML), prepared under different experimental conditions, and includes 24 bone marrow and 10 blood sample specimens. All samples have 7129 genes. The preprocessing procedure in this thesis has normalized the original data by setting the minimum threshold to 20 and the maximum to 16000 (Note: The expression values less than 20 or over 16,000 are considered by biologists as unreliable for the experiment and any value exceed the boundaries are replaced with the minimum or the maximum values, whichever is closer). We also standardized the data set as suggested in [25], namely, from each gene expression value, we subtracted its mean and divided the result by its standard deviation (this is also called z transformation). For this data set, the RSSMMC method is compared with the other two, the baseline method [25] and the SVM-based RFE method [28]. Three main factors are considered in the experiments:

1. The classification accuracy, i.e., how accurate the classifier obtained from the training set can predicate the classes of the samples in the test set;

2. The size of the gene subset, i.e., how many features are used to construct the classifier (usually smaller is better);

3. The information provided by the generated subset. This factor stems from practical significance in medical research, namely, how much information the feature subsets generated from the algorithms can provide.

4.3.1 Implementation Results of RSSMMC, RFE, and the Baseline Method

We report the performance of the three classifiers performing the best on the test data (34 samples) in Table 4.1. Note in this table, *Genes#* refers the number of genes of the subset selected by the given method yielding best classification performance. *Error#* indicates the classification errors on the test set, and the *Reject#* represents those samples that could not be determined of the class labels (this happens when the value returned by Equation 2.12 is smaller than a pre-determined threshold). For example, with the 64 top ranked genes generated by RSSMMC, the baseline classifier returns zero classification error and the labels for all samples can be determined without rejection. The patient id numbers of the classification errors are shown in brackets. The results of using all 7129 genes in the three classifiers with no feature selection are also reported for comparison. Note that, in classification test, it is not surprising that RSSMMC has similar results as that of RFE since both methods are SVM-based (they have the same classification method but have different selection methods). The RSSMMC method uses less genes to obtain the same classification accuracy as that of RFE, while both outperform the baseline method. Specifically, RSSMMC only needs 2 genes to achieve zero classification error whereas RFE needs 8 genes (with which REF achieves the minimum *Leave-One-Out* (LOO) classification

Table 4.1: The Comparisons between Classification Results

Select method		SVM classifier	Baseline classifier
RSSMMC	Genes#	2	64
	Error#	0{}	0{}
	Reject#	0{}	0{}
RFE	Genes#	8,16	64
	Error#	0{}	0{}
	Reject#	0{}	0{}
Baseline	Genes#	64	64
	Error#	1{28}	1{28}
	Reject#	6{4,16,22,23,28,29}	6{4,16,22,23,28,29}
Noselection	Genes#	7129	7129
	Error#	5{16, 19, 22, 23, 28}	5{16, 19, 22, 27, 28}
	Reject#	11{2,4,14,16,19,20	22{1,2,4,5,7,11,13,14
		,22,23,24,27,28}	,16 – 20,22 – 29,33}

error [5]) for the same purpose. With fewer features, both RSSMMC and RFE have less classification errors than the baseline method.

The feature subsets generated with the three methods were then investigated. The 16 top ranked genes in the RSSMMC subset are illustrated in Figure 4.5. To compare

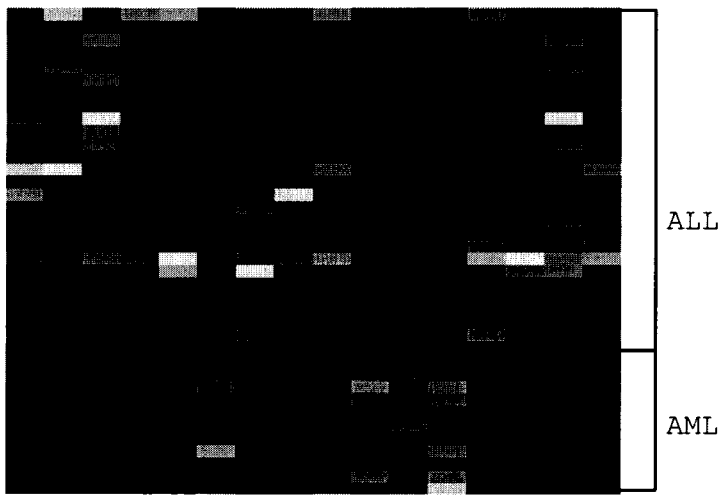

Figure 4.5: The 16 Top Ranked Genes Generated by RSSMMC

the results, The 16 top ranked genes in the RFE and baseline subsets are shown in Figure 4.6. In these matrices, the columns represent different genes and the rows different patients from the training set. The 27 top rows are all ALL patients and the 11 bottom rows are AML patients. In all the figures, the gray shading indicates gene

[5]The Leave-One-Out cross validation method works as follows. Given n samples, including one of the n subset of $n-1$ samples as the training set and the left one the test set each time to implement the classification algorithm. This process is iterated n times and the total classification errors are counted. LOO is often adopted when the number of samples is small and the experiments on the obesity data set in this thesis applied LOO to evaluate the classification accuracy.

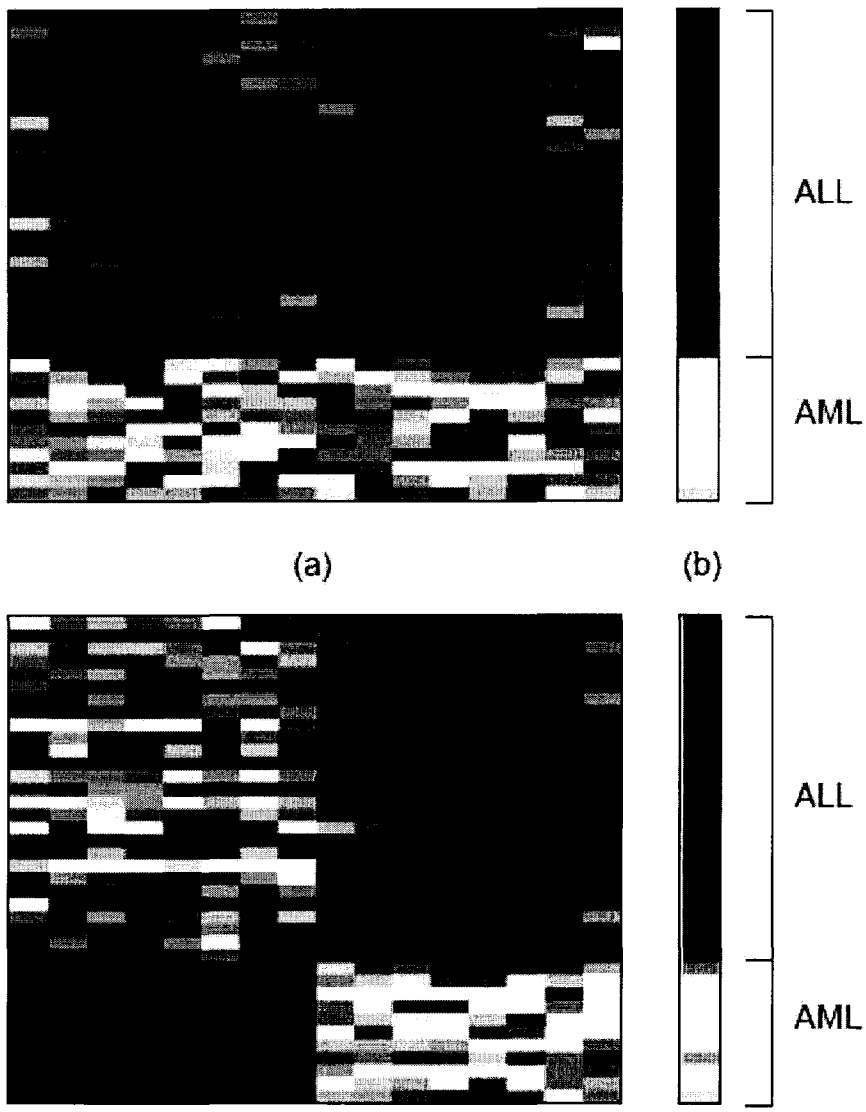

Figure 4.6: The 16 Top Ranked Genes Generated by RFE and the Baseline Method

expression: the lighter the stronger. All the genes selected by RFE are more AML correlated (i.e., most gene expression levels are higher in AML than in ALL), while 4 genes selected by RSSMMC are more AML correlated and the remaining 12 are more ALL correlated. In Figure 4.5, genes are ranked from right to left, the best one at the extreme right, while in Figure 4.6, genes are ranked in the opposite order, the best one at the extreme left (Note the existence of different ordering directions between the two methods is just because the visualization software packages for RSSMMC and RFE/Baseline method have different output formats. In matrices (a) and (c) of Figure 4.6, the columns represent different genes and the lines different patients from the training set. (b) represents the weighted (the weights are the coefficients in the SVM equation) sum of the 16 RFE genes used to make the classification decision. (d) represents the weighted sum of the 16 baseline genes used to make the classification decision). The result that the 16 top ranked genes include both ALL and AML correlated genes shows that RSSMMC has the ability to explore a broader range of relevant genes and creates a gene subset consisting of diversified features.

The baseline Neighborhood Analysis method imposes that half of the genes are AML correlated and half are ALL correlated. The most relevant genes are in the middle. As indicated by Guyon et al [28], the genes selected by the baseline method are strongly correlated with either AML or ALL and therefore there is a lot of redundancy in this gene set. In essence, all the 16 genes by baseline method carry the same information. On the contrary, RSSMMC and RFE carry complementary information since the decision function of SVM based algorithms is actually a weighted sum of gene expression values of selected genes (see Equation 2.12). Both RSSMMC and RFE show a clear ALL/AML separating line. The result of the baseline method is

not as contrasted as the former two.

RSSMMC and RFE produce some overlapping results. For example, the $11th$ ranked gene Zyxin and the $20th$ ranked gene Macmarcks by RSSMMC are the $1st$ and the $3rd$ ranked genes by RFE running on the complete data set with 72 samples, respectively. They have been indicated to have possible function to leukemia in [28]. Our literature research indicates that all the top 16 ranked genes by RSSMMC are more or less related to leukemia function, as listed in Table 4.2 and 4.3. The 50 top ranked genes by RSSMMC are listed in Table D.1, D.2, and D.3 in Appendix D.

Since RFE and RSSMMC are both SVM-based methods. They have some SVM specific settings that need to be stated.

1. RSSMMC applies the cube polynomial kernel and RFE implements dot product for kernel evaluation. This makes RFE runs faster than RSSMMC for a single kernel evaluation.

2. RSSMMC uses SMO algorithm for SVM computation while RFE implements the general version of SVM algorithm. This makes RSSMMC runs faster than RFE for a single kernel evaluation.

3. RSSMMC achieves the feature subset 3-4 times faster than RFE while RFE runs faster than RSSMMC when all the genes are to be screened.

To summarize, while RSSMMC uses less genes to attain the same classification accuracy as that of RFE, both RSSMMC and RFE have better performance than the baseline method. Comparing with RFE, the RSSMMC method has a different ranking order of the generated relevant genes and the members in the relevant subsets by both methods are different, although the top ranked genes from the two methods have some overlapping. These different genes show that RSSMMC has the ability to find

Table 4.2: Possible Leukemia Functions of the 16 Top Ranked Genes by RSSMMC (1-8)

Gene Description	Possible Functions to Leukemia
Terminal deoxynucleotidyl Transferase mRNA (TdT)	TdT is a useful marker in the diagnosis of ALL and distinguishing ALL from mature B-lymphoid neoplasms. Faber et al, 2000 [22].
14-3-3 PROTEIN TAU	Specific 14-3-3 isoforms are linked to genetic. disorders and cancers. MACKINTOSH, 2004 [43].
TCF3 Transcription factor 3 (E2A immunoglobulin enhancer binding factors E12/E47)	TCF3 (E2A) together with TFPT (FB1) gene play an important role in childhood pre-B cell acute lymphoblastic leukemia (ALL). Brambillasca et al, 2001 [8].
CD19	CD19 is a B-cell lymphoma markers. Alkan et al, 1996 [2].
LYN V-yes-1 Yamaguchi sarcoma viral related oncogene homolog	LYN is highly ranked as a feature to distinguish AML/ALL. Bo et al, 2002 [7].
Nucleoside-diphosphate kinase	Nucleoside diphosphate kinase protein (NM23) is involved in tumor metastasis. Okabe et al, 1992 [50].
ATP6C Vacuolar H+ ATPase proton channel subunit	ATP6C is a highly ranked feature in experiments on three public cancer data sets. Ben-Dor et al, 2000 [5].
Interferon-gamma induced protein (IFI 16)	IFI 16 is a highly ranked feature in experiments on three public cancer data sets.Ben-Dor et al, 2000 [5].

Table 4.3: Possible Leukemia Functions of the 16 Top Ranked Genes by RSSMMC (9-16)

LMP2 gene extracted from H.sapiens genes TAP1, TAP2, LMP2, LMP7 and DOB	LMP2 is highly ranked as a feature to distinguish AML/ALL. Bo et al, 2002 [7].
Transcriptional activator hSNF2b	hSNF2b is recognized as an important gene in distinguishing AML/ALL. Daniel et al, 2001 [46].
Zyxin	Encodes a LIM domain protein localized at focal contacts in adherent erythroleukemia cells. Teresita et al, 1996 [42].
TOP2B Topoisomerase (DNA) II beta (180kD)	Purification of TOP2B allows the production of specific antisera of leukemia. Drake et al, 1987 [20].
MB-1	MB-1 is a sensitive and specific reagent for B-lineage lymphoblastic leukemia and in the identification of biphenotypic leukemia presenting as AML. Buccheri et al, 1993 [9].
SPTAN1 Spectrin, alpha, non-erythrocytic 1 (alpha-fodrin)	SPTAN1 is a recognized as an important marker gene in distiguishing AML/ALL. Guan et al, 2005 [26].
Dihydropyrimidinase related protein-2	Dihydropyrimidinase related protein-2 is a highly ranked feature in experiments on three public cancer data sets. Ben-Dor et al, 2000 [5].
CCND3 Cyclin D3	CCND3 as a dominant oncogene in the pathogenesis and transformation in several histologic subtypes of mature B-cell malignancies with chromosomal translocation. Sonoki et al, 2001 [58].

a relevant gene subset which has more diversified combination of genes. Specifically, RSSMMC list not only the genes with higher average expression levels in AML than in ALL but also the genes that are more ALL over-expressed. All the 16 top ranked genes by RSSMMC are related to some leukemia functions. Moreover, unlike the nested subset generated by RFE (the size the subsets decreases according to 2^n where n is the number of iterations), RSSMMC generates a fixed number of relevant gene subset when the maximum margin value is achieved (In the leukemia data set, RSSMMC returns 24 genes which are recognized to be most relevant on the classification while the margin of the gene subset reaches the maximum value). RSSMMC starts from an empty set and expands it to the point where the maximum margin value is obtained while RFE starts from full sized subset and decreases the subset in each iteration. Since usually a small number of genes of the data set are included in the relevant gene subset, the RSSMMC method is much faster than the sequential version of RFE in the implementation generally and slightly faster than the modified version of RFE. As claimed in [28], the modified version of RFE spends 3 hours to process the leukemia data set on a Pentium based PC while RSSMMC just needs about 45 minutes on a similar computer to obtain the gene subset. Note that the modified version of RFE, which removes a chunk of genes at a time to increase the speed, does not provide a clue of which genes in the same chunk are more relevant to the differentiation between the given two classes and therefore are less helpful than RSSMMC which explicitly indicate the relevance of the genes in the resulting subset.

4.4 Obesity Data Set

In recent years, many medical experiments ([24][34][32]) indicate that obesity is a complex molecular function which involves the collaboration of a group of genes. These genes, however, are far from being recognized. In different organisms, the obese gene expression levels vary greatly. Even at the same organism, e.g. abdomen, these expression levels may vary across different races and patients from different geographic areas.

The laboratory in the Discipline of Genetics of Memorial University of Newfoundland collected data from more than 1000 volunteers for the obesity project research. 16 of these volunteers are randomly chosen for comparison research. 8 volunteers are from lean group and the other 8 are from the obese group. The 16 members are all male volunteers selected from the St. John's, Newfoundland area and are all at least the 3rd generation Caucasian Newfoundlanders. Some of their statistical features are summarized in Table 4.4. Data in the table are presented as mean, range in parenthesis. The Gene expression data were obtained using *Agilent*© microarray chips. The whole genome for each individual was recorded according to the standard quality control requirement. The output data are alternatively arranged to form a matrix with rows representing genes and columns the lean and obese samples. Several rows in this matrix are shown in Table 4.5. A question mark in the table indicates that the value at that position is missing. Notice that there are outliers in this table. For example, the expression value of $Gene_1$ of sample L_2 is very large. Similarly, $Gene_3$ of sample O_8 obesity sample has expression value 13.40577 which is significantly larger than all others in the same row. The resulting data from the experiments include a nontrivial

Table 4.4: The Physical and Biochemical Characteristics of Lean and Obese Subjects

Features	Lean	Obese
Age	21.9(20, 25)	23.3(21, 27)
Height(cm)	180(171, 190)	176(167, 187)
Weight(kg)	74.7(56.7, 96.0)	94.5(73.5, 135.0)
% of Body fat	14.9(7.6, 20)	30.5(25.5, 40.3)
Insulin(pmol/L)	46.19(28.6, 78.6)	90.71(34.5, 175)
Glucose(mmol/L)	4.58(1.9, 5.2)	5.35(4.5, 6.5)
Tg(mmol/L)	0.78(0.42, 1.11)	1.12(0.79, 1.55)
Total Chol(mmol/L)	3.82(2.87, 5.41)	4.85(4.01, 5.61)
HDL-C(mmol/L)	1.36(0.94, 1.52)	1.48(0.95, 3.6)
LDL-C(mmol/L)	2.11(1.29, 3.26)	2.87(1.59, 3.68)

Table 4.5: An Obesity Gene Expression Data Example

	O_1	L_1	O_2	L_2	$O_{...}$	$L_{...}$	O_8	L_8
$Gene_1$	0.745295	0.476984	1.657497	1603844	...	...	0.549685	0.314288
$Gene_2$	0.568145	0.196675	0.762890	0.39253	...	...	1.333494	0.297203
$Gene_3$	1.931050	0.391953	3.265161	?	...	...	13.40577	0.296254
$Gene_4$	1.410610	?	0.962336	0.502556	...	...	3.396910	0.238794
$Gene_5$	0.593619	0.119576	0.389406	0.371646	...	...	5.341282	0.174925

portion of outliers. Therefore, an outlier-insensitive algorithm to preprocess the data is desired for unbiased outputs.

The experiments on the obesity data set consist of two main parts, including the implementation of RSSMMC and the implementation of RSSMMC-GA. Note that, unlike the leukemia data set, the original data in the obesity data set have 8.6% missing entries. The existence of these missing values as well as the very small sample size present a new challenge to existing machine learning algorithms. In the experiments in this thesis, we fill these missing data with a bayesian method BPCA proposed by Oba et al [49] (due to the space limitation, we skip the description of this algorithm, interested readers can refer [49] for details). This method has been shown experimentally to have much better estimation ability than other popular methods, such as *singular value decomposition* and K-nearest neighbors. Some marker genes generated by the manufacturer are not real genes and have been removed. The preprocessed genes were then sorted according to the p-values. Paired two tailed t-tests were used to determine significant differences between the expression value of each gene in the obese samples comparing to the lean samples. Data were transformed using log base 2, normalized to eliminate those extreme outliers and standardized with z-scores to ensure normal distribution on all arrays. Significance test was done only on genes which have complete data for at least 5 out of the 8 pairs. When p-value threshold is set to 0.05, 917 genes are reserved. For simplicity, the RSSMMC algorithm is only implemented over the set of 917 genes. In the following, this set is called the *base set.* On the selection of obese relevant genes from the base set, RSSMMC shows a significant improvement over the p-value ranking method, SVM without using the cumulative maximum margin criterion described in this thesis, and

the randomized selection method in terms of the inclusion of the known obese gene candidates (these genes all have annotations on the obesity functions and are called obesity-relevant genes hereafter).

4.4.1 RSSMMC Experiments

Since the samples of the obesity data set were prepared in pairs (i.e., lean-obese pairs) for p-value test originally, the ranking results by p-value are very informative. Therefore, in addition to using the SVM algorithm and randomized selection as the comparing algorithm, we will also compare the feature selection results by RSSMMC with that by p-value on the obesity data set.

To evaluate the results, the genes (from the base set) which have annotations of the obese functions are identified from recent medical literature and summarized in Table 4.6.

Since we will compare the ranking effects by the four mentioned methods on the obesity data set, the RSSMMC algorithm is allowed to iterate through all the genes, i.e., RSSMMC is not stopped when the maximum margin is achieved [6]. For this purpose, a slight modification of Algorithm-1 in section 3.2.1 produces Algorithm-2:

[6] In such a case, the maximum number of SVM calculation (mostly on the SVM kernel evaluations) will be $n + (n-1) + (n-2) + \cdots + 1$. Note that though the number of SVM calculation decreases, the dimensionality in the kernel increases which results in more time consumption in each kernel evaluation. In other words, there are two factors affect the computational complexity, the number of the SVM calculation and the size of the gene set. Thus, we can predict that the running time of the algorithm will reach its peak value at some point in the middle of the run.

Table 4.6: The Obesity Gene List

Systematic Name	Gene Symbol
NM_000863	$HTR1B$
NM_003356	$UCP3$
NM_002024	$FMR1$
NM_002734	$PRKAR1A$
NM_005399	$PRKAB2$
NM_003749	$IRS2$
NM_139322	$ATRN$
NM_000142	$FGFR3$
NM_006399	$BATF$

T: the HC-list.

Algorithm-2

1. $T \leftarrow \Phi$
2. $R \leftarrow \Phi$
3. *while* $R \neq G$ do
4. $CP \leftarrow max\{SVM(S_1, S_2, \{q\} \cup R) : q \in G - R\}$
5. $p = argMax_{q \in G-R}(SVM(S1, S2, q \cup R))$
6. *append* p *to* T
7. $R \leftarrow R \cup \{p\}$
8. *end while*
9. *return* T

Algorithm-2 differs from Algorithm-1 presented in Chapter 3 only in that step 1,

5, and 6 in Algorithm-1 are removed and new steps 1, 5, 6, and 9 are added. What it accomplishes is to reorder the genes based on their contribution to distinguish obese from lean groups. (We call this list HC-list.) In each iteration, it expands the working set by one gene based on the same idea as that in Algorithm-1. At the same time, it appends that gene to the list. The genes entered into the list earlier contribute more than those entered later. The Ranked result of the 9 obesity related genes is shown in Table 4.7.

Table 4.7: The Obesity Gene Ranked List

Rank	Systematic Name	Gene Symbol
1	NM_003749	$IRS2$
2	NM_139322	$ATRN$
3	NM_000863	$HTR1B$
4	NM_006399	$BATF$
5	NM_002024	$FMR1$
6	NM_000142	$FGFR3$
7	NM_005399	$PRKAB2$
8	NM_002734	$PRKAR1A$
9	NM_003356	$UCP3$

4.4.1.1 Experimental Results

Since the size of the samples is very small, 10 fold cross validation is applied on the obesity data set. In the early experiments, no single gene can perfectly separate the

two groups, i.e. the classification error always occurs, when the dot product kernel or quadratic kernel is adopted. The perfect separation is achieved in two genes when the cube polynomial kernel is applied. Since this research is to investigate the influence of biological difference on the margin difference, the cube polynomial kernel is used throughout the experiments to avoid the additional complications created by the non-separable classes.

As mentioned in Section 4.2, the SVM margin value increases monotonously if the mathematical factor is not handled. This is illustrated in Figure 4.7 where the mathematical factor is not neutralized. The final maximum margin distribution with-

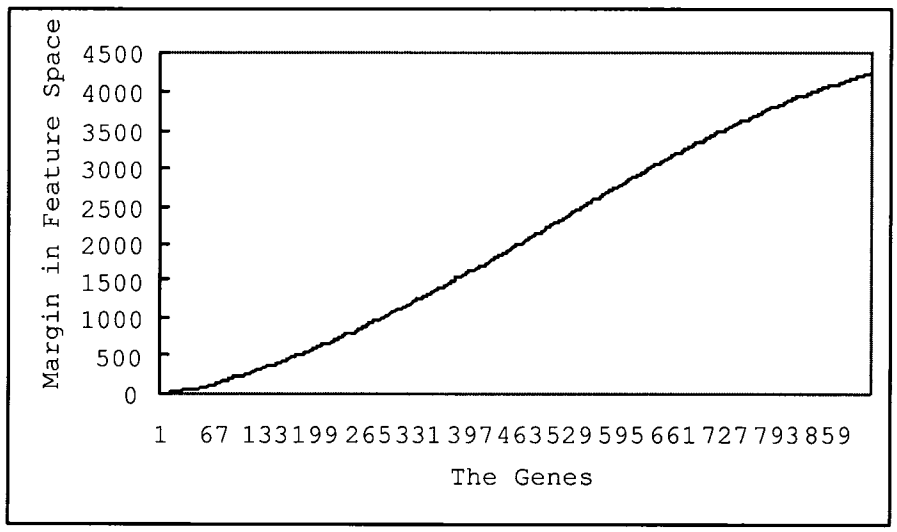

Figure 4.7: The Maximum Margin Distribution across the Obesity-relevant Genes without Neutralizing the Effects of the Mathematical Factor

out the influence of the mathematical factor is shown in Figure 4.8. The curve in Figure 4.8 matches the prediction. The margin value reaches the highest value some-

where in the curve by those most relevant genes. After the maximum margin value is reached, the margin value decreases slowly, signifying the newly added genes weaken the discriminative power of the selected gene subset.

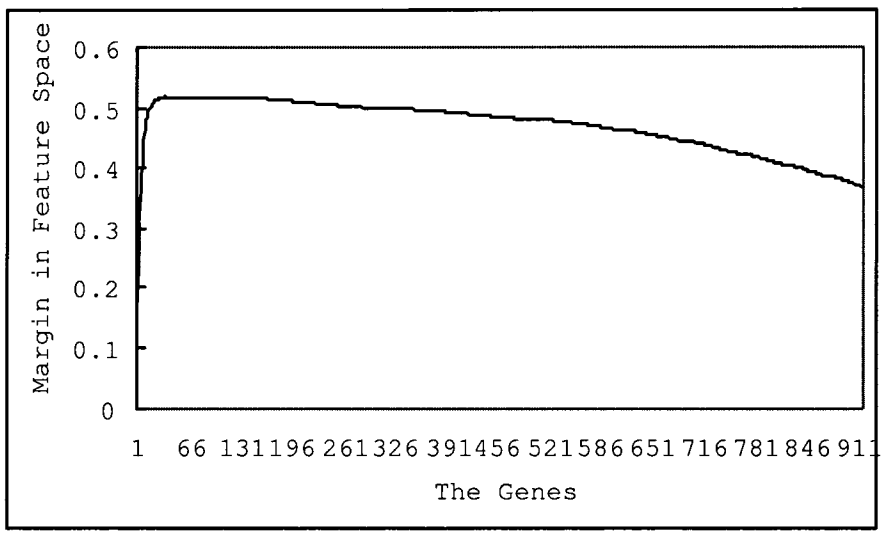

Figure 4.8: The Maximum Margin Distribution across the Obesity-relevant Genes

Although accuracy can be used to achieve the evaluation task for the algorithm RSSMMC and the comparing algorithms, due to the fact that the relevant obesity genes are much fewer than the total genes in the base set, in the obesity data set experiments, the following metrics are defined for the evaluation. The *selectivity* (bigger is better) of an algorithm is a ratio $\frac{p_1}{p_2}$ where p_2 is the proportion of genes in the base set that have obesity annotation, and p_1 is the proportion of genes in the set returned by the algorithm that are known to have obesity annotation. The *recall* of the algorithm is a ratio $\frac{n_1}{n_2}$ where n_2 is the total number of obese genes in the base set, and n_1 is the total number of the obese genes in the set returned by the algorithm.

The selectivity and recall of RSSMMC are compared with that of the randomized method, SVM (using 10 fold cross validation) and p-value method (Note the main difference between RSSMMC and SVM method is that RSSMMC uses the maximum margin criterion to rank the genes while SVM only bases its ranking results on the classification performance). For randomized method, the same number of genes as returned by the p-value method is randomly selected from the base gene set. For SVM and RSSMMC, the top ranked genes with the same size as selected by p-Value method are selected from the base gene set. By ranking the same number of genes, the four algorithms are compared equally. The result is shown in Table 4.8. These data are then plotted in Figure 4.9 and Figure 4.10.

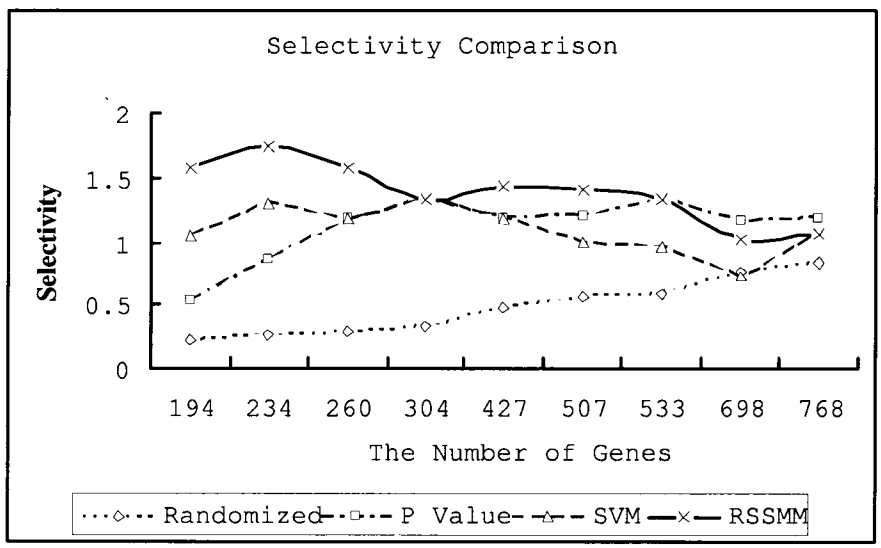

Figure 4.9: The Comparison of Selectivity between Randomized Selection, P-value, SVM, and RSSMMC

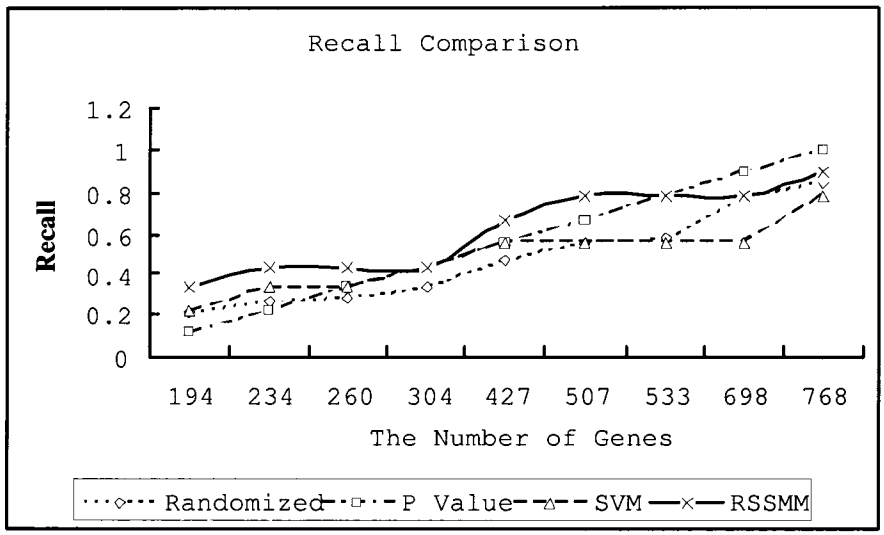

Figure 4.10: The Comparison of Recall between Randomized Selection, P-value, SVM, and RSSMMC

These figures show that the RSSMMC algorithm performs better in 7 out of the 9 obese genes than the p-value method while universally outperforms the SVM and randomized method.

Recall (Section 3.2.3) that to some point, when the normalization suppressor r decreases, more genes will be selected by RSSMMC. This is because a larger normalization suppressor makes it harder for a new gene to satisfy the selection condition. In Figure 4.11, this trend can be observed, i.e., both the best selectivity and the best recall are achieved when the normalization suppressor equals to 0.494. Examination of the relevant genes discovered at the normalization suppressor 0.494 includes annotated obesity-relevant genes, IRS2, ATRN, and HTR1B.

Table 4.8: The Selectivity and Recall Comparisons between Randomized method, P-value, SVM and RSSMMC

Obese Gene	Randomized Method		P-value		SVM		RSSMMC	
	Selectivity	Recall	Selectivity	Recall	Selectivity	Recall	Selectivity	Recall
$1st$	0.21	0.21	0.53	0.11	1.05	0.22	1.58	0.33
$2nd$	0.26	0.26	0.87	0.22	1.31	0.33	1.74	0.44
$3rd$	0.28	0.28	1.18	0.33	1.18	0.33	1.57	0.44
$4th$	0.33	0.33	1.34	0.44	1.34	0.44	1.34	0.44
$5th$	0.47	0.47	1.19	0.56	1.19	0.56	1.43	0.67
$6th$	0.55	0.55	1.21	0.67	1.00	0.56	1.41	0.78
$7th$	0.58	0.58	1.34	0.78	0.96	0.56	1.34	0.78
$8th$	0.76	0.76	1.17	0.89	0.73	0.56	1.02	0.78
$9th$	0.84	0.84	1.19	1.00	1.06	0.78	1.06	0.89

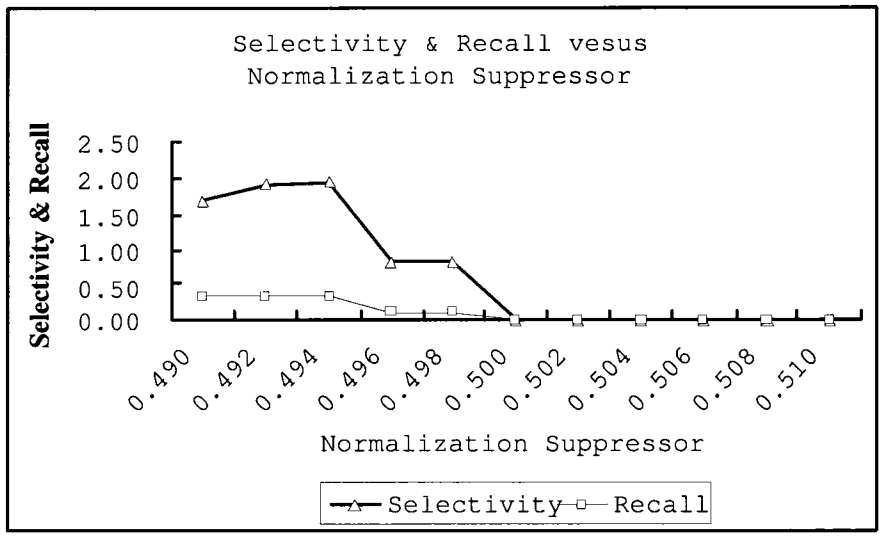

Figure 4.11: The Selectivity Curve over Different Normalization Suppressors by RSS-MMC

4.4.1.2 Selectivity Comparison when Recall is Given

It is also interesting to compare different methods in terms of one metric when the other metric is fixed. In this section, RSSMMC is compared with p-value method for selectivity when recalls are given.

Given a number i of obese genes, we can find the shortest prefixes from both HC-list of Algorithm-2 in section 4.4.1 and p-value list that contain i obese genes (This renders in giving a recall value). We then compare the lengths of both prefixes. A shorter length signifies a higher selectivity for the given i. The result is depicted in Figure 4.12. From the figure, it can be seen that the RSSMM algorithm outperforms p-value method in the first 7 of the 9 obese genes.

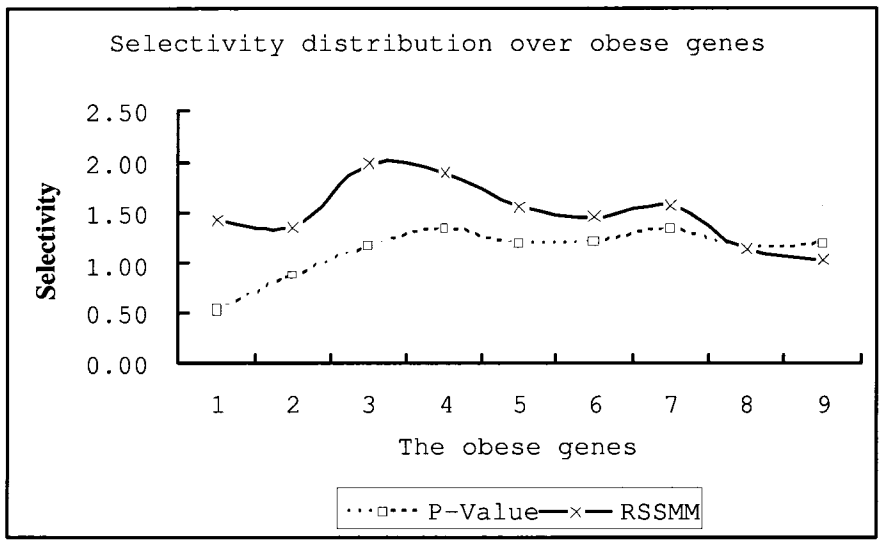

Figure 4.12: The Comparison of the Position of Obese Genes between P-value and RSSMM

4.4.2 RSSMMC-GA Experiments

In this section, The research is expanded by applying a GA version of the RSSMMC algorithm on the obesity data set. In GA, the string of chromosomes is obtained by representing the selected gene in the obese function with 1 and 0 otherwise. The chromosome string hence represents the collaborative functioning of the genes in the obesity metabolic pathway. The fitness function is the maximum margin in the feature space after removing the influence of the mathematical factor, i.e.,

$$f = \frac{Maximum\ Margin}{D^r} \tag{4.1}$$

where D is the number of dimensions in the feature space and $Maximum\ Margin$ is the maximum margin value that can be reached by a chromosome in the GA population in one generation, and r is the normalization suppressor. We already

indicated previously that the maximum number of SVM calculation (this is equivalent to the fitness value used in RSSMMC-GA) is $\frac{n.(n+1)}{2}$ and thus its time complexity is $O(n^2)$. In RSSMMC-GA, the number of fitness evaluation is much larger than RSSMMC, depending on the population size and the number of generation settings. For example, n is 917 in the obesity data set for RSSMMC and the time complexity is 917^2 (this is the worst case) whereas the time complexity for RSSMMC-GA is 3000×5000 when the population size equals 3000 and the generation number equals 5000.

Since the size of the search space, 2^{917}, is huge, a relatively big GA population size, 3000, is adopted in the final experiment. 90% are selected as the crossover rate and 0.1% as the mutation rate. Figure 4.13 shows the maximum margin distribution across the generations by RSSMMC-GA. A close look at the resulting subset discovers that

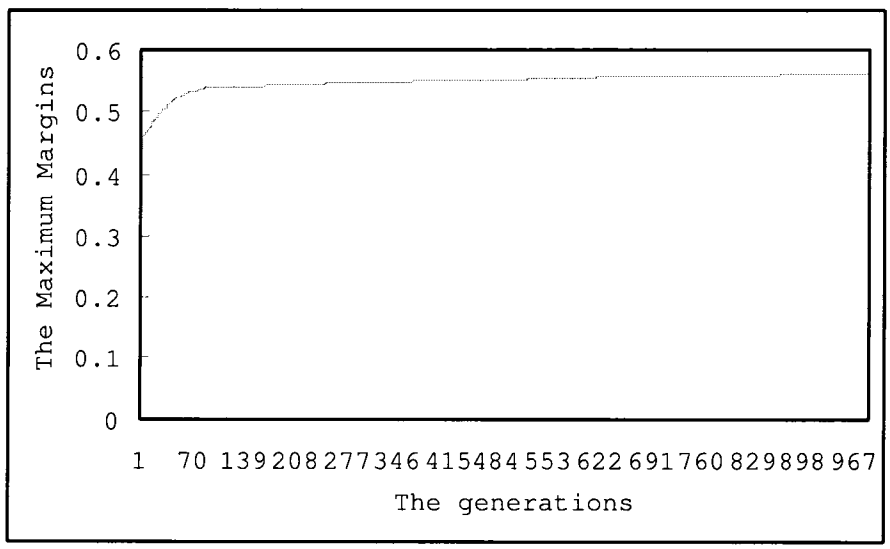

Figure 4.13: The Maximum Margin Distribution across the Generations

all the three obesity-relevant genes in the subset created by the RSSMMC algorithm, namely, IRS2, ATRN, and HTR1B, are also in the subset generated by the RSSMMC-GA implementation. To verify that this result is not obtained by chance, experiments with different configurations of GA parameters have been implemented. The result is shown in Table 4.9. The rows from the $2nd$ to the $5th$ in the Table 4.9 are the necessary GA parameters. The sixth row lists the obese genes found in the subset. The seventh row lists the number of genes in the subset. The eighth row lists the maximum fitness in each implementation. The three-digit entries from row 9 to row 17 are the positions of the obesity-relevant genes in the p-value ranking list. The obesity-relevant genes, IRS2, ATRN and HTR1B (Obese Gene 1, 2, and 3 in Table 4.7, respectively), occur in all the experiments. The parenthesized numbers are those that have one bit difference to the obesity-relevant genes in the p-value list and have been identified by RSSMMC-GA. These neighboring genes also exhibit importance since their expression levels are close to the obesity relevant genes. One possible reason is that, due to the systematic error in microarray experiments, one obese gene might not show as strong over/under-expressed values as its neighboring genes do. Note that when 273 genes are included in the subset, 6 out of the 9 obese genes are discovered. For the 3 that are not located, their closest neighbors are identified. Moreover, when a feature subset as small as 84 was achieved in Test2 in Table 4.9, RSSMMC-GA still locates the 3 mentioned obese genes in the relevant gene subset correctly.

Table 4.9: The Results of GA Implementations with Different Configurations

	Test1	Test2	Test3	Test4	Test5	Test6	Test7	RSSMMC
Population Size	800	1000	3000	2000	800	800	3000	
Generation Number	1000	2000	2000	5000	800	1800	988	
Crossover Rate	90%	90%	90%	70%	95%	90%	90%	
Mutation Rate	0.1%	0.1%	0.1%	0.1%	0.5%	0.5%	0.1%	
Obese genes found	3	3	3	2	6	5	3	
# of Genes in the subset	176	84	105	89	273	248	159	159
The Maximum fitness	0.562	0.575	0.572	0.573	0.552	0.554	0.563	0.558
Obese Gene 1	194	194	194	194	194	194	194	194
Obese Gene 2					234	234		
Obese Gene 3								
Obese Gene 4	(305)			(305)	(305)	304		
Obese Gene 5					427	(426)		
Obese Gene 6	507	507	507	507	507	507	507	507
Obese Gene 7	533	533	533	533	533	533	533	533
Obese Gene 8	(699)	(699)	(699)	(699)	(699)	698	(699)	
Obese Gene 9			(767)		768		(769)	

4.4.3 The Analysis of the Results Generated from both RSSMMC and RSSMMC-GA

The experiments discover that three obesity-relevant genes, IRS2, ATRN, and HTR1B, show strongly differentiated expression levels between obese and lean samples.

In [39], HTR1B (human serotonin receptor 1B) is found to be associated with minimum lifetime body mass index in women with bulimia nervosa. In [51], the IRS2 (the insulin-receptor substrate 2) gene at amino acid 1057 from Glycine to Asparaginic acid ($G1057D$) allele was shown to increase the risk of insulin resistance among obese individuals. Specifically, type 2 diabetic patients, particularly obese patients, carrying the $D1057$ allele and the CA haplotype were associated with insulin resistance, which is strongly correlated with obesity. As indicated in [21], ATRN has multiple variants, three of which have been characterized and found to encode different isoforms. One of the isoforms is a membrane-bound protein with sequence similarity to the mouse mahogany protein, a receptor involved in controlling obesity.

The reason that the other six annotated obesity-relevant genes are not included in the subset, when a small feature subset (e.g. 84 in Test 2 in Table 4.9) is found, is considered to be twofold. First, they may have secondary or indirect effects on the obese function and hence their contributions on the margin increase are not obvious. Second, the gene expression profiles in the experiments are obtained solely from abdomen adipose tissue and only very few obese genes are under/over expressed in adipose issue to be detectable by computational methods. In fact, the obese genes can be expressed in many different tissues with different levels. It is very hard to determine what all the genes recognized as obesity-related in abdomen adipose tissue

are at the time the thesis is written. To be more specific, other obesity-relevant genes can be either not expressed in abdomen adipose tissue or the expression level is not distinct enough to pass the selection condition set by the normalization factor, and thus the gene expression may be biased.

Chapter 5

Conclusions

When some information about the research objects (e.g. the class label of each sample) is known in advance, supervised learning methods often show better performance than unsupervised ones. SVM is a very effective supervised learning technology for the binary classification task, especially when the samples express high dimensional features. In other words, SVM can still perform well in many cases while other supervised methods lack effectiveness due to the overlapping of target classes. This is because SVMs have a nice property to map the features from a low dimensional space to a high dimensional space using the kernel function, as described in chapter 2. On the other hand, since some genes in the real microarray data dominate the classification process while others have either secondary or no effects on the classification, rather than treating all genes equally, the genes that contribute most to the classification should be considered more carefully. Moreover, instead of identifying the difference of genetic expression profiles between two groups, it is more important to find a group of genes which contribute maximally to the group separation. If the

contribution of a set of genes to the separation is to be evaluated, the margin is a good criterion to work with. That is, in general case, the larger the margin, the better the separation.

This thesis introduces a new margin-based hill-climbing method RSSMMC to search relevant genes, given the labeled training data samples. This method establishes a connection between each gene's contribution on the separation of two different groups and the biological differences among the genes. Besides the gene expression data, this method does not require additional information for the searching process. The degree to which each gene is relevant to the class separation is ranked according to its contribution to the increase of the margin between the two groups. Due to the influence of the mathematical factor, described in Chapter 3, the margin value always increases, if at all, whenever a new gene is included in the relevant subset. This thesis presents an analytic method to remove the influence of the mathematical factor and expands the solution to the feature space when the nonlinear SVM is under consideration. The RSSMMC method is implemented iteratively and includes a gene, one at a time, in the relevant gene subset until the maximum margin value is achieved. When the iterative process finishes, a subset of ranked genes is generated.

After RSSMMC has been implemented on a simulated data set to exhibit its ability on locating the relevant features, two real-world data sets were then analyzed using RSSMMC. In the leukemia data set, RSSMMC shows better performance than both the comparing methods, the baseline method and RFE. Specifically, RSSMMC only uses 2 genes to achieve the same classification performance as that of RFE which uses 8. With much fewer features, both RSSMMC and RFE have a less number of classification errors than the baseline method. While RSSMMC shows similar

classification results as that of RFE, it produces a different gene subset from what RFE generates. Comparing with RFE, the RSSMMC method has a different ranking order and composition of the generated relevant genes, although the top ranked genes from the two methods have some overlapping. The difference deserves more research since the output genes from RSSMMC that are not in the RFE top list are all over or under-expressed in either type of the samples in the microarray experiments. In fact, all of them have been summarized to have close relations to the leukemia function (See Chapter 4 for details). Moreover, unlike the nested subset generated RFE, RSSMMC generates a fixed number of relevant gene subset when the maximum margin value is achieved. In addition, RSSMMC starts from zero size subset and expand it to the point where the maximum margin value is obtained while RFE starts from a full sized subset and decreases the subset in each iteration. Since usually a small number of genes of the data set are included in the relevant gene subset, the RSSMMC method is much faster than the basic RFE in the implementation and slightly faster than the improved version of RFE. RSSMMC also provides more explainable results comparing with the improved version of RFE.

The obesity data set is much harder to process than the leukemia one since it has many missing values and outliers. Furthermore, although not all obesity-relevant genes are over or under-expressed in the abdomen adipose tissue, abdomen is the only place the samples were collected. Two metrics, the selectivity and recall, have been defined to measure the capability of obesity-relevant gene location. The RSSMMC method shows better performance than the p-value, SVM, and randomized method in both metrics. It also includes 3 annotated obesity-relevant genes, $HTR1B$, $IRS2$, and $ATRN$ in the relevant gene subset. To investigate other possible gene subsets,

a GA version of RSSMMC, RSSMMC-GA, is applied to search the solutions in the solution space. With different settings of parameters, the RSSMMC-GA algorithm discovered the 3 obesity-relevant genes consistently. Since these genes are also in the subset generated by the RSSMMC method, it suggests that RSSMMC can effectively find a good solution to approximate the global maximum in terms of the maximum margin value, is enhanced.

We have discussed the RSSMMC algorithm in the context of binary classification throughout this thesis. Nevertheless, it is foreseeable that RSSMMC can be applied to data with more than two classes with minor modifications. One possible solution is, by applying RSSMMC to some selected paired classes iteratively until all the feature subsets have been generated, the features with the highest appearance frequencies across all the subset can be selected to construct the final subset. This deserves more research work in the future.

Appendix A

Basic Knowledge of Molecular Biology and the General Procedure of a Microarray Experiment

Although varying in the size and shape, cells that contain biological structural features constitute basic building blocks of life. Various molecules (e.g. *Proteins, DNAs, and RNAs*) perform different functions in cells. A protein is a large molecule composed of one or more chains of amino acids linked together in particular orders. Proteins perform a wide variety of functions in the cell, including serving as enzymes, structural components, or signaling molecules and regulating the body's cells, tissues, and organs. A DNA (Deoxyribonucleic Acid) sequence encodes complete genetic information to synthesize proteins. Protein synthesis consists of three stages which are transcription, splicing and translation [15]. A strand of DNA molecule in nucleus is transcribed to an *mRNA* (messenger Ribonucleic Acid) and then the mRNA is

translated to protein in cytosol. The mRNA is a critical intermediary link in protein synthesis. Its expression levels in general can reflect a quantification of protein synthesis. In fact, the gene expression level is believed to be correlated with the approximate number of copies of peptides produced in a cell. Other than traditional genetic and molecular approaches, which usually examine and collect data on a single gene, microarray techniques monitor the expression pattern of tens of thousands of genes in parallel [12][55].

Generally, a microarray is a collection of microscopic DNA spots attached to a solid surface, such as glass, plastic or silicon chip forming an array, onto which single-stranded DNA molecules are attached at fixed locations (spots), where each spot relates to a DNA sequence. The mRNA samples (or targets) are reverse-transcribed into cDNA, labeled with different fluorescent dyes (e.g. a red-fluorescent dye Cy5 and a green-fluorescent dye Cy3), then mixed and hybridized with the arrayed DNA sequences (or probes). The process of joining two complementary strands of DNA or one each of DNA and RNA to form a double-stranded molecule is call hybridization. After the competitive hybridization, the relative abundance of those spotted DNA sequences in two mRNA samples can be assessed by testing the two differential hybridizations to the sequences on the array. The slides are then imaged using a scanner. Fluorescence measurements are made separately for each dye at each spot on the array [30]. Some other hybridization-based high-throughput methods exist to measure the mRNA levels such as Oligonucleotide chips, SAGE (Serial Analysis of Gene Expression) [17]. Data collected by these methods are noted as gene expression data. In some cases, these raw data need preprocessing such as normalization and noise reduction before they are used for further analysis.

Appendix B

The SVM Solution in Non-separable Case

The SVM problem in non-separable case can be described in Equation 2.15. The separable case now corresponds to $\gamma = \infty$. Thus, the construction procedure of the Lagrangian is

$$L_P = \frac{1}{2} + \gamma \sum_{i=1}^{N} \xi_i - \sum_{i=1}^{N} \alpha \left[y_i(x_i^T \beta + \beta_0) - (1 - \xi_i) \right] - \sum_{i=1}^{N} \mu_i \xi_i. \tag{B.1}$$

The constraint $\xi_i \geq 0$ is represented by the third term $\sum_{i=1}^{N} \mu_i \xi_i$ while the constraint $y_i(x_i^T \beta + \beta_0) \geq 1 - \xi_i$ is represented in $\sum_{i=1}^{N} \alpha \left[y_i(x_i^T \beta + \beta_0) - (1 - \xi_i) \right]$. L_P can now be minimized with respect to β, $\beta_0, and\ \xi_i$ by setting their respective derivatives to zero. After simplification,

$$\beta = \sum_{i=1}^{N} \alpha_i y_i x_i \tag{B.2}$$

$$0 = \sum_{i=1}^{N} \alpha_i y_i \tag{B.3}$$

$$\alpha_i = \gamma - \mu_i \quad \forall i \tag{B.4}$$

are obtained and the positive constraints $\alpha_i, \mu_i, \xi_i \geq 0$ hold. Substituting Equations B.2, B.3, and B.4 into B.1, the Lagrangian Wolfe dual

$$\text{L}_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j x_i^T x_j \tag{B.5}$$

is achieved. Recalling the KKT conditions, to maximize L_D, in addition to Equations B.2, B.3, and B.4,

$$\alpha_i \left[y_i (x_i^T \beta + \beta_0) - (1 - \xi_i) \right] = 0 \tag{B.6}$$

$$\mu_i \xi_i = 0 \tag{B.7}$$

$$y_i (x_i^T \beta + \beta_0) - (1 - \xi_i) \geq 0 \tag{B.8}$$

for $i = 1, \cdots, N$ need to be satisfied as well. From Equation B.2, The solutions of β are observed to have the form $\hat{\beta} = \sum_{i=1}^{N} \hat{\alpha}_i y_i x_i$. Only those data points that satisfy Equation B.6 have nonzero coefficients α_i. These data points are support vectors. From Equation B.6, when $\hat{\xi}_i = 0$, the support vectors are known to lie on the margin. Consider Equation B.4 and B.7. These coefficients satisfy $0 \leq \alpha_i \leq \gamma$. When $\hat{\xi}_i > 0$, the coefficients satisfy $\hat{\alpha}_i = \gamma$. One interesting point is that β_0 can be obtained from any data point on the margin which satisfies $\hat{\xi}_i = 0$. Practically, the mean of all the solutions can be taken to stabilize the results. Same as in the separable case, the classification function is

$$\hat{G}(x) = sign \left[x^T \hat{\beta} + \hat{\beta}_0 \right]. \tag{B.9}$$

Appendix C

The Sequential Minimal Optimization Algorithm

The main advantages of SMO include fast kernel evaluation speed based on its minimal (only two samples are evaluated each time) computation and good scalability due to its asymptotic calculation manner.

As stated in Chapter 2, the goal of the SVM algorithm is to solve the Quadratic Programming (QP) problem in order to minimize the dual objective function f_D, i.e.,

$$\begin{aligned} \min f_D = \min \tfrac{1}{2} \textstyle\sum_{i=1}^{N} \sum_{j=1}^{N} \mathbf{y}_i \mathbf{y}_j K(\vec{\mathbf{x}}_i, \vec{\mathbf{x}}_j) \alpha_i \alpha_j - \sum_{i=1}^{N} \alpha_i, \\ 0 \leq \alpha_i \leq C, \forall i, \\ \textstyle\sum_{i=1}^{N} \mathbf{y}_i \alpha_i = 0. \end{aligned} \tag{C.1}$$

Also the KKT conditions in Equation C.2

$$\begin{aligned} \alpha_i = 0 \Leftrightarrow \mathbf{y}_i \mathbf{u}_i \geq 1, \\ 0 < \alpha_i < C \Leftrightarrow \mathbf{y}_i \mathbf{u}_i = 1, \\ \alpha_i = C \Leftrightarrow \mathbf{y}_i \mathbf{u}_i \leq 1 \end{aligned} \tag{C.2}$$

must be satisfied to make the QP problem positive definite.

The major difference between SMO and other existing SVM algorithms is that SMO solves the smallest possible optimization problem, which involves two Lagrange multipliers, at every step. The reason that at least two Lagrange multipliers must be involved lies on a so-called linear inequality constraint in Equation C.1. The inequality constraints cause the Lagrange multipliers to be limited within a rectangle box, while the linear equality constraint causes the Lagrange multipliers to lie on a diagonal line. This is demonstrated in Figure C.1 for two Lagrange multipliers. Therefore, at least two Lagrange multipliers are needed to satisfy the linear equality constraint at every step. In fact, given $\omega(\alpha) = \sum_i \alpha_i y_i z_i$, the Lagrangian for the dual

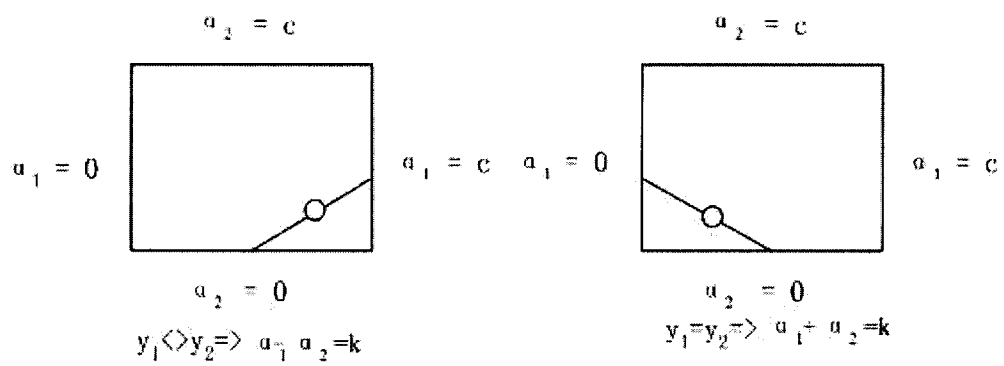

Figure C.1: The Two Lagrange Multipliers must Satisfy the Constraints: $k = \alpha_1^{old} + s\alpha_2^{old}$ *and* $s = y_1 y_2$

in Equation C.1 can be written as

$$\overline{L} = \frac{1}{2}\omega(\alpha)\omega\alpha - \sum_i \alpha_i - \sum_i \delta_i \alpha_i + \sum_i \mu_i(\alpha_i - C) - \beta \sum_i \alpha_i y_i. \qquad \text{(C.3)}$$

Setting

$$F_i = \omega(\alpha).z_i - y_i = \sum_j \alpha_j y_i k(x_i, x_j) - y_i, \tag{C.4}$$

taking the partial differential of $\overline{L}$ over α_i and considering the KKT conditions, the KKT conditions is simplified to three cases

$$\begin{aligned} &Case1.\ \alpha_i = 0: && \delta_i \geq 0, \mu_i = 0 \Rightarrow (F_i - \beta)y_i \geq 0 \\ &Case2.\ 0 < \alpha_i < C: && \delta_i = 0, \mu_i = 0 \Rightarrow (F_i - \beta)y_i = 0 \\ &Case3.\ \alpha_i = C: && \delta_i = 0, \mu_i \geq 0 \Rightarrow (F_i - \beta)y_i \leq 0. \end{aligned} \tag{C.5}$$

For all samples, Define five index sets:

$I_0 := \{i : 0 < \alpha_i < C\}$;

$I_1 := \{i : y_i = 1, \alpha_i = 0\}$;

$I_2 := \{i : y_i = -1, \alpha_i = C\}$;

$I_3 := \{i : y_i = 1, \alpha_i = C\}$;

$I_4 := \{i : y_i = -1, \alpha_i = 0\}$.

Conditions in Equation C.5 can be written as

$$\beta \leq F_i \ \forall i \in I_0 \cup I_1 \cup I_2 \ and \ \beta \geq F_i \ \forall i \in I_0 \cup I_3 \cup I_4. \tag{C.6}$$

If define

$$b_{up} = \min\{F_i : i \in I_0 \cup I_1 \cup I_2\} \ and \ b_{low} = \max\{F_i : i \in I_0 \cup I_3 \cup I_4\}, \tag{C.7}$$

the optimality conditions will hold at some α if and only if

$$b_{low} \leq b_{up}. \tag{C.8}$$

In numerical solution, optimality usually cannot be obtained exactly. Therefore, a tolerance parameter τ is added and Equation C.8 can be rewritten as

$$b_{low} \leq b_{up} + 2\tau. \tag{C.9}$$

Platt [52] selected β to be placed halfway between b_{low} and b_{up}. In this case, Equation C.5 with τ is

$$\begin{aligned} (F_i - \beta)y_i \geq -\tau \ if \ \alpha_i = 0 \\ |(F_i - \beta)| \leq \tau \ if \ 0 < \alpha_i < C \\ (F_i - \beta)y_i \leq \tau \ if \ \alpha_i = C \end{aligned} \cdot \tag{C.10}$$

As claimed in [33], simply placing β halfway between b_{low} and b_{up} will be inefficient; in particular, in some circumstances it will prompt some violation of the optimality criterion even though there is no violation at all. In other words, in the original version of the SMO algorithm, it is possible that SMO cannot detect an optimized α due to the incorrect choice of β. Therefore, Keerthi et al suggested two modified implementations of the original SMO algorithm [33]. The main idea is that rather than using a single threshold value β and Equation C.10 for the optimality examination, the modifications maintain two threshold parameters, b_{low} and b_{up}, and use Equation C.9 to check optimality. The RSSMMC and its GA version RSSMMC-GA algorithms implement the first version of modification in this thesis.

Appendix D

Top 50 Genes from Leukemia Data Set by RSSMMC

Table D.1: The 50 Top Ranked genes by RSSMMC (1-17)

Rank	Gene Description	GAN
1	Terminal transferase mRNA	M11722_at
2	14-3-3 PROTEIN TAU	X56468_at
3	TCF3 Transcription factor 3 (E2A immunoglobulin enhancer binding factors E12/E47)	M31523_at
4	CD19 gene	M84371_rna1_s_at
5	LYN V-yes-1 Yamaguchi sarcoma viral related oncogene homolog	M16038_at
6	Nucleoside-diphosphate kinase	Y07604_at
7	ATP6C Vacuolar H+ ATPase proton channel subunit	M62762_at
8	Interferon-gamma induced protein (IFI 16) gene	M63838_s_at
9	"LMP2 gene extracted from H.sapiens genes TAP1, TAP2, LMP2, LMP7 and DOB"	X66401_cds1_at
10	Transcriptional activator hSNF2b	U29175_at
11	Zyxin	X95735_at
12	TOP2B Topoisomerase (DNA) II beta (180kD)	Z15115_at
13	MB-1 gene	U05259_rna1_at
14	"SPTAN1 Spectrin, alpha, non-erythrocytic 1 (alpha-fodrin)"	J05243_at
15	Dihydropyrimidinase related protein-2	U97105_at
16	CCND3 Cyclin D3	M92287_at
17	"C-myb gene extracted from Human (c-myb) gene, complete primary cds and five complete alternatively spliced cds"	U22376_cds2_s_at

Table D.2: The 50 Top Ranked genes by RSSMMC (18-34)

Rank	Gene Description	GAN
18	CD19 CD19 antigen	M28170_at
19	MHC-encoded proteasome subunit gene LAMP7-E1 gene (proteasome subunit LMP7) extracted from H.sapiens gene for major histocompatibility complex encoded proteasome subunit LMP7	Z14982_rna1_at
20	Macmarcks	HG1612-HT1612_at
21	INTERLEUKIN-8 PRECURSOR	Y00787_s_at
22	VIL2 Villin 2 (ezrin)	X51521_at
23	RABAPTIN-5 protein	Y08612_at
24	Putative chloride channel	X83378_at
25	TTF mRNA for small G protein	Z35227_at
26	IL7R Interleukin 7 receptor	M29696_at
27	Interleukin 8 (IL8) gene	M28130_rna1_s_at
28	PROBABLE G PROTEIN-COUPLED RECEPTOR LCR1 HOMOLOG	L06797_s_at
29	"ARHG Ras homolog gene family, member G (rho G)"	X61587_at
30	26-kDa cell surface protein TAPA-1 mRNA	M33680_at
31	Transcriptional activator hSNF2b	D26156_s_at
32	C-myc binding protein	D89667_at
33	GPX1 Glutathione peroxidase 1	Y00433_at
34	BB1	S82470_at

Table D.3: The 50 Top Ranked genes by RSSMMC (35-50)

Rank	Gene Description	GAN
35	"FTL Ferritin, light polypeptide"	M11147_at
36	MitF mRNA	Z29678_at
37	Azurocidin gene	M96326_rna1_at
38	GLYCYLPEPTIDE N-TETRADECANOYLTRANSFERASE	U79285_at
39	"CYBA Cytochrome b-245, alpha polypeptide"	M21186_at
40	ADPRT ADP-ribosyltransferase (NAD+; poly (ADP-ribose) polymerase)	J03473_at
41	Mac25	HG987-HT987_at
42	FOS-RELATED ANTIGEN 2	X16706_at
43	"NFYA Nuclear transcription factor Y, alpha"	X59711_at
44	OS-9 precurosor mRNA	U41635_at
45	UBIQUITIN-LIKE PROTEIN GDX	J03589_at
46	"MEF2A gene (myocyte-specific enhancer factor 2A, C9 form) extracted from Human myocyte-specific enhancer factor 2A (MEF2A) gene, first coding"	U49020_cds2_s_at
47	GLRX Glutaredoxin (thioltransferase)	X76648_at
48	Skeletal muscle LIM-protein SLIM1 mRNA	U60115_at
49	MANA2 Alpha mannosidase II isozyme	L28821_at
50	"DAGK1 Diacylglycerol kinase, alpha (80kD)"	X62535_at

Bibliography

[1] M.A. Aizerman, E.M. Braveman, and L.I. Rozoner. Theoretical foundations of the potential function method in pattern recognition learning. *Automation and Remote Control*, 25:821–837, 1964.

[2] S. Alkan and D.S. Karcher. Indolent lymphomas: Classic subtypes and newer entities. *Cancer control Journal*, 3:152–157, 1996.

[3] H. Almuallim and T.G. Dietterich. Learning with many irrelevant features. In *Proceedings of the Ninth National Conference on Artificial Intelligence*, volume 2, pages 547–552, 1991.

[4] E. Amaldi and V. Kann. On the approximability of minimizing nonzero variables or unsatisfied relations in linear systems. *Theoretical Computer Science*, 209(1-2):237–260, 1998.

[5] A. Ben-Dor, L. Bruhn, N. Friedman, I. Nachman, M. Schummer, and Z. Yakhini. Tissue classification with gene expression profiles. *Journal of Computational Biology*, 7(3-4):559–583, 2000.

[6] A. Blum and P. Langley. Selection of relevant features and examples in machine learning. *Artificial Intelligence*, 97(1-2):245–271, 1997.

[7] T. Bo and I. Jonassen. New feature subset selection procedures for classification of expression profiles. *Genome biology*, 3:RESEARCH0017.1–0017.11, 2002.

[8] F. Brambillasca, G. Mosna, E. Ballabio, A. Biondi, K.E. Boulukos, and E. Privitera. Promoter analysis of tfpt (fb1), a molecular partner of tcf3 (e2a) in childhood acute lymphoblastic leukemia. *Biochemical and biophysical research communications*, 288(5):1250–7, 2001.

[9] V. Buccheri, B. Mihaljevic, E. Matutes, M.J. Dyer, D.Y. Mason, and D. Catovsky. mb-1: a new marker for b-lineage lymphoblastic leukemia. *Blood*, 82:853–857, 1993.

[10] C. J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167, 1998.

[11] S. Cang and D. Partridge. Feature ranking and best feature subset using mutual information. *Neural Computing and Applications*, 13(3):175–184, 2004.

[12] J.J. Chen, R. Wu, P.C. Yang, J.Y. Huang, Y.P. Sher, M.H. Han, W.C. Kao, P.J. Lee, T.F. Chiu, F. Chang, Y.W. Chu, C.W. Wu, and K. Peck. Profiling expression patterns and isolating differentially expressed genes by cdna microarray system with colorimetry detection. *Genomics*, 51:313–324, 1998.

[13] R. Courant and D. Hilbert. *Methods of Mathematical Physics.* Interscience, New York, 1953.

[14] M. Dash and H. Liu. Feature selection for classification. *Intelligent Data Analysis*, 1(3):131–156, 1997.

[15] K. Deb and A. Raji Reddy. Reliable classification of two-class cancer data using evolutionary algorithms. *BioSystems*, 72:111–129, 2003.

[16] P.A. Devijver and J. Kittler. *Pattern Recognition: A Statistical Approach.* Prentice Hall International, London, 1982.

[17] P. Dhaeseleer, S. Liang, and R. Somogyi. Tutorial: Gene expression data analysis and modeling. In *Proceedings of the Pacific Symposium on Biocomputing*, 1999.

[18] G.V. Dijck, M.M.V. Hulle, and M. Wevers. Genetic algorithm for feature subset selection with exploitation of feature correlations from continuous wavelet transform: a real-case application. *Journal of Computational Intelligence*, 1(1):1–12, 2004.

[19] J. Doak. Intrusion detection: The application of feature selection, a comparison of algorithms, and the application of a wide area network analyzer. Master's thesis, 1992.

[20] F.H. Drake, J.P. Zimmerman, F.L. McCabe, H.F. Bartus, S.R. Per, D.M. Sullivan, W.E. Ross, M.R. Mattern, R.K. Johnson, and S.T. Crooke. Purification of topoisomerase ii from amsacrine-resistant p388 leukemia cells. evidence for two forms of the enzyme. *Journal of Biological Chemistry*, 262(34):16739–16747, 1987.

[21] J.S. Duke-Cohan, J. Gu, D.F. McLaughlin, Y. Xu, G.J. Freeman, and S.F. Schlossman. Attractin (dppt-l), a member of the cub family of cell adhesion and guidance proteins, is secreted by activated human t lymphocytes and mod-

ulates immune cell interactions. *Proceedings of National Academy of Sciences of the United States of America*, 95(19):11336–11341, 1998.

[22] J. Faber, H. Kantarjian, M.W. Roberts, M. Keating, E. Freireich, and M. Albitar. Terminal deoxynucleotidyl transferase-negative acute lymphoblastic leukemia. *Archives of pathology and laboratory medicine*, 124:92–97, 2000.

[23] R. Fletcher. *Practical Methods of Optimization.* John Wiley and Sons, Inc., second edition, 1987.

[24] B.G. Gabrielsson, L.E. Olofsson, A. Sjogren, M. Jernas, A. Elander, M. Lonn, M. Rudemo, and L.M. Carlsson. Evaluation of reference genes for studies of gene expression in human adipose tissue. *Obesity Research*, 13(4):649–652, 2005.

[25] T.R. Golub, D.K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J.P. Mesirov, H. Coller, M. Loh, J.R. Downing, M.A. Caligiuri, C.D. Bloomfield, and E.S. Lander. Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. *Science*, 286(5439):531–537, 1999.

[26] Z. Guan and H.Y. Zhao. A semiparametric approach for marker gene selection based on gene expression data. *Bioinformatics: Original Paper*, 21:529536, 2005.

[27] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182, 2003.

[28] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. *Journal of Machine Learning Research*, 46(1-3):389–422, 2002.

[29] J.H. Holland. Outline for a logical theory of adaptive systems. *Journal of Association for Computing Machinery*, 3:297–314, 1962.

[30] D.X. Jiang and A.D. Zhang. Cluster analysis for gene expression data: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 16(11):1370–1386, 2004.

[31] G.H. John, R. Kohavi, and K. Pfleger. Irrelevant features and the subset selection problem. In *Proceedings of the Eleventh International Conference on Machine Learning*, pages 121–129, 1994.

[32] J. Kaput, K.G. Klein, E.J. Reyes, W.A. Kibbe, C.A. Cooney, B. Jovanovic, W.J. Visek, and G.L. Wolff. Identification of genes contributing to the obese yellow avy phenotype: caloric restriction, genotype, diet x genotype interactions. *Physiological Genomics*, 18:316 – 324, 2004.

[33] S. Keerthi, S. Shevade, C. Bhattacharyya, and K. Murthy. Improvements to platt's smo algorithm for svm classifier design. *Neural Computation*, 13:637–649, 2001.

[34] K. Kim, H. Thomsen, J. Bastiaansen, N.T. Nguyen, J.C.M Dekkers, G.S. Plastow, and M.F. Rothschild. Investigation of obesity candidate genes on porcine fat deposition quantitative trait loci regions. *Obesity Research*, 12(12):1981–1994, 2004.

[35] R. Kohavi and G.H. John. Wrappers for feature subset selection. *Journal of Artificial Intelligence Research*, 97(1-2):273–324, 1997.

[36] D. Koller and M. Sahami. Toward optimal feature selection. In *Proceedings of the 13th International Conference on Machine Learning*, pages 284–292, 1996.

[37] M. Kudo and J. Sklansky. Comparison of algorithms that select features for pattern classifiers. *Pattern Recognition*, 33:25–41, 2000.

[38] N. Kwak and C.H. Choi. Input feature selection by mutual information based on parzen window. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:1667–1671, 2002.

[39] R.D. Levitan, A.S. Kaplan, and M. Masellis. Polymorphism of the serotonin 5-ht1b receptor gene (htr1b) associated with minimum lifetime body mass index in women with bulimia nervosa biol psychiatry. *Medline*, 50:640–643, 2001.

[40] J. Loughrey and P. Cunningham. Overfitting in wrapper-based feature subset selection: The harder you try the worse it gets. In *Proceedings of the 24th SGAI International Conference on Innovative Techniques and Applications of Artificial Intelligence*, pages 33–43, 2004.

[41] G.F. Luger. *Artificial Intelligence, Structures and Strategies for Complex Problem Solving.* Addison-Wesley, London, fourth edition, 2002.

[42] T. Macalma, J. Otte, M.E. Hensler, S.M. Bockholt, H.A. Louis, M. Kalff-Suske, K.H. Grzeschik, D. Ahe, and M.C. Beckerle. Molecular characterization of human zyxin. *Journal of Biological Chemistry*, 271:31470–31478, 1996.

[43] C. MACKINTOSH. Dynamic interactions between 14-3-3 proteins and phosphoproteins regulate diverse cellular processes. *Biochemical Journal*, 381:329342, 2004.

[44] K.Z. Mao. Feature subset selection for support vector machines through discriminative function pruning analysis. *IEEE Transactions on Systems, Man and Cybernetics, Part B*, 34(1):60–67, 2004.

[45] O. Marek and S. Pavel. Visualization of genetic algorithms in a learning environment. In *Proceedings of the Spring Conference on Computer Graphics*, pages 101–106, 1999.

[46] D.R. Masys, J.B. Welsh, J.L. Fink, M. Gribskov, I. Klacansky, and J. Corbeil. Use of keyword hierarchies to interpret gene expression patterns. *Bioinformatics*, 17(4):319326, 2001.

[47] A. Miller. *Subset Selection in Regression.* Chapman and Hall, New York, 1990.

[48] P.M. Narendra and K. Fukunaga. A branch and bound algorithm for feature subset selection. *IEEE Transactions on Computers*, 26:917–922, Septemper 1977.

[49] S. Oba, M. Sato, I. Takemasa, M. Monden, K. Matsubara, and S. Ishii. A bayesian missing value estimation method. *Bioinformatics*, 19:2088–2096, 2003.

[50] J. Okabe-Kado, T. Kasukabe, Y. Honma, M. Hayashi, W.J. Henzel, and M. Hozumi. Identity of a differentiation inhibiting factor for mouse myeloid leukemia cells with nm23/ nucleoside diphosphate kinase. *Biochemical and biophysical research communications*, 182(3):987–94, 1992.

[51] K. Okazawa, Y. Yoshimasa, Y. Miyamoto, A. Takahashi-Yasuno, T. Miyawaki, H. Masuzaki, T. Hayashi, K. Hosoda, G. Inoue, and K. Nakao. The haplotypes of the irs-2 gene affect insulin sensitivity in japanese patients with type 2 diabetes. *Diabetes Research and Clinical Practice*, 68:39–48, 2005.

[52] J.C. Platt. Fast training of support vector machines using sequential minimal optimization. In *Advances in Kernel Methods: Support Vector Machines.* MIT Press, Cambridge, MA, 1998.

[53] J. Reunanen. Overfitting in making comparisons between variable selection methods. *Journal of Machine Learning Research*, 3:1371–1382, 2003.

[54] R.O.Duda and P.E. Hart. *Pattern Classification and Scene Analysis.* John Wiley and Sons, New York, 1973.

[55] M. Schena, D. Shalon, R. Heller, A. Chai, P.O. Brown, and R.W. Davis. Parallel human genome analysis: microarray-based expression monitoring of 1000 genes. In *Proceedings of the National Academy of Sciences of the United States of America*, volume 93, pages 10614–10619, 1996.

[56] W. Siedlecky and J. Sklansky. On automatic feature selection. *Pattern Recognition and Artificial Intelligence*, 2(2):197–220, 1988.

[57] V. Sindhwani, S. Rakshit, D. Deodhare, D. Erdogmus, J.C. Principe, and P. Niyogi. Feature selection in mlps and svms based on maximum output information. *IEEE Transactions on Neural Networks*, 15(4):937- 948, 2004.

[58] T. Sonoki, L. Harder, D.E. Horsman, L. Karran, I. Taniguchi, T.G. Willis, S. Gesk, D. Steinemann, E. Zucca, B. Schlegelberger, F. Sol, A.J. Mungall, R.D. Gascoyne, R. Siebert, and M.J.S. Dyer. Cyclin d3 is a target gene of t(6;14)(p21.1;q32.3) of mature b-cell malignancies. *Journal of Machine Learning Research*, 98(9):2837–2844, 2001.

[59] Z. Sun, X. Yuan, G. Bebis, and S. Louis. Genetic feature subset selection for gender classification: A comparison study. In *Sixth IEEE Workshop on Applications of Computer Vision*, page 165, 2002.

[60] V. Vapnik. *The Nature of Statistical Learning Theory.* Springer-Verlag,New York, 1995.

[61] V. Vapnik. *Statistical Learning Theory.* John Wiley and Sons, New York, 1998.

[62] D.F. Wang, P.P.K. Chan, D.S. Yeung, and E.C.C. Tsang. Feature subset selection for support vector machines through sensitivity analysis. In *Proceedings of the Third International Conference on Machine Learning and Cybernetics*, volume 7, pages 4257–4262, 2004.

[63] Jason Weston, Sayan Mukherjee, Olivier Chapelle, Massimiliano Pontil, Tomaso Poggio, and Vladimir Vapnik. Feature selection for svms. In *Neural Information Processing Systems*, pages 668–674, 2000.

[64] J. Yang and V. Honavar. Feature subset selection using a genetic algorithm. *IEEE Intelligent Systems*, 13:44–49, 1998.

[65] B. Yu and B. Yuan. A more efficient branch and bound algorithm for feature selection. *Pattern Recognition*, 26(6):883–889, 1993.