Gene selection for DNA microarray data

Zhou, Xin

2007

Zhou, X. (2007). Gene selection for DNA microarray data. Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/46974>

<https://doi.org/10.32657/10356/46974>

Nanyang Technological University

Downloaded on 26 Apr 2023 23:20:49 SGT

a778360

Gene Selection for DNA Microarray Data

Zhou Xin

School of Electrical & Electronic Engineering

A thesis submitted to the Nanyang Technological University
in fulfillment of the requirements for the degree of
Doctor of Philosophy

2007

QP
624.5
.D 726
Z63
2007

Statement of Originality

I hereby certify that the work embodied in this thesis is the result of original research and has not been submitted for a higher degree to any other University or Institution.

Mar. 14, 2007

Date

Zhou Xin

To my family

Acknowledgements

First of all, I would like to express my sincerest gratitude to my supervisor, Associate Professor Mao Kezhi, for his patient guidance, constructive comments and strong support to my research work. I really appreciate the time he spent on discussing the research projects, and sharing his insight and expertise with me. I can hardly count the help Dr. Mao gave to me in so many respects. In my heart, Dr. Mao is not only a nice supervisor, but also a close brother to me.

I am grateful to my friends, Mr. Li Jun, Mr. Liu Sheng, Mr. Wu Yilei, Mr. Zhang Runxuan, Ms. Zhao Peng, Ms. Tang Wenyin, Mr. Zhou Hao, Ms. Han Dan, Mr. Liu Chao, Mr. Diao Xiaoning, and all the lab mates for their encouragement and kind help. It is my pleasure of being studying and living with them. I believe our friendship will be an invaluable treasure throughout my life.

I would like to thank the Nanyang Technological University for providing me the scholarship to pursuit a PhD degree, and also letting me meet my wife here. I thank Bioinformatics Research Centre for providing the high performance computers for my experiments.

I am highly thankful to my family, especially my mother, for her continuous support and selfless love throughout my studies. I also would like to extend my deepest appreciation to my wife, Xingye, who offers her love, care, encouragement and help during the past two years.

Table of Contents

Acknowledgements	i
Table of Contents	iii
Summary	viii
List of Figures	xi
List of Tables	xvii
1 Introduction	1
1.1 Motivation	1
1.2 Objectives	3
1.3 Major Contribution of the Thesis	4
1.4 Organization of the Thesis	6
2 DNA Microarrays	8
2.1 Genetic Background	8

TABLE OF CONTENTS	iv
2.2 DNA Microarrays Technology	11
2.3 Microarray Data Analysis	15
2.3.1 Clustering of Microarray Data	19
2.3.2 Discriminant Analysis of Microarray Data	20
3 Literature Review of Gene Selection for DNA Microarray Data	22
3.1 Reduction of Dimensionality	23
3.2 Feature Selection	25
3.2.1 Evaluation Criterion	27
3.2.2 Search Procedure	28
3.2.2.1 Individual Ranking	29
3.2.2.2 Sequential Forward Selection (SFS)	30
3.2.2.3 Sequential Backward Selection (SBS)	31
3.2.2.4 Floating Search	32
3.3 Existing Gene Selection Methods	33
3.3.1 Fold Change	33
3.3.2 Individual Gene Ranking	33
3.3.2.1 Weighting Factor	34
3.3.2.2 t-statistic	34
3.3.2.3 Fisher's Ratio	35
3.3.2.4 Ratio of Between-group to Within-groups Sum of Squares	35

TABLE OF CONTENTS	v
3.3.3 Gene Pair Ranking	36
3.3.4 Gene Selection by GA/KNN	37
3.3.5 Gene Selection based on Support Vector Machines	37
3.3.5.1 Recursive Gene Ranking	38
3.3.5.2 SVM Recursive Feature Elimination (RFE)	38
3.3.5.3 SVM Feature Selection based on Discriminative Func- tion Pruning Analysis (DFPA)	39
3.4 Selection Bias	40
4 The Ties Problem Resulting from Counting-based Error Estimators and its Impact on Gene Selection Algorithms	44
4.1 Introduction	44
4.2 Classification Algorithms	46
4.2.1 Support Vector Machines	47
4.2.1.1 Linearly Separable Cases	47
4.2.1.2 Linearly Non-separable Cases	50
4.2.1.3 Non-linear Cases	53
4.2.2 k -Nearest Neighbors	54
4.3 Evaluation Criteria	55
4.3.1 Counting-based Error Estimation	55
4.3.2 Continuous Evaluation Criteria	58
4.4 System and Methods	61

TABLE OF CONTENTS	vi
4.4.1 Datasets	61
4.4.2 Wrapper Schemes	62
4.5 Experimental Studies on Microarray Datasets	63
4.6 Experimental Studies on UCI Data	86
4.7 Experimental Studies for Bolstered Error Estimator	90
5 LS Bound-based Gene Selection for DNA microarray data	95
5.1 Introduction	95
5.2 Least Squares Support Vector Machines	96
5.3 LS Bound Measure for Gene Selection	98
5.4 Efficient Computation of LS Bound Measure	102
5.5 Experimental Results	104
5.5.1 Colon Cancer Dataset	107
5.5.2 Breast Cancer Dataset	107
5.5.3 Leukaemia Dataset	111
5.6 Proof of Theorem 5.1	111
6 Regularization Network-based Gene Selection for Microarray Data Analysis	120
6.1 Introduction	120
6.2 Regularization Networks	121
6.3 RN-based Gene Selection	123

TABLE OF CONTENTS	vii
6.3.1 Algorithm	123
6.3.2 Selection of the Kernel	128
6.3.3 Efficient Computation of RN-based Criterion	130
6.4 Experimental Results	130
6.4.1 Colon Cancer Dataset	132
6.4.2 Breast Cancer Dataset	132
6.4.3 Leukaemia Dataset	135
7 Efficient Gene Selection Using Iterative Reduced Forward Search	139
7.1 Introduction	139
7.2 Support Vector Machines	140
7.3 SVM-based Criteria for gene selection	142
7.3.1 Generalized $\ \mathbf{w}\ ^2$ Measure	143
7.3.2 Generalized Radius/Margin Bound	143
7.4 Moving Bounds for SVM-based Criteria	145
7.4.1 Moving Bound for Generalized $\ \mathbf{w}\ ^2$ Measure	147
7.4.2 Moving Bound for Generalized Radius/Margin Bound	148
7.5 Iterative Reduced Search Algorithms	149
7.5.1 Iterative Reduced Search (IRS)	150
7.5.2 Iterative Reduced Forward Search (IRFS)	154
7.6 SVM Recursive Feature Elimination	155

TABLE OF CONTENTS	viii
7.7 Experimental Studies	158
7.8 Proof of Theorem 7.1	167
8 Conclusions and Recommendations	184
8.1 Conclusions	184
8.2 Recommendations for Further Research	187
Author's Publications	191
Bibliography	192
A. Lagrangian Optimization Theory	204

Summary

One problem with discriminant analysis of DNA microarray data is that each sample is represented by quite a large number of genes, and many of them are irrelevant, insignificant or redundant to the discriminant problem under investigation. Methods for selecting important genes are, therefore, of much significance in microarray data analysis. Gene selection can be cast as a feature selection problem in the context of pattern classification. Feature selection approaches can be divided into two broad groups, filter methods and wrapper methods. Generally speaking, the wrapper method provides features with better predictive accuracy than the filter method. In this thesis, wrapper-based gene selection is the focus.

We investigate the performances of several error counting-based wrapper methods, including the resubstitution error, leave-one-out error, cross-validation error and bootstrap error, in gene selection on the high dimensional and small-sized microarray data. Although error rate is widely used for feature selection, our experimental results show that discrete counting-based error estimators could encounter severe ties problem, *i.e.* more than one gene subset score equally, and this in turn results in uncertainty in gene selection. The main cause of the ties problem lies in the discrete estimators which lack sufficient resolution to differentiate gene subsets. Motivated by this finding, we suggest employing well-designed continuous evaluation criterion for gene subset evaluation. In the rest parts of this thesis, we developed three novel gene selection algorithms. They all employ continuous evaluation criteria to tackle this ties problem.

A new criterion, called LS Bound measure, is developed to address the gene selection problem. The LS Bound measure is derived from the leave-one-out procedure of LS-

SVM (least squares support vector machine). As an upper bound for leave-one-out classification results, it reflects to some extent the generalization performance of gene subsets. The strength of the LS Bound-based gene selection algorithm is that the obtained gene subsets can achieve more accurate classification results than the filter method does, while its computational complexity is at the level of the filter method.

A new gene selection algorithm based on Regularization Network (RN) is proposed. The evaluation criterion is derived from the leave-one-out procedure, but it does not demand repeated training in the leave-one-out procedure. Hence the computation cost in the gene selection algorithm is significantly reduced. Our experimental results show that the gene selection algorithm is effective and efficient for high dimensional microarray data.

Efficient gene selection algorithms, called SVM-IRFS, are proposed. Two evaluation criteria with respect to Support Vector Machines (SVM) are employed in this work. They are modified from well-known separating margin and radius/margin bound with applications to both separable and non-separable cases. We also develop a new search procedure, named Iterative Reduced Forward Search (IRFS). It relies on one very important property of the evaluation criterion, namely that for two feature subsets, A and A' , $J(A') \geq h(J(A), A, A')$. The right hand side of the inequality is called the moving bound of criterion $J(\cdot)$. Our experimental results show that IRFS is quite efficient to avoid unnecessary computations.

List of Figures

2.1	The DNA structure	9
2.2	The central dogma of molecular biology	10
2.3	Alternative splicing	11
2.4	cDNA microarray schema.	13
2.5	Oligonucleotide microarray schema.	14
2.6	Main steps in design and data analysis of a microarray experiment . .	16
2.7	A sample raw image from scanning a hybridized rat microarray which contains more than 5000 genes.	17
2.8	Illustration of gene expression matrix	18
3.1	Framework of typical feature selection algorithms	26
3.2	Illustration of selection bias and its correction	42
4.1	Optimal separating hyperplane	47
4.2	Linear separating hyperplane of SVM.	48
4.3	Illustration of the test procedure.	67

LIST OF FIGURES	xii
4.4 The test performance of error rate-based wrapper methods on the leukaemia dataset using SVM as classification algorithm and SFS as search procedure.	69
4.5 The test performance of error rate-based wrapper methods on the breast cancer dataset using SVM as classification algorithm and SFS as search procedure.	70
4.6 The test performance of error rate-based wrapper methods on the colon cancer dataset using SVM as classification algorithm and SFS as search procedure.	71
4.7 Comparison of SVM-based wrapper methods using four error estimators and generalized $\ \mathbf{w}\ ^2$ measure, respectively, as evaluation criterion and using SFS as search procedure.	72
4.8 The test performance of error rate-based wrapper methods on the leukaemia dataset using KNN as classification algorithm and SFS as search procedure.	74
4.9 The test performance of error rate-based wrapper methods on the breast cancer dataset using KNN as classification algorithm and SFS as search procedure.	75
4.10 The test performance of error rate-based wrapper methods on the colon cancer dataset using KNN as classification algorithm and SFS as search procedure.	76
4.11 Comparison of KNN-based wrapper methods using four error estimators and modified Relief's measure, respectively, as evaluation criterion and using SFS as search procedure.	77

4.12 The test performance of error rate-based wrapper methods on the leukaemia dataset using SVM as classification algorithm and SFFS as search procedure.	78
4.13 The test performance of error rate-based wrapper methods on the breast cancer dataset using SVM as classification algorithm and SFFS as search procedure.	79
4.14 The test performance of error rate-based wrapper methods on the colon cancer dataset using SVM as classification algorithm and SFFS as search procedure.	80
4.15 Comparison of SVM-based wrapper methods using four error estimators and generalized $\ w\ ^2$ measure, respectively, as evaluation criterion and using SFFS as search procedure.	81
4.16 The test performance of error rate-based wrapper methods on the leukaemia dataset using KNN as classification algorithm and SFFS as search procedure.	82
4.17 The test performance of error rate-based wrapper methods on the breast cancer dataset using KNN as classification algorithm and SFFS as search procedure.	83
4.18 The test performance of error rate-based wrapper methods on the colon cancer dataset using KNN as classification algorithm and SFFS as search procedure.	84
4.19 Comparison of KNN-based wrapper methods using four error estimators and modified Relief's measure, respectively, as evaluation criterion and using SFFS as search procedure.	85

4.20 The test performance of error rate-based wrapper methods on the sonar dataset using SVM as classification algorithm. 87

4.21 The test performance of error rate-based wrapper methods on the sonar dataset using KNN as classification algorithm. 88

4.22 Comparison of wrapper methods using four error estimators and continuous measure, respectively, as evaluation criterion on the sonar dataset. 89

4.23 Performance of wrapper-based gene selection using bolstered error as evaluation criterion and SVM as classification algorithm on the leukaemia dataset. 92

4.24 Performance of wrapper-based gene selection using bolstered error as evaluation criterion and SVM as classification algorithm on the breast cancer dataset. 93

4.25 Performance of wrapper-based gene selection using bolstered error as evaluation criterion and SVM as classification algorithm on the colon cancer dataset. 94

5.1 The external .632 bootstrap error rate when conducting gene selection with different γ 's on the leukaemia dataset. 106

5.2 LS Bound measure during the SFS procedure on the colon cancer dataset. 107

5.3 Comparison of gene selection methods, including LS Bound-based selection, Weighting factor and SVM RFE, on the colon cancer dataset. 109

5.4 LS Bound measure during the SFS procedure on the breast cancer dataset. 110

LIST OF FIGURES

xv

5.5 Comparison of gene selection methods, including LS Bound-based selection, Weighting factor and SVM RFE, on the breast cancer dataset. 113

5.6 LS Bound measure during the SFS procedure on the leukaemia dataset. 115

5.7 Comparison of gene selection methods, including LS Bound-based selection, Weighting factor, and SVM RFE, on the leukaemia dataset. 116

6.1 The external .632 bootstrap error rate when conducting gene selection with different λ 's on the leukaemia dataset. 132

6.2 Comparison of gene selection methods, including RN-based selection, Weighting factor and SVM RFE, on the colon cancer dataset. 133

6.3 Comparison of gene selection methods, including RN-based selection, Weighting factor and SVM RFE, on the breast cancer dataset. 136

6.4 Comparison of gene selection methods, including RN-based selection, Weighting factor and SVM RFE, on the leukaemia dataset. 138

7.1 The external .632 bootstrap error rate when conducting R^2W^2 -IRFS ($\epsilon = 0$) gene selection with different C 's on the breast cancer dataset. 160

7.2 Comparison of the selection performance of SVM-IRFS algorithm for different ϵ settings on the breast cancer dataset 162

7.3 Comparison of the selection performance of SVM-IRFS algorithm for different ϵ settings on the colon cancer dataset 163

7.4 Comparison of the selection performance of SVM-IRFS algorithm for different ϵ settings on the leukaemia dataset 164

7.5	Comparison of the selection performances of R^2W^2 -IRFS algorithms for $\epsilon = 1\%$ and $\epsilon = 0$ on several particular bootstrap sample sets of leukaemia dataset.	168
7.6	Comparison of gene selection methods, including W^2 -IRFS, R^2W^2 -IRFS and SVM-RFE, on the three benchmark datasets.	169
8.1	Comparison of four gene selection algorithms we proposed, including LS Bound-based gene selection, RN-based gene selection, W^2 -IRFS and R^2W^2 -IRFS, on the three benchmark datasets.	188

List of Tables

4.1 Description of the components of 10 wrapper methods. 63

4.2 The number of genes which lead to the same minimal criteria at each
step of SFS for SVM-based wrapper methods on the leukaemia dataset. 64

4.3 The number of genes which lead to the same minimal criteria at each
step of SFS for KNN-based wrapper methods on the leukaemia dataset. 64

4.4 The number of genes which lead to the same minimal criteria at each
step of SFS for SVM-based wrapper methods on the breast cancer
dataset. 65

4.5 The number of genes which lead to the same minimal criteria at each
step of SFS for KNN-based wrapper methods on the breast cancer
dataset. 65

4.6 The number of genes which lead to the same minimal criteria at each
step of SFS for SVM-based wrapper methods on the colon cancer
dataset. 65

4.7 The number of genes which lead to the same minimal criteria at each
step of SFS for KNN-based wrapper methods on the colon cancer
dataset. 65

LIST OF TABLES

xviii

5.1	The selected genes by LS Bound-based selection algorithm for the colon cancer dataset.	108
5.2	The selected genes by LS Bound-based selection algorithm for the breast cancer dataset.	112
5.3	The selected genes by LS Bound-based selection algorithm for the leukaemia dataset.	114
6.1	The top 20 genes selected by RN-based selection algorithm for the colon cancer dataset.	134
6.2	The top 20 genes selected by RN-based selection algorithm for the breast cancer dataset.	135
6.3	The top 20 genes selected by RN-based selection algorithm for the leukaemia dataset.	137
7.1	Description of the datasets	159
7.2	Running times (in minutes) of the generalized $\ w\ ^2$ measure related algorithms, including W^2 -IRFS, W^2 -SFS, and RFE, when 50 genes are selected on the three datasets.	160
7.3	Running times (in minutes) of the generalized radius/margin bound related algorithms, including R^2W^2 -IRFS and R^2W^2 -SFS, when 50 genes are selected on the three datasets.	161
7.4	The relative differences and computational savings using W^2 -IRFS algorithm at different settings of $\epsilon(> 0)$ on three datasets.	165
7.5	The relative differences and computational savings using R^2W^2 -IRFS algorithm at different settings of $\epsilon(> 0)$ on three datasets.	165

LIST OF TABLES

xix

7.6

The top 25 genes selected by W^2 -IRFS algorithm ($\epsilon = 1\%$) on the breast cancer dataset. The genes marked with * are also selected by R^2W^2 -IRFS.

170

7.7

The top 25 genes selected by R^2W^2 -IRFS algorithm ($\epsilon = 1\%$) on the breast cancer dataset. The genes marked with * are also selected by W^2 -IRFS.

171

7.8

The top 25 genes selected by W^2 -IRFS algorithm ($\epsilon = 1\%$) on the colon cancer dataset. The genes marked with * are also selected by R^2W^2 -IRFS.

172

7.9

The top 25 genes selected by R^2W^2 -IRFS algorithm ($\epsilon = 1\%$) on the colon cancer dataset. The genes marked with * are also selected by W^2 -IRFS.

173

7.10

The top 25 genes selected by W^2 -IRFS algorithm ($\epsilon = 1\%$) for the leukaemia cancer dataset. The genes marked with * are also selected by R^2W^2 -IRFS.

174

7.11

The top 25 genes selected by R^2W^2 -IRFS algorithm ($\epsilon = 1\%$) on the leukaemia cancer dataset. The genes marked with * are also selected by W^2 -IRFS.

175

8.1

The running times (in minutes) using our proposed algorithms, including LS Bound-based gene selection, RN-based gene selection and two SVM-IRFS algorithms (W^2 -IRFS and R^2W^2 -IRFS), respectively, to identify 50 most important genes (from 1000 genes) on three datasets

187

Chapter 1

Introduction

1.1 Motivation

In 2001 available was the first draft of the human genome, which contains more than 30,000 genes. Among those genes, many of them encode transcription factors, or are dedicated to control elements (for example, enhancer) to which those transcription factors bind. The existence of transcription factors and other control elements makes the genome a highly complex gene regulatory network. Since then, research focus in molecular biology and genetics has shifted from the study of individual genes to the exploration of the entire genome. With recently developed technologies, such as DNA Microarray (also known as DNA chips, gene chips or biochips), researchers are able to simultaneously measure expression levels of tens of thousands of genes in a single experiment. The microarray technique provides the power of creating a comprehensive overview of the gene regulatory network. Moreover, it is common to make use of the gene expression profiles, which are generated from microarray experiments, for phenotypic discrimination [9,45,114]. Nevertheless, the microarray experiments produce a gigantic amount of data, which present challenges to data analysis.

In the microarray data, each sample is represented by a large number of genes (features), usually several thousands. Many of them are irrelevant, insignificant or redundant to the discriminant problem under investigation. The identification of informative genes, which have the greatest power for classification, is of fundamental and practical importance to the investigation of specific phenotypic discriminant problems, such as presence of cancer. Such selected marker genes can also be found of great value in further investigation of the genotype-phenotype relationship. Furthermore, as gene expression changes always occur in pathological process before any symptoms showing up, the information obtained from selected genes is quite helpful for earlier diagnosis and earlier treatment.

From the viewpoint of machine learning, the number of samples in a typical microarray dataset is extremely small (normally fewer than 100) compared with the high dimensionality (number of genes). In such a sparse space, it is quite easy to find a decision function that perfectly separates the sample data if we consider a pattern recognition problem. However, such a classifier might perform poorly on the unseen test data. In statistics, this phenomenon is described as the curse of dimensionality. The term “curse of dimensionality” is coined by Richard Bellman [15], and it states the fact that the convergence of any estimator to the true value of a smooth function defined on a high-dimensional space is very slow. In the microarray analysis, the curse of dimensionality means that we require an enormous amount of samples to achieve a good estimation of a function of the genes. However, this is impractical. Usually, the curse of dimensionality is alleviated by reducing the number of genes.

Selecting relevant genes has received increasing attention, recently. However, the field is still evolving, and has not yet reached a level of maturity. Even very basic methods, such as fold change and weighting factor, are still widely used in practical applications to identify differentially expressed genes.

Considering each gene as a feature, gene selection can be cast as a feature selection

problem. However, traditional feature selection methods might not be directly applied in the microarray data analysis due to the small sample size and extremely high dimensionality. For example, the branch and bound algorithm [113] is a general search procedure for finding the optimal subset to solve feature selection problem. However, it is computational infeasible to use this algorithm in the microarray data analysis due to the high dimensionality (the number of possible subsets that must be evaluated is an exponential function of the number of features). Therefore, the gene selection problem is worthy of more investigations.

1.2 Objectives

The wrapper method is often used in the microarray data analysis, as it can provide features with better predictive accuracy. The wrapper method evaluates feature subsets based on the classification performance. Regularization Network (RN), Support Vector Machine (SVM) and Least Squares Support Vector Machine (LS-SVM) are powerful classification methods that are suitable for sparse data (with the high dimensional and small-sized settings). The design of appropriate evaluation criterion for gene selection based on these advanced classification methods need further exploration.

Error rate is commonly used as an evaluation criterion in wrapper methods, and it is often considered as one of the best criteria in feature selection. The applicability of error rate based wrapper methods in gene selection for microarray data is worthy of investigation.

The objective of this thesis is to fulfill the requirement for selecting gene subsets that have the greatest power to classify different biological phenotypes, such as cancerous and normal tissues. Our goal is to develop new robust and effective computational algorithms for gene selection and cancer classification. The developed gene selection

algorithms should have improved predictive accuracy, robustness and reasonable computational cost with applications to real microarray data. The focus of this thesis is purely on the computational part of gene selection, and not the biological explanation of the selected genes.

1.3 Major Contribution of the Thesis

The follows are the main contributions of the thesis:

1. The ties problem resulting from counting-based error estimators

Error rate is usually used as the evaluation criterion for wrapper-based feature selection. However, our study reveals that evaluating genes in terms of counting-based error estimators, such as resubstitution error, leave-one-out error, cross-validation error and bootstrap error, could encounter severe ties problem, *i.e.* more than one gene subset score equally, and this in turn results in uncertainty in gene selection. Our analysis finds that the ties problem is caused by the discrete nature of counting-based error estimators, and could be avoided by using continuous evaluation criteria instead. Experiment results show that well-designed continuous evaluation criteria, such as generalized $\|\mathbf{w}\|^2$ measure for Support Vector Machines and modified Relief's measure for k -Nearest Neighbor, produce improved gene selection performance compared with counting-based error estimators.

2. LS Bound-based gene selection algorithm

A new evaluation criterion, called LS Bound measure, related to LS-SVM (Least Squares Support Vector Machines) is proposed to address the gene selection problem. The LS Bound measure is derived from the leave-one-out procedure of LS-SVMs, and as an upper bound for the leave-one-out classifi-

cation results it reflects to some extent the generalization performance of gene subsets.

3. RN-based gene selection algorithm

We proposed a wrapper-based gene selection algorithm employing the Regularization Network (RN) as the classification algorithm. The computational cost in our RN-based gene selection algorithm is significantly reduced compared with error rate based wrapper method, because the evaluation criterion we used does not demand repeated training in the re-sampling procedure, although it is exactly calculated from the leave-one-out classification results.

4. Generalized radius/margin bound for separable and non-separable SVMs

The radius/margin bound [112] is the commonly used error bound for Support vector machines (SVMs). However, it is only applicable to separable cases. To solve the problem, we proposed a generalized radius/margin bound, which is modified from traditional radius/margin bound, to assess the generalization performance of SVMs in both separable and non-separable cases.

5. Iterative Reduced Forward Search (IRFS)

For gene selection on high dimensional microarray data, there are a huge number of possible feature subsets. Therefore, a well-designed search algorithm is critical for successful applications on high dimensional microarray data. We proposed an Iterative Reduced Forward Search (IRFS) algorithm to avoid unnecessary computations, and then to enhance the search efficiency. The IRFS algorithm relies on a very important property of the evaluation criterion, namely that for two feature subsets, A and A' ,

$$J(A') \geq h(J(A), A, A').$$

The right hand side of the above inequality is called the moving bound of criterion $J(\cdot)$. In this work, the IRFS algorithm is combined with two SVM-based criteria, generalized $\|\mathbf{w}\|^2$ measure and generalized radius/margin bound, respectively, to form two SVM-IRFS algorithms for gene selection on microarray data.

1.4 Organization of the Thesis

The material in this thesis covers biological background, literature review, novel gene selection algorithms, mathematical proofs and experimental studies on real microarray data. The structure of this thesis is organized as follows:

Chapter 1 describes the motivation of the research, the objectives, major contributions, and gives a brief outline of the content of each chapter. In Chapter 2, we briefly introduce the biological background and data analysis techniques for DNA microarrays. A detailed literature review on gene selection is presented in Chapter 3. In Chapter 4, we investigate the ties problem resulting from counting-based error estimators. Our analysis shows that the ties problem is caused by the discrete nature of counting-based error estimators, and could be avoided by using well-designed continuous evaluation criteria instead. In Chapter 5~7, three gene selection algorithms are developed to solve the ties problem by employing the continuous evaluation criteria. Chapter 5 proposes the LS Bound-based gene selection algorithm. The evaluation criterion, LS Bound measure, is derived from the leave-one-out procedure of Least Squares Support Vector Machines (LS-SVMs). Chapter 6 presents a wrapper-based gene selection algorithm employing the Regularization Network (RN) as the classification algorithm. Chapter 7 proposes the SVM-IRFS gene selection algorithms, which combine SVM-based evaluation criteria with a new search approach, Iterative Reduced Forward Search (IRFS). In this chapter, two SVM-

1.4 Organization of the Thesis

7

based criteria, generalized $\|\mathbf{w}\|^2$ measure and generalized radius/margin bound, are employed. Conclusions and suggestions for future research are presented in Chapter 8.

Chapter 2

DNA Microarrays

2.1 Genetic Background

The genetic material of all known organisms and many viruses is the *deoxyribonucleic acid* (DNA). DNA is composed of nucleotides, each of which consists of a deoxyribose sugar, a phosphate group and one of the four nitrogenous bases: adenine (A), guanine (G), thymine (T) and cytosine (C). Nucleotides link together to form a polynucleotide chain. The sequence of DNA can be thought as a “word” from a four lettered alphabet {A,C,G,T}. DNA exists as a double-stranded helical polymer. The two strands are held together by hydrogen bonds between complementary pairs of bases (A always paired with T, and G always paired with C). Fig. 2.1 shows a schematic view of this structure.

A *protein* is a group of complex organic macromolecules, and is composed of one or more chains of amino acids. Proteins are fundamental components of all living cells, and include many substances, such as enzymes, hormones and antibodies. The structure, function, development and reproduction of an organism depend on the proteins that exist in cells and tissues.

Figure 2.1: The DNA structure. The figure is from [108]

A *gene* is a segment of DNA involved in encoding a *protein*. It includes regions preceding and following the coding region and intervening segments (*introns*) between individual coding segments (*exons*). The *genome* is the complete set of genes of an organism. Genes interact with each other to influence an individual’s observable characteristics, such as hair or fur color, weight and susceptibility to diseases. The estimate for the total number of genes in human genome is 30,000 ~ 40,000.

How does the genetic information travel from DNA to the proteins that perform most of the critical functions of cells? The process of protein synthesis is revealed by the central dogma of molecular biology as shown in Fig. 2.2. The process consists of two major stages. The first stage, *transcription*, transfers the genetic information from the double-stranded DNA into *messenger ribonucleic acid* (abbreviated to mRNA), a single-stranded complementary copy of DNA molecules with the base uracil (U) replacing thymine (T). The second stage, *translation*, refers to the production of

protein on the mRNA template. Transcription and translation together are called *gene expression*.

Most genes consist of a sequence of DNA that codes for one protein. However, it is common, especially in the human genome, that one gene codes for more than one proteins. A single gene may generate a variety of mRNA products that include different content of exons. This is called alternative splicing, shown in Fig. 2.3. Alternative splicing expands the number of proteins relative to the number of genes by about 15% in flies and worms, but it has much bigger effects in human, where ~60% of human genes are alternatively spliced. By considering about 80% of the alternative splices change protein sequence, the human *proteome* (the complete set of proteins) has approximately 50,000 ~ 60,000 members.

Figure 2.2: The central dogma of molecular biology. DNA can replicate itself in a process involves many enzymes. DNA codes for production of mRNA during transcription, while in the process of translation mRNA acts as a template for protein synthesis. Together, transcription and translation constitute gene expression. The figure is from <http://users.ugent.be/~avierstr/>.

Figure 2.3: Various modes of alternative splicing. The figure is from Wikipedia, the free encyclopedia.

We refer the readers to [68] for an introduction to the relevant biology.

The study of genes that are expressed and those that are not expressed in different kinds of cells and under different environmental conditions helps scientists to understand how these genes function. Note that the main interest in studying the function of genes should be based on the understanding of the function of corresponding proteins. However, compared with the analysis of proteins, the measurement at the mRNA levels is considerably much easier at the moment. Although the mRNA is not the final product of a gene, the transcription levels to some extent reflect the abundance of corresponding proteins because transcription is the first stage in the process of gene expression. The recently developed DNA microarray technique has the ability to monitor the gene expression at the transcription level.

2.2 DNA Microarrays Technology

With the completion of the draft of the human genome, the genome research is entering a new area called *functional genomics*. Functional genomics focuses on

the assignment of biological functions to DNA sequences. The essential task of functional genomics is to define the role of each gene and to understand interactions and relationships among groups of genes and proteins. With the emergence of the microarray technique, the expression levels of tens of thousands of genes can be measured simultaneously, which makes this task practically possible.

A DNA microarray system usually consists of DNA probes formatted on a solid support, most commonly glass, together with instruments to read reporter (fluorescent or radioactive) molecules. Gene transcriptional material or target DNA is collected from samples and labeled with reporter molecules. Here the term *probe* refers to the known sequence, which is attached to the surface of microarray, and the term *target* describes the unknown sequence or collection of sequences to be analyzed. The labeled target is then allowed to hybridize with the probes formatted on the array. The relative abundance of the target hybridized to each DNA probe is quantified by measuring the intensity of the report molecules. As the configuration of attached DNA probes is known, the intensities can be associated with particular genes so as to report their expression levels.

The basic principle of microarray technology is *hybridization*. The term *hybridization* refers to the strong ability to form a hybrid for the complementary pair of DNAs from different sources according to base-pairing rules (i.e., pairing of A-T and G-C for DNA; A-U and G-C for RNA). There are two main types of DNA microarray formats currently in wide use: cDNA and oligonucleotide arrays. These two formats are illustrated in Fig. 2.4 and Fig. 2.5. Although the two formats follow the same processing procedure as we described before, however, the ways of making arrays and measuring the abundance of DNA targets are different.

The first step of producing cDNA arrays is to select the probes, which will be printed on the array, from databases including GenBank, UniGene or other cDNA sources. The selected probes are amplified by PCR (Polymerase Chain Reaction), and are

Figure 2.4: cDNA microarray schema. The figure is from [1].

printed on coated glass microscope slides using a computer-controlled, high-speed robot. The targets for cDNA arrays are synthesized from total RNA extracted from both the test and reference samples, and are then fluorescently labeled with either Cy5 (green) or Cy3 (red). The fluorescent targets are pooled and hybridized to the probes on the array. After hybridization, the array is scanned with two different wavelengths. A ratio between the two fluorescent intensities (that is, Cy5/Cy3) is calculated to present the relative abundance of a gene in the two cDNA samples.

By contrast, oligonucleotide arrays are with a single channel, and use the photolithography approach to make arrays. The oligonucleotide (“a short stretch of synthetic DNA that replicates a unique sequence identifying a gene” [78]) is grown base by base on the array surface by means of photolithographic methods. Labeled cDNA or cRNA targets derived from the mRNA of the sample are hybridized to the oligonucleotide probes attached to the array support. By monitoring the quantity of labeled targets associated with each DNA location, one may infer the abundance

Figure 2.5: Oligonucleotide microarray schema. The figure is obtained from the website of Affymetrix Inc. (<http://www.affymetrix.com>).

of each mRNA.

Although the human genome contains 30,000 ~ 40,000 genes, there may be no upper limit to the number of human genes that are measured on a microarray. The GeneChip[®] Human Genome U133 Plus 2.0 Array from Affymetrix (<http://www.affymetrix.com>) analyzes the expression levels of over 47,000 transcripts and variants, including 38,500 well-characterized human genes. Another Affymetrix array set, GeneChip[®] Human Genome U95 Set (5 arrays), contains almost 63,000 probe sets interrogating about 54,000 UniGene clusters.

The oligonucleotide approach has been commercially available for several years, and a large number of published studies use this system. For example, the datasets that we use in this thesis: colon cancer dataset [9], breast cancer dataset [114] and leukaemia dataset [45] are all generated by Affymetrix oligonucleotide arrays. Although it is a generally accepted method by many laboratories, the oligonucleotide approach has several disadvantages. For example, if different arrays are desired

it is not flexible to make relevant changes. In addition, the occurrence of cross-hybridization to unrelated probes may affect the accuracy of the system due to the short length of the oligonucleotide probes. While in the cDNA approach, the large size of the PCR product is helpful to reduce the occurrence of unexpected cross-hybridization. Another advantage of cDNA approach is that no sequence information is needed in advance. However, it is difficult and expensive to produce, analyze and purify the PCR products of a large number of cDNAs.

The thorough descriptions of the microarray experiment can be found in [1,2,13,14,72].

2.3 Microarray Data Analysis

The DNA microarray technique provides a wealth of data that can be used to develop a more complete understanding of gene function, regulation and interaction. However, the increasing flood of data presents enormous challenges for analyzing microarray data. General areas where computational methods can make significant contributions to microarray experiments include: analysis of microarray scanned images; normalization of fluorescence intensities across single slide or multiple slides; assessment of the quality of microarray data; clustering of similar genes and samples; recognition of specific patterns at the expression levels; and selection of important genes for the given biological problems. The main steps in the design and analysis of a microarray experiment are summarized in Fig. 2.6.

Microarrays measure the relative or absolute mRNA abundance by assessing the fluorescent intensity at each spot. The raw data produced by microarrays are monochrome images (Fig. 2.7). The brightness of the spot is proportional to the abundance of mRNA targets hybridized to the DNA probes. Softwares provided with the microarray image scanner can recognize these spots, determine their bound-

Figure 2.6: Main steps in design and data analysis of a microarray experiment

aries, measure the fluorescent intensity from each spot. Moreover, the measurements should be compared with similar images obtained from the control measurements (especially for cDNA microarray), and should be compared with replicates, if the experiment has, to make the data reliable. In this way raw images are transformed into the gene expression matrix for further analysis. [43]

The above is the general procedure for analyzing the raw images of microarray. For various microarray platforms, different statistical techniques are employed. We take Affymetrix GeneChip[®] oligonucleotide array as an instance. Affymetrix arrays pairing probes to control the problem of cross-validation. Typically, an mRNA molecule of interest is represented by a probe set which includes 11~20 probe pairs of oligonucleotides with 25 base-pairs. Each probe pair is composed of a perfect match (PM) and Mismatch (MM) probes. The PM probe is a section of the sequence

Figure 2.7: A sample raw image from scanning a hybridized rat microarray which contains more than 5000 genes. The figure is from [22]

of interest. The MM probe is created by changing the middle (13th) base of the PM probe. For example,

PM: GCTAGTCGATGCTAGCTTACTAGTC

MM: GCTAGTCGATGCAAGCTTACTAGTC.

Ideally, the MM value can be used as a rough assessment of the amount of non-specific binding (or cross-hybridization) associated with a given PM probe. The algorithms used for quantitation of raw images, background adjustment, and normalization have to take the PM/MM pairs into consideration. However, those issues are not the concerns of this thesis. Interested readers may refer to [3-6,55] for more details. We assume this procedure has been done. Our work starts from the gene expression matrix.

For n microarray experiments (corresponding to n samples), where in each experiment the expression levels of p genes are measured, the expression data can be represented by a $p \times n$ matrix, as illustrated in Fig. 2.8. For each sample, the expression levels of p genes are called its *expression signature*. Conversely, for each

gene, its expression levels across the different samples are called its *expression profile*. The n samples might correspond to n different patients, or correspond to samples from one patient taken at n different time points. The expression levels are taken to be the absolute intensities (for oligonucleotide microarrays) or ratios between two fluorescent intensities (for cDNA microarrays).

Figure 2.8: Microarray data from n experiments represented as a matrix of expression levels with p rows corresponding to p genes and n columns to n samples.

To analyze the data and to extract some information for interpreting the biological processes, the gene expression data can be explored in a variety of ways, such as clustering, discriminant analysis and gene selection. Clustering is an unsupervised task, that is, no predefined classification is required. Clustering algorithms divide the data under study into natural groups. Unsupervised learning is particularly useful to analyze the data in an unknown area. If we have some prior information about which samples or genes are expected to group together, this information can be utilized in a supervised manner (discriminant analysis). Discriminant analysis is to construct a model to classifying a set of samples into predefined classes. The model is built based on the training set, in which the class of each sample is known. Gene selection is the focus of this thesis, and we will give a detailed review of gene selection methods with Chapter 3. If put in the context of pattern classification, gene selection

can be solved as a feature selection problem. Actually, gene (feature) selection is not a separate area of gene expression data analysis. Some classification technologies, either supervised [45, 47, 69] or unsupervised [35], are involved in the process of gene selection, although practical considerations will force us to simply assume that the stages of feature selection and classification are mutually independent. In the follows, we will briefly introduce clustering and discriminant analysis.

2.3.1 Clustering of Microarray Data

Clustering approaches are currently the most widely used techniques for gene expression analysis. There are two categories of clustering algorithms applied to the gene expression data analysis. The first one is based on pair-wise comparison of samples or genes. The comparison is performed on the basis of the definition of distance metric between samples or genes. The Euclidean distance and Pearson's correlation coefficient are two commonly used metrics in the context of gene expression profiles. Although clustering techniques are extremely powerful, they are, in some sense, subjective because selecting different algorithms or different metrics will form different clusters. The clustering techniques in the first category applied to gene expression profiles include hierarchical clustering [7–9, 38, 42, 46, 59, 76, 83, 85, 91, 93, 109, 119] and K-means clustering algorithms [53, 105], as well as self-organizing maps (SOM) [45, 102]. Another category of clustering algorithms used in gene expression profiles includes principal component analysis (PCA) [10, 12, 60, 87, 89] and multi-dimensional scaling (MDS) [8, 46, 74]. The purpose of those techniques is to reduce the dimensionality of samples to visualize the data structure and get a feeling for what the data look like, what its properties are.

Hierarchical clustering is a bottom-up clustering method. Initially each cluster consists of only one sample or gene, then the closest pair of clusters merge to a new cluster successively until all samples or genes are in one cluster. The merging pro-

cess can be represented by a dendrogram, which can be cut at any level to form a specific number of clusters.

The purpose of PCA is to maintain the variation of the original data as much as possible in the first principal components. In the literature of gene expression data analysis, PCA is often employed in combination with other techniques, such as SOM or K-means, to help to visualize the complexity of the data in a two or three dimensional space.

2.3.2 Discriminant Analysis of Microarray Data

For most biological problems, there is some information available beforehand. The purpose of discriminant analysis is to utilize these information to develop a robust classifier that can successfully handle blinded test data. Many well developed supervised algorithms, such as support vector machine (SVM) [41,79,91], artificial neural network (ANN) [46,60,92], are applied in the context of microarray data.

SVMs are well suited to work with high dimensional data, such as gene expression data. When used for classification, SVMs separate one class from the other with a hyperplane that maximizes the distance between the hyperplane and the nearest sample of each class. For non-linear case, SVMs map the data from the original space into a higher dimensional space, called feature space, and search for the hyperplane to separate the data in the feature space. The most appealing feature with SVMs, although rarely used in the thesis, is that data can be implicitly mapped into feature space by defining a kernel function in the original space that plays the role of the dot product in the feature space. Brown *et al.* [23] compared the performance of SVMs with that of four supervised learning methods, including Fisher's linear discriminant analysis (also applied in [117]), Parzen windows, and two decision tree classifiers (C4.5 and MOC1). Their studies show that SVMs provide better performance than

the other four methods. We refer the readers to chapter 4, as well as [25, 112] for a thorough understanding of SVMs.

Khan *et al.* [60] utilized the multilayer perceptron, a kind of ANNs, for classification and diagnostic prediction of small, round blue cell tumors. The tumors include four different diagnostic categories. A multilayer perceptron is composed of an input layer, one or several hidden layers, and an output layer. The input data is fed into the input layer and triggers a response in the corresponding hidden layer. The response passes in turn from hidden layers to the output layer, in which each perceptron typically represents a class. In their study, ANN-based methods have proved to be powerful for classifying tumors based on the gene expression data.

Dudoit *et al.* [36] compared the performance of several different discriminant analysis approaches, including Fisher's linear discriminant analysis [117], K nearest neighbors (KNN) [87], maximum likelihood discriminant analysis (ML), classification trees [23], and aggregating classifiers [16], for the classification of tumors using gene expression data. In the study, they found that simple classification algorithms, such as diagonal linear discriminant analysis (a special case of ML) and KNN, perform remarkably well compared to other complex methods.

Besides the above algorithms introduced from the field of pattern classification, some novel supervised algorithms are proposed in the context of gene expression profiles. Golub *et al.* [45] described a weighted voting method for binary classification, and they also define a term, prediction strength, to indicate the margin of victory. After the publication of their work, the weighted voting method received much attention [76, 96, 99]. Ben-Dor *et al.* [16] proposed a clustering based classification method to determine the clustering threshold, which controls the number of clusters, through a supervised procedure.

Chapter 3

Literature Review of Gene Selection for DNA Microarray Data

One problem with gene expression data is that each sample has a large number of genes, usually more than 1000, and many of them are irrelevant or redundant to the discriminant problem under investigation. From the viewpoint of pattern recognition, identifying important genes is a key issue for developing a successful microarray application. In this chapter, we give a detailed review on gene selection methods.

This chapter is organized as follows. In Section 3.1, methods for dimensionality reduction are presented. In Section 3.2, a brief introduction to the framework of feature selection is given. The existing gene selection methods are described in Section 3.3. In the last section, the problem of selection bias for evaluating gene selection methods is discussed.

3.1 Reduction of Dimensionality

The great advantage of microarray technique is its ability to simultaneously monitor the expression of tens of thousands of genes. Facing with such high dimensional data, although more features seem to have more discriminant power, we have a few reasons to reduce the dimensionality.

- Among the large set of genes, many of them are irrelevant, insignificant or redundant to the particular discriminant problem under investigation. Such genes have no or few contributions to the given problem, and may bring unnecessary noises to the classification procedure. Studies have shown that a small subset of genes might be sufficient for a particular biological problem [45].
- Taking all genes into account will lead to computational difficulties during classification procedure, and make microarray data hard to handle and analyze.
- Reducing the number of genes relieves the demand for a large number of training samples, because the performance of a pattern classifier partly depends on the ratio of the number of samples to the number of features, that is, the higher the ratio is, the better the generalization of the classifier is [106]. The collection of a quite large number of samples for microarray measurement is expensive, time-consuming and even impossible.

Two major categories of dimensionality reduction methods, feature selection and feature extraction [34], are well applied in the literature of gene expression data analysis. For example, feature extraction methods, such as principal component analysis (PCA) [10, 12, 60, 87, 89], multi-dimensional scaling (MDS) [8, 46, 74], partial least squares (PLS) [40, 81, 82] and total principal component regression (TPCR) [103], are used to reduce the dimensionality of sample space for visualizing the data

structure. Feature selection methods, such as Weighting factor [8, 12, 41, 45, 87, 96] and Recursive Feature Elimination (RFE) [47, 91], are also employed to identify important genes for analyzing gene expression data.

The difference between feature selection and feature extraction is the space where features are selected. Feature selection methods select features in the original measurement space, while feature extraction methods choose features in a transformed space, in which new features are created by performing specific transformation on the original features. Generally, features generated by the feature extraction approach may not have clear meanings in the real world. By contrast, features provided by any feature selection approach retain their original meanings. The property of feature selection highly attracts us.

From the viewpoint of biologists, the importance of gene selection lies in its contribution to understanding diseases and functions of particular genes. One can compare, for example, the gene expression of normal cells with that of cancerous cells in order to learn which genes tend to be associated with the cancer; regulation of such genes could produce effective cancer treatment. Although the DNA microarray diagnostic technology has been successfully implemented in many laboratories, routine clinical use is still limited because of the high costs of the microarray system. If some genes are determined to be the most informative for the given diagnostic problem, economical devices, such as Flow-through Hybridization Device (US patent 6,020,187), can be utilized to replace the microarray system for clinical diagnosis.

In summary, reducing the dimensionality of the gene expression data is a key issue in developing successful gene expression data analysis. But how many genes are needed for a particular biological problem? In a typical biological system, the adequate number of genes is often hard to know. However, a biological phenotype seldom involves more than a few dozens of genes in practice. Golub *et al.* [45] selected 50 genes out of over 7000 genes to discriminate two types of acute leukaemias. But

on the same dataset, Inza *et al.* [54] only selected 2 to 4 genes when using wrapper approaches. They determined the number of genes by the classification performance on the training set or test set. Li and Yang [70] tried to solve the problem in another way. They utilized Akaike information criterion (AIC) and Bayesian information criterion (BIC) to determine the number of genes which provides the best model. These model selection criteria are defined as

$$AIC = -2\log(\hat{L}) + 2K \quad \text{and} \quad BIC = -2\log(\hat{L}) + \log(n)K,$$

where $\hat{L}$ is the maximized likelihood, K is the number of free parameters (number of genes included plus 1), and n is the sample size. The model with the lowest AIC or BIC is considered to be the best one. By comparing with null models, the two model selection criteria set upper limits of $22 \sim 25$ and $12 \sim 13$ genes, respectively, for the leukemia dataset [45]. As the purpose of gene selection from biological aspects is to discover the genes highly related to the given phenotype, we expect to excavate more useful information from those huge amount of data. Therefore, even if fewer genes may give the same performance on phenotypic classification, we prefer selecting 50 informative genes to selecting only 2 genes.

3.2 Feature Selection

Gene selection is a process of selecting a gene subset from the original full set of genes. In the context of pattern recognition, gene selection on DNA microarray data can be cast as a feature selection problem, where each gene represents one feature. The aim for feature selection is to either identify a subset of features to improve prediction accuracy, or simplify the structure without significantly decreasing prediction accuracy of the classifier induced only from the selected features [63]. In general, a feature selection algorithm consists of four basic components (see Fig. 3.1) [33]:

- a *search procedure* to generate candidate feature subsets;
- an *evaluation criterion* to evaluate the goodness of candidate subsets;
- a *stopping criterion* to decide when to stop the search procedure; and
- a *validation procedure* to check the validity of the selected feature subset.

Figure 3.1: Framework of typical feature selection algorithms

The *search procedure* generates candidate feature subsets for evaluation. In the search procedure, candidate feature subsets can be generated either sequentially or randomly. The well-known sequential forward selection (SFS) starts from an empty set and iteratively adds features, while the sequential backward selection (SBS) starts from the full feature set and iteratively discards features. The *evaluation criterion* measures the goodness of the candidate feature subsets generated by the search procedure. At each step of the iterative procedure, the value of the evaluation criterion on a candidate subset is calculated and compared with that on the previous best subset. The candidate subset replaces the previous best one if it produces a better evaluation result. The *stopping criterion* is used to decide when to terminate the feature selection procedure. The goal of including stopping criterion in feature selection is to prevent the feature selection process from running exhaustively. The commonly used stopping criteria include: (1) whether a predefined number of features are selected, and (2) whether addition (or deletion) of any feature does not significantly improve the value of the evaluation criterion. The feature selection process

outputs a selected feature subset to the validation procedure, when the predefined stopping criterion is satisfied. Although the validation procedure is not a part of the feature selection process itself, any feature selection algorithm must be validated in practice. The validation procedure evaluates the validity of the classifier induced from the selected subset by carrying out test procedure on different samples. Among the four components, evaluation criterion and search procedure are the most important for an accurate and efficient feature selection algorithm.

3.2.1 Evaluation Criterion

In order to identify a good feature subset, we need a measure to evaluate the goodness of feature subsets. This is achieved by defining a class separability measure that is optimized with respect to the candidate feature subsets. Generally, the feature subsets can be evaluated in two ways.

The first approach is based on the performance of a particular classification algorithm which is to be used in the validation procedure. We can induce a classifier on each candidate feature subset, and choose the subset on which the classifier performs well for an independent sample set. In this approach, the feature subset is chosen to match the classifier. Different choices of classification algorithm may result in different selected feature subsets. Error rate, the proportion of samples incorrectly classified, is a typical measure in this category. A minimum classification error rate is often expected in the feature selection process. In Chapter 4, we will discuss the performance of several error estimators with the applications to gene selection.

The second one is to estimate the overlap between the distributions from which the data are drawn. The feature subset with minimal overlap (that is, maximal separability) is preferred. This is independent of the classification algorithm that is employed in the validation procedure, so it has a lower computational cost. However,

it has the disadvantage that the assumptions made in determining the overlap are often crude and may result in a poor classification performance.

The two approaches above are often referred to as the wrapper and the filter methods, respectively. In the filter method, intrinsic properties of data, such as Weighting factor [45] and Mahalanobis class separability measure [34], are often used as the criteria for feature subset evaluation, while in the wrapper method, the performance of the classifier, such as error rate, is widely used as the evaluation criterion. In most pattern recognition applications, the wrapper method outperforms the filter method. However, the accuracy of wrapper methods is coupled with intensive computations.

3.2.2 Search Procedure

The problem of feature selection is to choose the best feature subset from a set of p features. If $\mathcal{X}$ represents the set of all possible feature subsets, then the optimal subset, X^* , is identified as:

$$X^* = \arg \max_{X \in \mathcal{X}} J(X), \quad (3.1)$$

where $J(\cdot)$ is the function of evaluation criterion. Without loss of generality, here we assume that the maximum of J is preferred.

The exhaustive search, one of the optimal search methods, evaluates the values of the criterion for all possible feature subsets. Suppose that we expect to select m features from p original features. The difficulty arises because the number of possible subsets, $\frac{p!}{(p-m)!m!}$, will be too large if p has a high value. For example, to select the best 5 out of 1000 features, we have to evaluate more than 8×10^{12} feature subsets. For microarray data, the dimensionality p is normally much larger than 1000, hence optimal search methods are computationally impractical. Therefore, suboptimal search methods, in which the optimality is traded off for computational efficiency, are

usually used in gene selection for microarray data analysis. Suboptimal techniques, varying in complexity, are reviewed next.

3.2.2.1 Individual Ranking

The simplest method for choosing the best m features is to assign an estimated discriminatory power to each feature in the original set, $\{x_i | i = 1, \dots, p\}$, individually. Thus, the features are re-ordered so that

$$J(x_1) \geq J(x_2) \geq \dots \geq J(x_p)$$

and the m features with the best individual scores: $\{x_i | i \leq m\}$ are selected.

With its linear time complexity in terms of dimensionality p , the individual ranking method is quite efficient for high dimensional microarray data. Therefore, individual ranking methods, such as Weighting factor [8,12,41,45,87,96], are widely used in the literature of microarray data analysis. However, this method may select redundant features. For high dimensional data containing a large number of severely correlated features, this method may lead to results far from optimal, as redundant features, which add little discriminatory power, can be selected.

To overcome the influence of redundant features, the following methods are performed in the manner of subset evaluation [17,48], that is, evaluating the goodness of a feature subset, instead of individual features. Methods of subset evaluation can remove both irrelevant features and redundant features, because the evaluation criteria used by these methods implicitly take into account the effect of irrelevant or redundant features. The feature subsets identified by these methods approximate the optimal subset.

3.2.2.2 Sequential Forward Selection (SFS)

Sequential forward selection is a bottom-up search algorithm. It starts with an empty set, and adds new features one at a time until the final feature subset is reached. Suppose we have selected a set of t features, S^t . For each of the feature z_j not yet selected (i.e. not in the set S^t), the value of evaluation criterion $J_j = J(S^t + z_j)$ is evaluated. The feature that yields the maximum value of J_j is chosen as the $(t + 1)$ 'th feature, and it is added to S^t . Thus, at each step, the feature is chosen to maximize the evaluation criterion when added into the current set. When the stopping criterion is satisfied, the algorithm terminates. The SFS algorithm is described as follows,

Algorithm 1: Sequential forward Selection algorithm

Input: n data pairs $\{\langle \mathbf{x}_i, y_i \rangle\}$, m (predefined number of features to be selected)

Output: indices of m selected features

Initialize S to an empty set;

// S is the set of selected features

Initialize C to the full feature set;

// C is the set of candidate features for selection

for $i \leftarrow 1$ **to** m **do**

p = number of features in set C ;

$J_{max} = -\infty$; // J_{max} stores the best value for comparison purpose

$I_{best} = 0$; // I_{best} stores the index of best feature

for $j \leftarrow 1$ **to** p **do**

 Take the j 'th feature in set C and temporarily put into set S ;

 Calculate the criterion J_j using all features in set S ;

 // evaluate the goodness of the temporary set S

if $J_j > J_{max}$ **then**

$J_{max} = J_j$; $I_{best} = j$;

end

end

 Put the I_{best} 'th feature into set S ;

 Remove the selected feature from set C ;

end

return feature set S

The main disadvantage of the SFS method is the “nesting effect”. The SFS method

does not discard any features from the selected feature subset once they have been added in even if further additions make them unnecessary.

3.2.2.3 Sequential Backward Selection (SBS)

Sequential backward selection is a top-down search procedure. It starts with the full feature set, and discards features one at a time until the final feature subset is reached. Suppose we have selected a set of t features, S^t . For each of the feature z_j in the set S^t , the value of evaluation criterion $J_j = J(S^t - z_j)$ is computed. The feature that yields the maximum value of J_j is discarded from the set S^t . The process is repeated until the stopping criterion is satisfied. The SBS algorithm is described as follows,

Algorithm 2: Sequential backward selection algorithm

Input: n data pairs $\{\langle \mathbf{x}_i, y_i \rangle\}$, m (predefined number of features to be selected)

Output: indices of m selected features

Initialize S to the full feature set;

// S is the set of selected features

p = number of features in Set S ;

while $p \neq m$ **do**

for $j \leftarrow 1$ **to** p **do**

 Temporarily discard the j 'th feature from set S ;

 Calculate the criterion J using all features in set S ;

 // evaluate the goodness of the temporary set S

end

 Identify the feature with the maximal J ;

 Remove the identified feature from set S ;

p = number of features in set S ;

end

return feature set S

Similar as the “nesting effect” of the SFS algorithm, the SBS procedure has the disadvantage that the features which have been discarded at previous steps would not be added into the feature set.

3.2.2.4 Floating Search

Floating search methods [88], including SFFS (sequential forward floating selection) and SBFS (sequential backward floating selection), allow backtracking as long as it optimizes the evaluation criterion function. Such backtracking overcomes the “nesting effect” in SFS and SBS.

We take SFFS as an example. The SFFS is basically a bottom-up search procedure, which adds new features by means of applying the basic SFS procedure, followed by a series of successive conditional exclusion of the worst feature. The exclusion process makes further improvement to the previous selected feature subsets. Suppose that the feature subset, X_k , containing k features have been selected with the corresponding evaluation criterion $J(X_k)$ at the k 'th step. In addition, the values of $J(X_i)$ for all preceding selected subsets X_i of size $i = 1, 2, \dots, k-1$, are known and stored. The process to select subset X_{k+1} consists of several inclusion and conditional exclusion steps which are described as follows.

- *Step 1 (inclusion)*: Using the basic SFS method, select feature x_{k+1} from the features not in the subset X_k . Therefore $X_{k+1} = X_k + x_{k+1}$. Go to Step 2.
- *Step 2 (conditional exclusion)*: Using the basic SBS method, identify the least significant feature, x_r , in the feature subset X_{k+1} . If $J(X_k) \geq J(X_{k+1} - x_r)$, then set $k = k + 1$ and return to Step 1.

If $J(X_k) < J(X_{k+1} - x_r)$, then discard feature x_r from X_{k+1} to form a new feature set X'_k . Note that $J(X'_k) > J(X_k)$. Hence set $X_k = X'_k$, $k = k - 1$; and repeat Step 2

The SFFS algorithm is initialized by setting $k = 0$ and $X_0 = \emptyset$, where $\emptyset$ denotes an empty set, and the SFS method is used until 2 features are selected. Then the algorithm continues with Step 1.

3.3 Existing Gene Selection Methods

3.3.1 Fold Change

The fold change approach for identifying differentially expressed gene is commonly used in the literature at least in the first wave of publications [52, 94] and in the applications for pathway elucidation [75]. The idea of the approach is easy to understand. A gene is declared to have been significantly changed if its average expression level varies by more than a constant factor, say 2 commonly, between the treatment and control conditions. In practice, two boundaries are used, for instance 2 and 0.5. When the expression ratio is over 2, the gene is considered as upregulated, while downregulated when the ratio is less than 0.5.

Although the approach has been applied to a large number of applications, the shortcoming of the fold change approach is apparent. Many events may cause a significant change between expression levels of two conditions. For instance, in a cancer experiment, the change could be associated with a direct link (activation of an oncogene), with a indirect link (signaling pathway), with a non-specific related event (cell division), or even with noise. The only way to eliminate the effect of noise is to increase the replicates or samples. Some statistics tests, such as t-statistic, are involved.

3.3.2 Individual Gene Ranking

Informative genes are considered to be highly relevant to the studied phenotype. Evaluating the relevance of an individual gene can produce a simple gene (feature) ranking, then top m informative genes are selected. Some concepts from statistics are introduced to evaluate the relevance.

3.3.2.1 Weighting Factor

The Weighting factor [45] is defined as:

$$W(i) = \frac{\mu_1(i) - \mu_2(i)}{\sigma_1(i) + \sigma_2(i)}$$

where $\mu_1(i)$, $\mu_2(i)$, $\sigma_1(i)$ and $\sigma_2(i)$ are means and standard deviations of gene expression levels of gene i for two classes, respectively. The large value of $|W(i)|$ indicates a strong correlation between the gene expressions and the studied phenotype, while the sign of $W(i)$ corresponds to the class in which gene i is more highly expressed. The Weighting factor method was first used by Golub *et al.* [45] to analyze the gene expression data. In their work, they selected an equal number of genes with positive and negative correlations. This criterion was also used in [8,12,87,96]. The absolute value of $W(i)$ was used as ranking criterion in [41].

3.3.2.2 t-statistic

Hedenfalk *et al.* [51] employed the t-statistic to measure the difference of gene expressions between two classes. The t-statistic is defined as:

$$t(i) = \frac{\mu_1(i) - \mu_2(i)}{\sqrt{\frac{s_1^2(i)}{n_1} + \frac{s_2^2(i)}{n_2}}}$$

where $\mu_1(i)$, $\mu_2(i)$, $s_1^2(i)$ and $s_2^2(i)$ are means and sample variances of gene expression levels of gene i for two classes respectively, and n_1 , n_2 are sizes of two classes. Large absolute value of t-statistic suggests that the corresponding gene has significantly different expression level between two classes. Such genes are the most likely to be selected.

3.3.2.3 Fisher's Ratio

The Fisher's discriminant ratio is defined as:

$$f(i) = \frac{(\mu_1(i) - \mu_2(i))^2}{\sigma_1^2(i) + \sigma_2^2(i)}$$

where $\mu_1(i)$, $\mu_2(i)$, $\sigma_1^2(i)$ and $\sigma_2^2(i)$ are means and variances of i 'th gene's expression levels for two classes, respectively. Similar to t-statistic, the large ratio indicates the distinct difference between two classes.

3.3.2.4 Ratio of Between-group to Within-groups Sum of Squares

The above criteria are only applied to binary cases. Dudoit *et al.* [36] performed a selection of genes for multi-class cases based on the ratio:

$$\frac{BSS(j)}{WSS(j)} = \frac{\sum_i \sum_k I(y_i = k)(\bar{x}_{kj} - \bar{x}_{.j})^2}{\sum_i \sum_k I(y_i = k)(\bar{x}_{ij} - \bar{x}_{kj})^2},$$

where $\bar{x}_{.j}$ denotes the average expression level of gene j across all samples and $\bar{x}_{kj}$ denotes the average expression level of gene j across samples belonging to class k . They selected m genes with the largest BSS/WSS ratios.

Other standard statistical methods, such as analysis of variance (ANOVA) [58], Mann-Whitney U-test [61] and Wilcoxon rank test [100], can be easily applied in a similar way.

The individual ranking methods, especially Weighting factor, are widely used in the literature of microarray data analysis. However, they remain within the confines of the old one-gene-at-a-time paradigm. These methods make an implicit assumption that all genes are uncorrelated. Actually, most, if not all, genes act in concert with

others. What DNA microarrays are really after is the expression pattern and mutual regulation across multiple genes. To detect such patterns, we need more sophisticated gene selection algorithms, taking into account mutual information among multiple genes.

3.3.3 Gene Pair Ranking

Bø and Jonassen [18] proposed a novel feature subset selection method by analyzing pairs of genes. They assigned each pair of genes a score, called pair t-score, to reflect how well the pair of genes distinguishes the two classes.

In [18], two alternative methods were employed for selecting a gene subset based on gene pair t-score, one exhaustive method called “all pairs” and one faster method “greedy pairs”. In the all pairs approach the pair with the highest pair t-score among the combinations from all genes is selected, then the top pair among the combinations from remaining genes is chosen, and so on until a desired number of genes are selected. The greedy pairs approach first ranks all genes on the basis of individual t-statistic. The best gene g_i is selected, then the complementary gene g_j that together with g_i has the highest pair t-score is chosen. The procedure is repeated on the remaining gene set until the desired number of genes are selected. Bø and Jonassen showed that their methods outperform individual gene ranking [18].

Compared with individual gene ranking, this method may select some complementary genes that individually do not separate the data well. Although the gene pair ranking method takes into account the dependence within the gene pair, it ignores mutual relationships between gene pairs. And this method, as well as individual gene ranking, does not yield a compact gene subset because many selected genes are redundant.

3.3.4 Gene Selection by GA/KNN

Li *et al.* [69] proposed a “Genetic Algorithm/ k -Nearest Neighbor” method for gene assessment and sample classification. This method employs a non-parametric pattern classification approach, k -nearest neighbor (KNN), to distinguish the discriminative genes, and employs a search procedure, genetic algorithm (GA), to choose the gene subsets for testing. The main idea of GA/KNN is to find a huge number of optimal or near-optimal subsets, and to assess the importance of genes for classification by examining the frequency of gene memberships in those subsets. The role of the genetic algorithm used in the method is not to find the specific optimal subset, but to facilitate the search procedure of finding a huge number of optimal or near-optimal subsets so as to compute the frequency of genes. The method gives emphasis to combinations of genes rather than individual dominant genes for sample classification. Note that the selected gene subset is chosen on the basis of the frequency of genes in the optimal and near-optimal subsets. Therefore, the selected subset could not be compact as many redundant genes may also present at a high frequency.

3.3.5 Gene Selection based on Support Vector Machines

Support vector machines (SVMs) have recently been intensively studied. SVM maximizes the distance between a hyperplane, $\mathbf{w}$, and the closest samples to the hyperplane from two classes. Relevant studies have shown that SVMs outperform other classification techniques on extremely scarce samples in a high dimensional space, especially microarray data (see for instance of [41]). Because of their good performance on microarray data, SVMs received more attention on application of gene selection. Some authors conjectured that SVMs have an implicit feature selection mechanism relying on support vectors [47], and several gene selection methods based

on SVMs have been developed.

3.3.5.1 Recursive Gene Ranking

Zhang *et al.* [121] proposed a gene selection method that ranks genes according to their contributions. The intuitive idea behind this method is to check the contribution of a single gene to the classification by determining whether this gene can enlarge the difference between samples of the two classes or not. Zhang *et al.* define the criterion for gene ranking as $r_i = |w_i(m_i^+ - m_i^-)|$, where m_i^+ and m_i^- are two class means of gene i , and w_i is the coefficient of gene i in the linear classifier, $f(\mathbf{x}) = \sum_{i=1}^n w_i x_i + b$. A recursive procedure was used in the gene selection. The procedure starts from all genes. Firstly, an optimal SVM classifier is built, and the genes are ranked according to the criterion above. A subset consisting of top genes is selected. Secondly, a new SVM model based on these selected genes is built and the selected genes are re-ranked and selected again. This classification - ranking - selection procedure is repeated until a desired small subset of genes is selected.

3.3.5.2 SVM Recursive Feature Elimination (RFE)

SVM-RFE [47] is a similar gene selection approach to the above algorithm [121]. SVM-RFE is a top-down feature elimination procedure, using the weight magnitude as the ranking criterion. The criteria, w_i^2 , estimate the effect of removing the corresponding feature. The feature with the smallest ranking criterion is removed. The above procedure is repeated until a small subset of features is obtained. This iterative procedure is an instance of backward feature selection. To reduce the computational cost, the procedure can be performed by removing several features at one time. Rakotomamonjy [90] explained the reason of using w_i^2 as a feature ranking criterion from another aspect. In the linear case, the derivative of $\|\mathbf{w}\|^2$ with respect

to a virtual scaling factor is equal to $-w_i^2$, the same as the SVM-RFE criterion. But the criterion of gradient of $\|\mathbf{w}\|^2$ can be extended to non-linear cases. SVM-RFE will be discussed in details in Chapter 7.

Ramaswamy *et al.* [91] combined RFE with one-versus-all (OVA) support vector machines for classification and gene selection to investigate multi-class cancer diagnosis, and obtained satisfactory performance on classification for 14 common human cancer classes.

3.3.5.3 SVM Feature Selection based on Discriminative Function Pruning Analysis (DFPA)

The above two approaches are based on linear classification, which limits their extensions to other applications. Mao [77] proposed a feature selection approach through discriminative pruning analysis based on linear or non-linear SVMs. In this method, support vectors and those near the boundaries are considered more important for feature selection than other samples. A criterion for pruning analysis

$$J = \sum_{i=1}^n \{f(\mathbf{x}_i) - g(\mathbf{z}_i)\}^2, \quad (3.2)$$

where $\mathbf{z}_i$ is a vector formed by the selected feature subset of the original sample $\mathbf{x}_i$, is employed to evaluate the performance of selected subsets. In Eq. (3.2), $f(\mathbf{x})$ is the optimal classifier induced by all features, and $g(\mathbf{z})$ is the linear least square estimate for $f(\mathbf{x})$,

$$f(\mathbf{x}) = \sum_{i=1}^L w_i K(\mathbf{x}_i, \mathbf{x}) + b, \quad \text{and} \quad g(\mathbf{z}) = \sum_{i=1}^L v_i K(\mathbf{z}_i, \mathbf{z}) + a,$$

$\mathbf{x}_i, i = 1, \dots, L$ are support vectors, and $\mathbf{z}_i$'s are vectors of selected feature subset of corresponding support vectors $\mathbf{x}_i$. Mao utilized sequential forward selection, com-

combined with minimizing the above criterion, for feature subset selection. The method evaluates features on a collective basis, and it reduces the computational cost by replacing the quadratic programming problem of inducing the SVM classifier in the feature subset with a simple linear regression problem.

3.4 Selection Bias

A large number of gene selection methods were proposed in the literature of microarray data analysis. However, almost all the algorithms have their own advantages and disadvantages. It is of great importance to select a suitable gene selection method in real applications. Therefore, how to evaluate the performance of a gene selection method is very significant in practice.

To assess the performance of the gene selection method, some researchers employed the entire dataset for gene selection, and the performance of selected genes was tested using k -fold cross validation. This kind of cross validation, called internal cross validation (see Algorithm 3), produces a biased estimate. Ambroise and McLachlan [11] investigated the magnitude of the selection bias, and suggested techniques of external (10-fold) cross validation (see Algorithm 4) and external .632+ bootstrap (see Algorithm 6 in Chapter 5), in which gene selection and validation are carried out on different parts of the sample set, to obtain an unbiased estimate. Details of using cross validation and bootstrap for error estimate are stated in Chapter 4.

The selection bias of internal estimate is illustrated in Fig. 3.2, which is excerpted from [11]. The error rate of internal cross validation is almost zero when 16 genes are selected. However, the true prediction error rate, TE, is not negligible, being above or approximate 5% for all selected subsets. The selection bias of internal cross validation cannot be ignored. It can be seen from the figure that the estimation of the prediction error by either external 10-fold cross validation or external bootstrap

Algorithm 3: Internal cross validation

Input: Dataset D , k (number of folds for cross validation)**Output:** error ratePerforming gene selection on the dataset D to identify gene subset S ;// $D(S)$ represents the dataset only using selected genes S $D(S)$ is divided into k subsets $D_i(S)$, $i = 1, \dots, k$ of equal size;**for** $i \leftarrow 1$ **to** k **do** $D_i(S)$ is considered as test set; $D(S) - D_i(S)$ is considered as training set, T_i ; Train the classifier on the dataset T_i , and test on $D_i(S)$;**end**Average the error rates on k folds;

Algorithm 4: External cross validation

Input: Dataset D , k (number of folds for cross validation)**Output:** error rate D is divided into k subsets D_i , $i = 1, \dots, k$ of equal size;**for** $i \leftarrow 1$ **to** k **do** D_i is considered as test set; $D - D_i$ is considered as training set, T_i ; Performing gene selection on the dataset T_i to identify gene subset S_i ; // $T_i(S_i)$ and $D_i(S_i)$ represent the training set and test set only
 using selected genes S_i Train the classifier on the dataset $T_i(S_i)$, and test on $D_i(S_i)$;**end**Average the error rates on k folds;

corrects for the selection bias, and presents close results to the true prediction error rate TE.

Figure 3.2: Illustration of selection bias and its correction with SVM RFE on the leukaemia dataset. The dataset is randomly split into a training set and a test set of 38 and 34 samples, respectively, for 50 times. The training set is used to carry out gene selection and to form the apparent error rate (AE), internal leave-one-out cross validation error rate (CV11E), external 10-fold cross validation error rate (CV10E) and external .632+ bootstrap error rate (B.632+). An unbiased error rate (TE) is estimated by the test set. The figure is from [11].

Why is the internal estimate for the gene selection method so severely biased? The reason is the improper use of the information in the test set. The internal estimate employs all the data for gene selection, and tests the performance of selected genes by using k -fold cross validation or bootstrap on the same dataset. Thus the dataset is employed twice, not only in gene selection procedure but also in the validation process. The test data used in the validation process should be kept unseen during the classifier training procedure (including gene selection). However, the test data are involved in the gene selection process in advance. This is the main cause of the selection bias. Because the genes are selected from the entire dataset, the selected

genes must contain almost all important information for classification of the dataset. Thus, if the same dataset is used to test the performance of the selected genes, the error rate would be much lower than the true error rate. For the external techniques, the gene selection and test process are performed on different parts of the dataset. In general, the error rate of external estimate is much higher than that of internal version.

Considering the high variance problem of cross validation, especially for the small-sample microarray data [21], we employ the external .632+ bootstrap [11, 37] to evaluate the performance of our gene selection algorithms in Chapters 5 ~ 7.

Chapter 4

The Ties Problem Resulting from Counting-based Error Estimators and its Impact on Gene Selection Algorithms

4.1 Introduction

As we mentioned in Chapter 3, feature selection approaches can be divided into two broad groups, filter methods and wrapper methods. In most pattern classification applications the wrapper method provides features with better predictive accuracy than the filter method does, whereas the computational cost of the wrapper method is much higher than that of the filter method. Both approaches have been used in microarray data analysis. For example, Furey *et al* [41], Golub *et al* [45] and West *et al* [114] employed filter methods, while Cho *et al.* [29], Guyon *et al* [47] and Inza *et al* [54] employed wrapper methods for gene selection.

The goodness of a feature subset is often evaluated by measuring its predictive capacity. Hence it is natural to consider the error rate or error probability as an ideal criterion in the wrapper-based feature selection [57,62]. As a large independent test set is rarely available in practice, the error rate is usually estimated based on the training data. There are several approaches to error estimation, including resubstitution, cross-validation and bootstrap. The leave-one-out error, a special case of cross-validation error, gives an almost unbiased estimate of the probability of test error [73,120]. Therefore, several authors employed the leave-one-out error as the evaluation criterion in the wrapper-based gene selection, and achieved satisfactory performance according to their reported experimental results [54,69,84,118].

Recent study [20] found, however, that leave-one-out error estimator and other three error estimators (resubstitution, cross-validation and bootstrap) exhibit excessive variability in small-sized data, which could affect the performance of wrapper-based feature selection [97] or feature-set ranking [98]. In this chapter, we reveal another problem of counting-based error estimators. We find that in high dimensional and small-sized microarray data, employing counting-based error estimators as gene evaluation criteria could result in severe ties problem, *i.e.* more than one gene subset score equally. In the occurrences of ties, the evaluation criterion is unable to differentiate these gene subsets, and the selection decision is usually made in terms of the order in which the candidate gene subsets are evaluated. In a sense, this selection is arbitrary, and less significant genes could be selected while more significant genes could be discarded. Our analysis finds that the root cause of the ties problem lies in the discrete nature of the counting-based error estimators. For microarray data, the counting-based error estimators just have a very limited number of possible values due to the small number of samples available, while the number of gene subset evaluated is huge because of the excessive high dimensionality. The mapping from gene subsets to error rates is thus many-to-one, and it is highly possible that more than one gene subsets have the same minimal error at some steps of the gene selection

procedure.

For any data, it is possible that there are ties in the estimated error among different gene subsets when the counting-based error estimator is employed as the evaluation criterion, but the probability of the occurrence of ties is limited in conventional data (low dimensionality and large or medium sample size). We believe it is the reason that the ties problem has not received many attentions in the literature of feature selection. However, the high dimensionality and small sample size of microarray data drastically enhance this probability, which results in an arbitrary gene selection. As the high dimensional and small-sized settings are common for biological data, such as microarray data and mass spectrometry data, our finding is significant for successful exploration of biological applications.

To solve the ties problem, we suggest employing well-designed continuous criteria, which could always have different values for different feature subsets although the difference may not necessarily be substantial, to replace the counting-based error estimators. In the present study, we examined two continuous evaluation criteria including generalized $\|\mathbf{w}\|^2$ measure for Support Vector Machines and modified Relief's measure for k -Nearest Neighbor respectively. Experimental results show that well-designed continuous evaluation criteria could eliminate the uncertainty raised by the ties problem and improve gene selection performance.

4.2 Classification Algorithms

Consider n training data pairs: $S = \{(\mathbf{x}_i, y_i)\}$, $i = 1, \dots, n$, where $\mathbf{x}_i$ is a feature vector representing the i 'th sample, and y_i is the class label of $\mathbf{x}_i$. In this work, we only consider the two-class problem, *i.e.*, $y_i \in \{-1, 1\}$.

A classifier is a function that maps an unlabelled sample $\mathbf{x}$ to a label y . The

classification algorithm builds a classifier from training data. In this classification problem the goal is to find a classifier which can separate the available examples and at the same time expect to work well on unseen examples. Here, we review two classification algorithms: Support vector machine and k -Nearest neighbor.

4.2.1 Support Vector Machines

4.2.1.1 Linearly Separable Cases

Consider the example in Fig. 4.1. There are many possible linear classifiers which can separate the training data. Intuitively, the one with the maximal distance between itself and the closest samples from two classes is expected to generalize well as opposed to the other possible classifiers.

Figure 4.1: Optimal separating hyperplane

The support vector machine separates samples of the two classes by an optimal hyperplane which is described by:

$$\mathbf{w}^T \mathbf{x} + b = 0 \quad (4.1)$$

where $\mathbf{w} = [w_1, w_2, \dots, w_n]^T$, and b is a scalar. The optimal hyperplane is the one

with the maximal distance to the closest samples from the training set [112].

Figure 4.2: Linear separating hyperplane of SVM. The support vectors are circled. The figure is from [25]

For a separable case, we can constrain $\mathbf{w}$ and b by

$$y_i[\mathbf{w}^T \mathbf{x}_i + b] \geq 1, \quad i = 1, \dots, n \quad (4.2)$$

As shown in Fig. 4.2, the separating margin

$$\rho(\mathbf{w}) = \frac{2 \min |\mathbf{w}^T \mathbf{x} + b|}{\|\mathbf{w}\|} = \frac{2}{\|\mathbf{w}\|} \quad (4.3)$$

Hence the hyperplane that optimally separates the samples is the one that satisfies the following optimization problem,

$$\begin{aligned} & \text{minimize} \quad \mathcal{P}(\mathbf{w}) = \frac{1}{2} \|\mathbf{w}\|^2, \\ & \text{subject to} \quad y_i[\mathbf{w}^T \mathbf{x}_i + b] \geq 1, \quad i = 1, \dots, n \end{aligned} \quad (4.4)$$

We now switch to a Lagrangian formulation of the problem. For the details about

Lagrangian Optimization Theory, please refer to Appendix A. The Lagrangian function is,

$$\mathcal{L}(\mathbf{w}, b, \boldsymbol{\alpha}) = \frac{1}{2} \|\mathbf{w}\|^2 - \sum_{i=1}^n \alpha_i [y_i(\mathbf{w}^T \mathbf{x}_i + b) - 1] \quad (4.5)$$

where $\alpha_i \geq 0$ are the Lagrange multipliers. The Lagrangian $\mathcal{L}$ has to be minimized with respect to $\mathbf{w}$ and b , and simultaneously maximized with respect to all the α_i , which subject to the constraints $\alpha_i \geq 0$. To find the minimum over $\mathbf{w}$ and b , we require,

$$\frac{\partial \mathcal{L}(\mathbf{w}, b, \boldsymbol{\alpha})}{\partial \mathbf{w}} = 0 \Rightarrow \mathbf{w} = \sum_{i=1}^n y_i \alpha_i \mathbf{x}_i, \quad (4.6)$$

$$\frac{\partial \mathcal{L}(\mathbf{w}, b, \boldsymbol{\alpha})}{\partial b} = 0 \Rightarrow \sum_{i=1}^n y_i \alpha_i = 0. \quad (4.7)$$

Eq. (4.6) indicates that the solution of the optimization problem (4.4) is a linear combination of the training samples, $\mathbf{x}_1, \dots, \mathbf{x}_n$. By substituting Eqs. (4.6) and (4.7) into the Lagrangian (4.5), we convert the optimization problem (4.4) into the following dual problem,

$$\begin{aligned} &\text{maximize} \quad \mathcal{W}(\boldsymbol{\alpha}) = \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n y_i y_j \alpha_i \alpha_j \mathbf{x}_i^T \mathbf{x}_j \\ &\text{subject to} \quad \sum_{i=1}^n y_i \alpha_i = 0, \\ &\quad \alpha_i \geq 0, \quad i = 1, \dots, n. \end{aligned} \quad (4.8)$$

It is a quadratic programming (QP) optimization problem. Now to construct the optimal classifier we have to find the Lagrange multipliers α_i^* that maximize the Eq. (4.8). Using α_i^* , we obtain the solution for Eq. (4.4),

$$\mathbf{w}^* = \sum_{i=1}^n y_i \alpha_i^* \mathbf{x}_i \quad (4.9)$$

Note that, while $\mathbf{w}$ is explicitly determined by the dual problem (4.8), the threshold b is not. However b can be easily found by using the Karush-Kuhn-Tucker (KKT) complementarity conditions (see Appendix A). The conditions state that the optimal solutions α^* , $\mathbf{w}^*$, b^* must satisfy

$$\alpha_i^* [y_i (\mathbf{w}_i^{*T} \mathbf{x}_i + b^*) - 1] = 0, \quad i = 1, \dots, n. \quad (4.10)$$

Therefore, for the sample $\mathbf{x}_i$ whose $\alpha_i^* \neq 0$, $b^* = y_i - \mathbf{w}_i^{*T} \mathbf{x}_i$.

Let

$$\begin{aligned} f(\mathbf{x}) &= \mathbf{w}^{*T} \mathbf{x} + b^* \\ &= \sum_{i=1}^n y_i \alpha_i^* \mathbf{x}_i^T \mathbf{x} + b^*, \end{aligned} \quad (4.11)$$

then the decision rule for sample $\mathbf{x}$ is given by $\text{sign}(f(\mathbf{x}))$.

The conditions (4.10) also imply that only for samples $\mathbf{x}_i$ that lie on the optimal hyperplane, the corresponding α_i^* are non-zero. All the other Lagrange multipliers α_i^* are zero. Hence, in the expression for the weight vector $\mathbf{w}$ in Eq. (4.9), only these samples with non-zero α_i^* are involved. It is for this reason that they are called support vectors, as shown in Fig. 4.2.

4.2.1.2 Linearly Non-separable Cases

In many real-world problems, there will in general be no linear separation in the feature space. For these cases, the optimization problem (4.4) cannot be solved as it has an empty feasible region. In order to solve the problem, we need to introduce

slack variables as follows, to allow the constraints to be violated.

$$\begin{aligned} \text{subject to} \quad & y_i[\mathbf{w}^T \mathbf{x}_i + b] \geq 1 - \xi_i, \quad i = 1, \dots, n, \\ & \xi_i \geq 0, \quad i = 1, \dots, n. \end{aligned} \quad (4.12)$$

Then two kinds of soft margin optimization problems are addressed to solve the non-separable cases:

$$\begin{aligned} \text{minimize} \quad & \mathcal{P}(\mathbf{w}, b, \boldsymbol{\xi}) = \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^n \xi_i, \\ \text{subject to} \quad & y_i[\mathbf{w}^T \mathbf{x}_i + b] \geq 1 - \xi_i, \quad i = 1, \dots, n, \\ & \xi_i \geq 0, i = 1, \dots, n. \end{aligned} \quad (4.13)$$

and

$$\begin{aligned} \text{minimize} \quad & \mathcal{P}(\mathbf{w}, b, \boldsymbol{\xi}) = \frac{1}{2} \|\mathbf{w}\|^2 + \frac{C}{2} \sum_{i=1}^n \xi_i^2, \\ \text{subject to} \quad & y_i[\mathbf{w}^T \mathbf{x}_i + b] \geq 1 - \xi_i, \quad i = 1, \dots, n, \\ & \xi_i \geq 0, i = 1, \dots, n. \end{aligned} \quad (4.14)$$

where the parameter C is a trade-off between training accuracy and generalization. The first one, Eq. (4.13), is called 1-Norm Soft Margin optimization, while the second, Eq. (4.14), is 2-Norm Soft Margin optimization. In this chapter, we only consider the 1-Norm Soft Margin SVMs, and the 2-Norm Soft Margin SVMs will be presented in Chapter 7.

The corresponding Lagrangian for the 1-norm soft margin optimization problem

(4.13) is

$$\begin{aligned}\mathcal{L}(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = & \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^n \xi_i \\ & - \sum_{i=1}^n \alpha_i [y_i(\mathbf{w}^T \mathbf{x}_i + b) - 1 + \xi_i] - \sum_{i=1}^n \beta_i \xi_i\end{aligned}\quad (4.15)$$

where $\alpha_i \geq 0$ and $\beta_i \geq 0$. We obtain the following dual problem,

$$\begin{aligned}\text{maximize } \mathcal{W}(\boldsymbol{\alpha}) = & \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n y_i y_j \alpha_i \alpha_j \mathbf{x}_i^T \mathbf{x}_j \\ \text{subject to } & \sum_{i=1}^n y_i \alpha_i = 0, \\ & 0 \leq \alpha_i \leq C, \quad i = 1, \dots, n.\end{aligned}\quad (4.16)$$

After finding the Lagrange multipliers α_i^* that maximize Eq. (4.16), we obtain the solution for optimization problem (4.13),

$$\begin{aligned}\mathbf{w}^* &= \sum_{i=1}^n y_i \alpha_i^* \mathbf{x}_i, \\ b^* &= y_i - \mathbf{w}^{*T} \mathbf{x}_i, \quad \text{for any } i \text{ with } 0 < \alpha_i^* < C.\end{aligned}$$

The decision function is,

$$\begin{aligned}f(\mathbf{x}) &= \mathbf{w}^{*T} \mathbf{x} + b^* \\ &= \sum_{i=1}^n y_i \alpha_i^* \mathbf{x}_i^T \mathbf{x} + b^*.\end{aligned}\quad (4.17)$$

So surprisingly the solution of this problem is equivalent to the separable cases with an additional constraint that all the $\boldsymbol{\alpha}$ are upper bounded by C .

4.2.1.3 Non-linear Cases

Note that the only way in which the training data appear in the solution, for example, Eqs. (4.16) and (4.17) for non-separable cases, is in the form of inner products, $\mathbf{x}_i^T \mathbf{x}_j$. Now suppose we map the training data to some other (higher dimensional) Euclidean space $\mathcal{H}$, using a mapping which we will call Φ :

$$\Phi : \mathbb{R}^p \mapsto \mathcal{H} \quad (4.18)$$

Then the training procedure would only depend on the data through inner products in $\mathcal{H}$, that is, on the functions of the form $\Phi(\mathbf{x}_i)^T \Phi(\mathbf{x}_j)$. If there were a “kernel function” k such that $k(\mathbf{x}_i, \mathbf{x}_j) = \Phi(\mathbf{x}_i)^T \Phi(\mathbf{x}_j)$, we would only need to use $k(\cdot, \cdot)$ in the training procedure, and would never need to explicitly know what the space $\mathcal{H}$ is.

So using the kernel trick, Support Vector Machines can be generalized to the non-linear cases. For example, the decision function of 1-Norm Soft Margin SVM is in the following form,

$$f(\mathbf{x}) = \sum_{i=1}^n y_i \alpha_i^* k(\mathbf{x}_i, \mathbf{x}) + b^*, \quad (4.19)$$

where α_i^* are obtained from the QP optimization problem,

$$\begin{aligned} & \text{maximize} \quad \mathcal{W}(\boldsymbol{\alpha}) = \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n y_i y_j \alpha_i \alpha_j k(\mathbf{x}_i, \mathbf{x}_j) \\ & \text{subject to} \quad \sum_{i=1}^n y_i \alpha_i = 0, \\ & \quad \quad \quad 0 \leq \alpha_i \leq C, \quad i = 1, \dots, n, \end{aligned} \quad (4.20)$$

and $b^* = y_j - \sum_{i=1}^n y_i \alpha_i^* k(\mathbf{x}_i, \mathbf{x}_j)$, for any j with $0 < \alpha_j^* < C$.

The idea of using kernel trick to deal with non-linear classification problem is really attractive, and broadens the applications of SVMs on real-world problems. In

Chapter 5, we will combine this trick with Regularization Networks to reduce the influence of absence of bias b .

4.2.2 k -Nearest Neighbors

The k -Nearest Neighbors rule (KNN) is a simple but powerful classification algorithm [50]. Nearest neighbor methods are based on a *distance function* for pairs of samples. In this work, the Euclidean distance is employed as the *distance function* to compute the similarity among samples. When classifying an unlabelled sample $\mathbf{x}$, KNN selects k closest samples in the training set $S = \{(\mathbf{x}_i, y_i)\}$, and predicts the class of sample $\mathbf{x}$ by majority vote, that is, chooses the class that is most common among those k neighbors.

We denote $\mathbf{n}^j$ ($j = 1, \dots, k$) as the nearest neighbors of unlabelled sample $\mathbf{x}$. For two-class problem, where the class label is either $+1$ or -1 , the predicted label of $\mathbf{x}$ can be expressed as

$$L(\mathbf{x}) = \text{sign}\left(\sum_{i=1}^k L(\mathbf{n}^i)\right), \quad (4.21)$$

where $L(\cdot)$ denoted as the class label or predicted class label.

The k -Nearest Neighbors rules are different from the Support Vector Machines. In SVMs, the decision function induced from the training data can be expressed as $f(\mathbf{x}) = \sum_{i=1}^n y_i \alpha_i^* \mathbf{x}_i^T \mathbf{x} + b^*$, while there is no explicit training phase in the process of KNN classification. Actually, the training phase of KNN is delayed until the sample which need to be classified is received. This kind of classification algorithm is called lazy learning.

4.3 Evaluation Criteria

Error rate, the most commonly used evaluation criterion in wrapper methods, is usually estimated from training data based on error-counting mechanism. Criteria that do not count errors explicitly but reflect error probability implicitly are also seen in other wrapper methods. For example, SVM-RFE [47] employs separating margin, which implicitly reflects error probability, as gene evaluation criterion. In this section, counting-based error estimators and continuous measures implicitly reflecting error probability are reviewed and discussed.

4.3.1 Counting-based Error Estimation

A classifier is a function that maps an unlabelled sample $\mathbf{x}$ to a label y . We use notation $\mathcal{C}_S(\mathbf{x})$ to denote the class label assigned to sample $\mathbf{x}$ by the classifier built on training dataset S .

The error rate of a classifier is the probability of incorrectly classifying a randomly selected sample. The true error rate depends on the feature-label distribution. If the feature-label distribution were known, the true error could be computed exactly. However, the distribution is often unknown in practice, and the error rate has to be estimated from the given dataset. Most of the commonly used error estimators, including resubstitution, cross-validation and bootstrap, are based on a certain form of error counting.

The resubstitution error, $\hat{\epsilon}_{resub}$, also called apparent error, is the ratio of the number of misclassifications on the training dataset S to the number of samples in the dataset:

$$\hat{\epsilon}_{resub} = \frac{1}{n} \sum_{i=1}^n I(\mathcal{C}_S(\mathbf{x}_i) \neq y_i), \quad (4.22)$$

where $I(A) = 1$ if the statement A is true and 0 otherwise. Resubstitution error

usually gives a severely biased estimate of the true error on small-sized datasets.

In k -fold cross-validation, the dataset S is randomly split into k mutually exclusive folds S_t , $t = 1, \dots, k$, of nearly equal size. The training and test procedures are performed k times. Each time $t \in 1, \dots, k$, the classifier is trained on $S \setminus S_t$ (the notation “ $\setminus$ ” denotes operation of set subtraction) and tested on S_t . The cross-validation error rate is the ratio of overall misclassifications of k test subsets to the number of samples in the dataset:

$$\hat{e}_{kcv} = \frac{1}{n} \sum_{i=1}^n I(C_{S \setminus S_{(i)}}(\mathbf{x}_i) \neq y_i), \quad (4.23)$$

where $S_{(i)}$ is the fold that includes sample $\langle \mathbf{x}_i, y_i \rangle$. To obtain a more reliable estimate, the cross-validation process may be repeated multiple times using different splits of the data.

The leave-one-out error is a special case of k -fold cross-validation error, with $k = n$:

$$\hat{e}_{loo} = \frac{1}{n} \sum_{i=1}^n I(C_{S \setminus \langle \mathbf{x}_i, y_i \rangle}(\mathbf{x}_i) \neq y_i). \quad (4.24)$$

It has been proved that the leave-one-out error is almost an unbiased estimate of the true error [73, 120].

In the bootstrap error estimation, a bootstrap sample set S^* is created by sampling n samples uniformly from the original dataset S with replacement. Some of the samples in S may appear once or multiple times in S^* , whereas others may not appear at all. In practice, the bootstrap error is approximated on a total of B independent bootstrap sample sets S_b^* , for $b = 1, \dots, B$. Define $J_b^i = 1$ if sample $\langle \mathbf{x}_i, y_i \rangle$ is not contained in the b 'th bootstrap sample set S_b^* , and 0 otherwise. The

bootstrap error estimate on the basis of the B bootstrap sample sets is given by

$$\hat{\epsilon}_{B1} = \frac{1}{n} \sum_{i=1}^n \frac{\sum_{b=1}^B I(\mathcal{C}_{S_b^*}(\mathbf{x}_i) \neq y_i) J_b^i}{\sum_{b=1}^B J_b^i}. \quad (4.25)$$

$\hat{\epsilon}_{B1}$ is upwardly biased, and Efron and Tibshirani [37] proposed the .632+ bootstrap estimator to correct the upward bias:

$$\hat{\epsilon}_{B.632+} = (1 - \hat{w}) \cdot \hat{\epsilon}_{resub} + \hat{w} \cdot \hat{\epsilon}_{B1}, \quad (4.26)$$

where the weight $\hat{w}$ is given by $\hat{w} = \frac{0.632}{1 - 0.368 \cdot \hat{R}}$, $\hat{R} = \frac{\hat{\epsilon}_{B1} - \hat{\epsilon}_{resub}}{\hat{\lambda} - \hat{\epsilon}_{resub}}$ is the relative overfitting rate, and $\hat{\lambda}$ is the no-information error rate that would apply if the feature vector $\mathbf{x}$ and label y were independent. For the two-class problem, $\hat{\lambda}$ is estimated by $\hat{\lambda} = \hat{p}_1(1 - \hat{q}_1) + (1 - \hat{p}_1)\hat{q}_1$, where $\hat{p}_1$ is the proportion of label y_i equaling 1, and $\hat{q}_1$ is the proportion of $\mathcal{C}_S(\mathbf{x}_i)$ equaling 1. In this study, .632+ bootstrap [37] is employed.

All the error estimators mentioned above are based on 0–1 decision in error counting. That is, if $\mathcal{C}(\mathbf{x}_i) \neq y_i$, one error is counted, otherwise zero error. For resubstitution and leave-one-out error rates, the possible values are $\frac{0}{n}, \frac{1}{n}, \frac{2}{n}, \dots$. Obviously, the two error estimators are discrete functions with at most $n + 1$ possible values for a dataset with n samples. Cross-validation and bootstrap errors could have more possible values, but the two error estimators are still discrete functions in nature.

A study in [97] showed that the four counting-based error estimators exhibit variability which has impacts on feature selection. Next, we analyze that the discrete nature of the counting-based error estimator could result in the ties problem in gene selection due to small sample size and high dimensionality of microarray data, and this in turn induces uncertainty to gene selection.

Without loss of generality, we take the leave-one-out error based wrapper method as an example. At every step of the feature selection procedure, no matter what search procedure is used, a number of candidate feature subsets are evaluated. For the high dimensional microarray data, the number of candidate feature subsets to be evaluated is huge. For example, at every step of sequential forward gene selection (SFS), the number of candidate gene subsets is almost equal to the dimensionality of the dataset (usually a few thousands). However, the number of possible values of the leave-one-out error is very limited, say 10, no matter which classification algorithm is used. This is just like the following scenario: thousands of balls are randomly tossed into 10 boxes marked 0 to 9. Obviously, it is almost impossible that the number of balls falling into box 0 is only 1. This example indicates that at some steps of the gene selection procedure, there might exist a few candidate gene subsets producing the same minimal error. If this ties problem occurs, the error rate as an criterion is unable to judge which gene subset is the best, and the selection decision is often made in terms of the order in which the candidate gene subsets are evaluated. In a sense, the selection loses its way and become arbitrary. As a direct result, some less significant genes might be selected while more important ones might be discarded.

For conventional data with low dimensionality and large or medium sample size, the ties problem is not a major issue. But in microarray data which is of small size and high dimensionality, the ties problem often occurs. This will be demonstrated in experimental studies.

4.3.2 Continuous Evaluation Criteria

As analyzed in Section 4.3.1, the main reason of the arbitrary gene selection is the discrete evaluation criterion. To overcome this problem, we suggest employing well-designed continuous criterion, which maps different gene subsets to different

values, to replace the discrete one. Two continuous measures are investigated for wrapper-based gene selection in the present study.

Generalized $\|\mathbf{w}\|^2$ Measure for Support Vector Machines

Support vector machines (SVMs) are well suited to work with high dimensional data, such as microarray data [41]. When used for classification, SVMs separate one class from the other with a hyperplane that maximizes the distance between the hyperplane and the nearest sample of each class. For separable cases, SVMs maximize the separating margin $\rho(\mathbf{w}) = 2/\|\mathbf{w}\|$. The mechanism of SVMs is to minimize the $\frac{1}{2}\|\mathbf{w}\|^2$, and hence to maximize the separating margin. In feature selection applications, the feature subset with minimal $\frac{1}{2}\|\mathbf{w}\|^2$, which leads to the maximal separating margin, is preferred. For non-separable cases, SVM is generalized to minimize the following criterion to implement the 1-Norm Soft Margin optimization,

$$\begin{aligned} \text{minimize} \quad & \mathcal{P}(\mathbf{w}, \boldsymbol{\xi}) = \frac{1}{2}\|\mathbf{w}\|^2 + C \sum_{i=1}^n \xi_i, \\ \text{subject to} \quad & y_i[\mathbf{w}^T \mathbf{x}_i + b] \geq 1 - \xi_i, \quad i = 1, \dots, n, \\ & \xi_i \geq 0, i = 1, \dots, n. \end{aligned}$$

Therefore the $\frac{1}{2}\|\mathbf{w}\|^2$ can also be generalized to $\frac{1}{2}\|\mathbf{w}\|^2 + C \sum_{i=1}^n \xi_i$ for feature evaluation. The latter is called generalized $\|\mathbf{w}\|^2$ measure in this work.

Our goal here is not only to find the hyperplane with minimal generalized $\|\mathbf{w}\|^2$ measure, but also the feature subset that yields minimal generalized $\|\mathbf{w}\|^2$ measure. To achieve this goal, we first induce the SVM classifiers on all candidate feature subsets generated by the search procedure, then identify the feature subset with minimal generalized $\|\mathbf{w}\|^2$ measure. The well-known SVM-RFE [47] minimizes the generalized $\|\mathbf{w}\|^2$ measure as well (please refer to Chapter 7), and it uses the Optimal Brain Damage (OBD) algorithm [66] to perform a backward elimination procedure.

Modified Relief's Measure for k -Nearest Neighbors

The modified Relief's measure is inspired by Relief algorithm [64]. The key idea of Relief is to estimate the quality of a feature according to how well its value distinguishes between samples that are near to each other.

In this study, instead of evaluating the significance of a single feature, we propose the modified Relief's measure to assess the quality of a feature subset. The measure is similar to k -Nearest Neighbor rules. For a given sample pair $\langle \mathbf{x}_i, y_i \rangle$, two groups of nearest neighbors can be identified from the remaining data $(S \setminus \langle \mathbf{x}_i, y_i \rangle)$: k nearest neighbors from the same class as y_i , called nearest hits $\mathbf{h}_i^j$ for $j = 1, \dots, k$, and k nearest neighbors from the other class, called nearest misses $\mathbf{m}_i^j$, for $j = 1, \dots, k$. k is a predefined number. Let $\mathbf{h}_i^*$ be the center of nearest hits, $\mathbf{m}_i^*$ be the center of nearest misses, that is, $\mathbf{h}_i^* = (\sum_{j=1}^k \mathbf{h}_i^j)/k$ and $\mathbf{m}_i^* = (\sum_{j=1}^k \mathbf{m}_i^j)/k$. The modified Relief's measure is defined as

$$R(S) = \frac{1}{n} \sum_{i=1}^n [D(\mathbf{x}_i, \mathbf{h}_i^*) - D(\mathbf{x}_i, \mathbf{m}_i^*)], \quad (4.27)$$

where $D(\mathbf{u}, \mathbf{v})$ is the Euclidean distance between vectors $\mathbf{u}$ and $\mathbf{v}$. The modified Relief's measure can be considered as the modification of leave-one-out error for k -Nearest Neighbors. The smaller the modified Relief's measure, the lower the leave-one-out error supposed to be. Hence the feature subset with the minimal modified Relief's measure is preferred.

4.4 System and Methods

4.4.1 Datasets

The leukaemia dataset was first described in [45]. The dataset contains expression levels of 7129 human genes of 72 patients with either acute lymphoblastic leukaemia (ALL, 47 cases) or acute myeloid leukaemia (AML, 25 cases). The raw data are available at <http://www-genome.wi.mit.edu/cancer/>.

The second dataset is the breast cancer dataset [114]. The dataset consists of expression of 49 tumor samples for 7129 human genes. There are two different response labels in the dataset, the status of the estrogen receptor (ER) and the lymph nodal (LN). In this work, we only considered the former. Among the 49 samples, 25 samples are ER+, while the remaining samples are ER-. The data are available at <http://data.cgt.duke.edu/west.php>.

The third dataset is the colon cancer dataset [9]. It contains gene expression levels of 2000 genes of 40 tumor and 22 normal colon tissues. The data are publicly available at <http://microarray.princeton.edu/oncology/>.

Each of these datasets was pre-processed using the procedure described in [36]: (a) thresholding, floor of 100 and ceiling of 16,000; (b) filtering, exclusion of genes with $\max/\min \leq 5$ or $(\max - \min) \leq 500$, where $\max$ and $\min$ refer to the maximum and minimum intensities for a particular gene across all samples; (c) based 10 logarithmic transformation. After thresholding, filtering and logarithmic-transforming, the microarray data were standardized to zero mean and unit standard deviation across genes. Since the dimensionality (number of genes) of microarray data is very high, and most of the genes are irrelevant to the discriminant task, we employed a pre-selection procedure to reduce the search space and to save the computational cost. We selected top 1000 genes based on Fisher's ratio. Fisher's

ratio is an individual gene ranking criterion, and it is used to evaluate how well a single gene is correlated with the separation between classes. For each gene the Fisher's ratio is defined as $f = \frac{(\mu_1 - \mu_2)^2}{\sigma_1^2 + \sigma_2^2}$, where μ_1 , μ_2 , σ_1 and σ_2 denote means and standard deviations of two classes, respectively. All the simulations and comparisons in this work were carried out based on the pre-processed and pre-selected data.

4.4.2 Wrapper Schemes

As we described in Chapter 3, typical feature selection algorithm consists of four basic components: *search procedure*, *evaluation criterion*, *stopping criterion* and *validation procedure*. Since our focus is the performance of different evaluation criteria, the search procedure is fixed to sequential forward selection (SFS) [34]. SFS is a simple hill-climbing search algorithm. SFS starts from an empty subset of features, and sequentially adds features to the subset until the stopping criterion is reached (please refer to Section 3.2.2.2 for details). The stopping criterion in our work is a predefined number of genes, say 10.

In wrapper methods, the evaluation criterion is the performance of classification algorithm used in the validation procedure. In this work, two groups of evaluation criteria were investigated. The first group includes the counting-based error rate estimators, such as resubstitution error, leave-one-out error, cross-validation error and .632+ bootstrap error [37]. The other group includes two continuous criteria: generalized $\|\mathbf{w}\|^2$ measure for Support Vector Machines (SVMs) and modified Relief's measure for k -Nearest Neighbors (KNNs).

For different combinations of the four components in a typical feature selection algorithm, we investigated 10 cases as summarized in Tab 4.1.

Index of wrappers	Evaluation criterion	Search procedure	Stopping criterion	Validation procedure
1	resubstitution error	SFS	10 genes	SVM
2	leave-one-out error	SFS	10 genes	SVM
3	10-fold cv error	SFS	10 genes	SVM
4	.632+ bootstrap error	SFS	10 genes	SVM
5	generalized $\ \mathbf{w}\ ^2$ measure	SFS	10 genes	SVM
6	resubstitution error	SFS	10 genes	KNN
7	leave-one-out error	SFS	10 genes	KNN
8	10-fold cv error	SFS	10 genes	KNN
9	.632+ bootstrap error	SFS	10 genes	KNN
10	modified Relief's measure	SFS	10 genes	KNN

Table 4.1: Description of the components of 10 wrapper methods.

4.5 Experimental Studies on Microarray Datasets

In this section, we tested the performance of wrapper methods listed in Tab 4.1 on three microarray benchmark datasets. For SVMs, the linear SVM classifier was employed, and the parameter C was set to 0.1. For k -Nearest Neighbors and modified Relief's measure, the parameter k was set to 3.

We first demonstrate the existence of the ties problem, taking the leukaemia dataset as an example. In the experiment, SVM is employed as the classification algorithm in the validation procedure, and sequential forward wrapper selection is performed. As illustrated in Tab 4.2, a few gene subsets could lead to equal minimal estimated error at some steps of the gene selection procedure when any counting-based error estimator is used as the evaluation criterion. For example, in the case of leave-one-out error, after four genes were selected, we may identify 563 genes out of the remaining 996 candidate genes, and the addition of any one of the 563 genes into the gene subset selected at the previous step could lead to the same minimal leave-one-out error. Thus, the next selected gene could be any one of the 563 genes. The selection decision is usually made in terms of the order in which the gene subsets are evaluated. In a sense, this selection is arbitrary, and it is reasonable to believe

Criteria	Index of steps in SFS									
	1	2	3	4	5	6	7	8	9	10
$\hat{\epsilon}_{resub}$	3	65	8	1	695	959	633	756	822	772
$\hat{\epsilon}_{loo}$	2	65	10	3	563	3	397	596	684	463
$\hat{\epsilon}_{10cv}$	2	65	1	1	1	48	540	795	732	612
$\hat{\epsilon}_{B.632+}$	1	30	1	1	1	3	83	73	204	157
generalized $\ \mathbf{w}\ ^2$ measure	1	1	1	1	1	1	1	1	1	1

Table 4.2: The number of genes which lead to the same minimal criteria at each step of SFS for SVM-based wrapper methods on the leukaemia dataset.

Criteria	Index of steps in SFS									
	1	2	3	4	5	6	7	8	9	10
$\hat{\epsilon}_{resub}$	1	19	147	436	690	374	577	414	337	353
$\hat{\epsilon}_{loo}$	2	6	15	320	12	317	354	465	644	500
$\hat{\epsilon}_{10cv}$	2	2	1	1	115	4	3	242	504	448
$\hat{\epsilon}_{B.632+}$	1	1	1	1	2	51	1	1	1	2
modified Relief's measure	1	1	1	1	1	1	1	1	1	1

Table 4.3: The number of genes which lead to the same minimal criteria at each step of SFS for KNN-based wrapper methods on the leukaemia dataset.

that less significant genes could be selected while more significant ones could be discarded. However, the ties problem does not occur in the continuous criterion-based wrapper method as shown in Tab 4.2 (the last row is the results of generalized $\|\mathbf{w}\|^2$ measure). Actually, the ties problem also happens in leukaemia dataset for KNN rules, and in breast cancer and colon cancer datasets for both rules (shown in Tab. 4.3~4.7), except for the bootstrap estimator on the colon dataset. Note that the 10-fold cross-validation error was estimated by the average of 10 runs of 10-fold cross-validation, and the bootstrap was performed with $B = 100$ replicates, with the goal of keeping the number of induced classifiers in the bootstrap to be the same as 10 runs of 10-fold cross-validation.

Next, we demonstrate the impact of the ties problem. The experiment design is as follows. Firstly, genes of the original dataset (leukaemia dataset, breast cancer dataset or colon cancer dataset) were randomly permuted to generate a new dataset.

Criteria	Index of steps in SFS									
	1	2	3	4	5	6	7	8	9	10
$\hat{\epsilon}_{resub}$	1	2	48	12	1	625	374	689	491	392
$\hat{\epsilon}_{loo}$	1	2	31	1	283	379	253	112	778	609
$\hat{\epsilon}_{10cv}$	1	1	15	1	2	1	31	346	830	696
$\hat{\epsilon}_{B.632+}$	1	1	1	1	1	1	1	9	180	38
generalized $\ \mathbf{w}\ ^2$ measure	1	1	1	1	1	1	1	1	1	1

Table 4.4: The number of genes which lead to the same minimal criteria at each step of SFS for SVM-based wrapper methods on the breast cancer dataset.

Criteria	Index of steps in SFS									
	1	2	3	4	5	6	7	8	9	10
$\hat{\epsilon}_{resub}$	2	1	7	127	220	222	295	342	311	502
$\hat{\epsilon}_{loo}$	3	23	9	4	174	335	378	290	425	310
$\hat{\epsilon}_{10cv}$	1	7	8	3	119	119	276	390	325	343
$\hat{\epsilon}_{B.632+}$	1	1	1	1	1	77	163	42	59	54
modified Relief’s measure	1	1	1	1	1	1	1	1	1	1

Table 4.5: The number of genes which lead to the same minimal criteria at each step of SFS for KNN-based wrapper methods on the breast cancer dataset.

Criteria	Index of steps in SFS									
	1	2	3	4	5	6	7	8	9	10
$\hat{\epsilon}_{resub}$	1	2	1	707	601	627	500	555	1	466
$\hat{\epsilon}_{loo}$	1	1	2	473	358	1	200	421	472	537
$\hat{\epsilon}_{10cv}$	1	1	1	49	140	228	307	310	125	131
$\hat{\epsilon}_{B.632+}$	1	1	1	1	1	1	1	1	2	2
generalized $\ \mathbf{w}\ ^2$ measure	1	1	1	1	1	1	1	1	1	1

Table 4.6: The number of genes which lead to the same minimal criteria at each step of SFS for SVM-based wrapper methods on the colon cancer dataset.

Criteria	Index of steps in SFS									
	1	2	3	4	5	6	7	8	9	10
$\hat{\epsilon}_{resub}$	1	7	3	10	31	28	114	42	55	84
$\hat{\epsilon}_{loo}$	1	5	1	2	2	25	21	32	18	18
$\hat{\epsilon}_{10cv}$	1	1	1	1	5	2	4	6	3	2
$\hat{\epsilon}_{B.632+}$	1	1	1	1	1	1	1	1	1	1
modified Relief’s measure	1	1	1	1	1	1	1	1	1	1

Table 4.7: The number of genes which lead to the same minimal criteria at each step of SFS for KNN-based wrapper methods on the colon cancer dataset.

In the experiment, the procedure was repeated 10 times to generate 10 datasets. Note that the only difference among the 10 datasets is the different order of genes in the data matrix. Secondly, each of the 10 newly generated datasets was split into a training and a test set with nearly equal size and approximately the same proportions of labels as the entire dataset. The training set is used to perform gene selection and classifier training, while the test set is used to assess quality of the selected gene subsets. The process was repeated 100 times using different training-test splits. Feature selection algorithm evaluation and comparison are based on the average of the 100 repeats. The procedure is illustrated in Fig. 4.3.

As discussed above, evaluating genes in terms of counting-based error estimators has two problems including excessive variability and ties, and both problems have impacts on gene selection. In the experiment, the 100 training-test data splits, the fold partitions in the 10-fold cross-validation and the replicates in B.632+ identical are kept identical across the 10 newly generated datasets. The design could separate the impact of ties from that of variability of error estimation. This is because under such settings, the impact of the variability in error estimation is almost identical across the 10 newly generated datasets, and comparisons among the results on the 10 newly generated datasets would reveal the impact of ties problem. Since the SFS procedure is a deterministic search algorithm, the selection performances for a specific evaluation criterion would be the same on the 10 newly generated datasets if there were no ties. If different results are produced in the 10 newly generated datasets, the difference is caused by the ties problem.

The experiment results of the three benchmark datasets, using the SVM classifier, were shown in Fig. 4.4~4.7. Assume 10 genes were selected. For the leukaemia dataset, the test errors of wrapper methods employing the resubstitution error, leave-one-out error, cross-validation error and bootstrap error as evaluation criteria vary in the range of [6.1% 8.9%], [6.5% 9.3%], [6.1% 8.7%] and [6.1% 7.8%], respectively,

Figure 4.3: Illustration of the test procedure. In the figure, we only plot 6 genes for demonstration purpose. In practice, the number of genes is more than 1000.

as shown in Fig. 4.4.1~4.4.4. For the breast cancer dataset, the test errors vary in the range of [13.2% 19.3%], [14.2% 19.6%], [13.9% 18.9%] and [14.0% 16.3%], respectively, as shown in Fig. 4.5.1~4.5.4. And for the colon dataset, the test errors vary in [20.0% 22.2%], [18.8% 22.3%], [20.2% 21.6%] and [21.3% 21.7%], respectively, as shown in Fig. 4.6.1~4.6.4. Except for the bootstrap estimator on the colon dataset, the variations of these test errors are apparent. Note that the variation is caused only by the different gene order in the data matrix. However, the difference in gene order does not change the intrinsic property of the data at all. This indicates that the counting-based error estimator may not necessarily be a reliable and consistent evaluation criterion. However, when continuous evaluation criterion such as generalized $\|\mathbf{w}\|^2$ measure was used, exactly the same results were obtained from the 10 newly generated datasets, as shown in Fig 4.7.1~4.7.3.

From the experiments, we can see that different gene order in the data matrix produces different result for the same wrapper scheme (that is, the same search procedure, evaluation criterion, stopping criterion and classification algorithm). Actually, the root cause of the problem lies in the error estimator. Due to the discrete error estimators, some gene subsets would have the same value of evaluation criterion. As there are ties during the gene selection process, the selection of gene subset would depend on how the ties are broken. Conventionally, the tie-breaking rule is that the first one would be selected among all the gene subsets with the same value of evaluation criterion. For the SFS algorithm, the gene order indicates a kind of tie-breaking rule, that is, the genes that rank in the front would have greater probability to be selected. For example, on a dataset with 5 genes, gene 1 and gene 2 lead to the same value of evaluation criterion when they are respectively added into the previous selected subsets. If the gene order is “1-2-3-4-5”, then gene 1 would be selected; whereas gene 2 would be selected if the order is “5-4-3-2-1”. When we permuted the order of genes, different tie-breaking rules were employed, which led to different results. In other words, the arbitrary selection we found is just caused

4.4.1: $\hat{\epsilon}_{resub}$ 4.4.2: $\hat{\epsilon}_{loo}$ 4.4.3: $\hat{\epsilon}_{10cv}$ 4.4.4: $\hat{\epsilon}_{B.632+}$

Figure 4.4: The test performance of error rate-based wrapper methods on the leukaemia dataset using SVM as classification algorithm and SFS as search procedure. The four figures show the test performance for the wrapper methods using $\hat{\epsilon}_{resub}$, $\hat{\epsilon}_{loo}$, $\hat{\epsilon}_{10cv}$ and $\hat{\epsilon}_{B.632+}$, respectively, as evaluation criterion. Each figure includes 10 curves. Every curve corresponds to the test performance when carrying out wrapper method on one newly generated dataset.

4.5.1: $\hat{\epsilon}_{resub}$ 4.5.2: $\hat{\epsilon}_{loo}$ 4.5.3: $\hat{\epsilon}_{10cv}$ 4.5.4: $\hat{\epsilon}_{B.632+}$

Figure 4.5: The test performance of error rate-based wrapper methods on the breast cancer dataset using SVM as classification algorithm and SFS as search procedure. The four figures show the test performance for the wrapper methods using $\hat{\epsilon}_{resub}$, $\hat{\epsilon}_{loo}$, $\hat{\epsilon}_{10cv}$ and $\hat{\epsilon}_{B.632+}$, respectively, as evaluation criterion. Each figure includes 10 curves. Every curve corresponds to the test performance when carrying out wrapper method on one newly generated dataset.

4.6.1: $\hat{\epsilon}_{resub}$ 4.6.2: $\hat{\epsilon}_{loo}$ 4.6.3: $\hat{\epsilon}_{10cv}$ 4.6.4: $\hat{\epsilon}_{B.632+}$

Figure 4.6: The test performance of error rate-based wrapper methods on the colon cancer dataset using SVM as classification algorithm and SFS as search procedure. The four figures show the test performance for the wrapper methods using $\hat{\epsilon}_{resub}$, $\hat{\epsilon}_{loo}$, $\hat{\epsilon}_{10cv}$ and $\hat{\epsilon}_{B.632+}$, respectively, as evaluation criterion. Each figure includes 10 curves. Every curve corresponds to the test performance when carrying out wrapper method on one newly generated dataset.

4.7.1: leukaemia dataset

4.7.2: breast cancer dataset

4.7.3: colon cancer dataset

Figure 4.7: Comparison of SVM-based wrapper methods using four error estimators and generalized $\|\mathbf{w}\|^2$ measure, respectively, as evaluation criterion and using SFS as search procedure. The curves for the four error estimators were obtained by averaging the test errors across 10 newly generated datasets.

from the ties of discrete counting-based error estimators.

We averaged the test errors across 10 newly generated datasets to assess the performance of wrapper methods using the four error estimators, respectively, as evaluation criterion. As shown in Fig. 4.7, the gene subset identified by the generalized $\|\mathbf{w}\|^2$ measure outperforms those by the other four error estimators. The same conclusion can be drawn from the results of KNN classifier, as shown in Fig. 4.8~4.11.

Note that we only employed SFS as the search procedure. A shortcoming of SFS is the “nesting effect”, that is, once a feature has been selected to the subset, it cannot be removed at later steps. Thus, the selected feature subset may not achieve the global minimum for the evaluation criterion. However, the ties problem we discussed above is caused by the discrete evaluation criterion rather than the search algorithm. Even other search procedures are used, the problem still exists. To illustrate this, we investigated two other search procedures. The first one is random selection. We randomly generated 1,000,000 gene subsets (each includes 10 different genes) from the leukaemia dataset. By calculating the leave-one-out error, there are 10,111 gene subsets with zero leave-one-out error. Each of the 1000 genes presents at least 10 times within the 10,111 subsets. This means any of the 1000 genes have chance to be selected, regardless of their merits. We also employed sequential forward floating selection (SFFS) described in [88] for gene selection (please refer to Section 3.2.2.4 for details). SFFS allows backtracking as long as this optimizes the evaluation criterion function, which overcome the “nesting effect” in SFS. However, SFFS presents similar performance with SFS for the error counting based wrapper methods. The experimental results are shown in Fig. 4.12~4.19.

4.8.1: $\hat{\epsilon}_{resub}$ 4.8.2: $\hat{\epsilon}_{loo}$ 4.8.3: $\hat{\epsilon}_{10cv}$ 4.8.4: $\hat{\epsilon}_{B.632+}$

Figure 4.8: The test performance of error rate-based wrapper methods on the leukaemia dataset using KNN as classification algorithm and SFS as search procedure. The four figures show the test performance for the wrapper methods using $\hat{\epsilon}_{resub}$, $\hat{\epsilon}_{loo}$, $\hat{\epsilon}_{10cv}$ and $\hat{\epsilon}_{B.632+}$, respectively, as evaluation criterion. Each figure includes 10 curves. Every curve corresponds to the test performance when carrying out wrapper method on one newly generated dataset.

4.9.1: $\hat{\epsilon}_{resub}$ 4.9.2: $\hat{\epsilon}_{loo}$ 4.9.3: $\hat{\epsilon}_{10cv}$ 4.9.4: $\hat{\epsilon}_{B.632+}$

Figure 4.9: The test performance of error rate-based wrapper methods on the breast cancer dataset using KNN as classification algorithm and SFS as search procedure. The four figures show the test performance for the wrapper methods using $\hat{\epsilon}_{resub}$, $\hat{\epsilon}_{loo}$, $\hat{\epsilon}_{10cv}$ and $\hat{\epsilon}_{B.632+}$, respectively, as evaluation criterion. Each figure includes 10 curves. Every curve corresponds to the test performance when carrying out wrapper method on one newly generated dataset.

4.10.1: $\hat{\epsilon}_{resub}$ 4.10.2: $\hat{\epsilon}_{loo}$ 4.10.3: $\hat{\epsilon}_{10cv}$ 4.10.4: $\hat{\epsilon}_{B.632+}$

Figure 4.10: The test performance of error rate-based wrapper methods on the colon cancer dataset using KNN as classification algorithm and SFS as search procedure. The four figures show the test performance for the wrapper methods using $\hat{\epsilon}_{resub}$, $\hat{\epsilon}_{loo}$, $\hat{\epsilon}_{10cv}$ and $\hat{\epsilon}_{B.632+}$, respectively, as evaluation criterion. Each figure includes 10 curves. Every curve corresponds to the test performance when carrying out wrapper method on one newly generated dataset.

4.11.1: leukaemia dataset

4.11.2: breast cancer dataset

4.11.3: colon cancer dataset

Figure 4.11: Comparison of KNN-based wrapper methods using four error estimators and modified Relief's measure, respectively, as evaluation criterion and using SFS as search procedure. The curves for the four error estimators were obtained by averaging the test errors across 10 newly generated datasets.

Figure 4.12: The test performance of error rate-based wrapper methods on the leukaemia dataset using SVM as classification algorithm and SFFS as search procedure. The four figures show the test performance for the wrapper methods using $\hat{\epsilon}_{resub}$, $\hat{\epsilon}_{loo}$, $\hat{\epsilon}_{10cv}$ and $\hat{\epsilon}_{B.632+}$, respectively, as evaluation criterion. Each figure includes 10 curves. Every curve corresponds to the test performance when carrying out wrapper method on one newly generated dataset.

Figure 4.13: The test performance of error rate-based wrapper methods on the breast cancer dataset using SVM as classification algorithm and SFFS as search procedure. The four figures show the test performance for the wrapper methods using $\hat{\epsilon}_{resub}$, $\hat{\epsilon}_{loo}$, $\hat{\epsilon}_{10cv}$ and $\hat{\epsilon}_{B.632+}$, respectively, as evaluation criterion. Each figure includes 10 curves. Every curve corresponds to the test performance when carrying out wrapper method on one newly generated dataset.

4.14.1: $\hat{\epsilon}_{resub}$ 4.14.2: $\hat{\epsilon}_{loo}$ 4.14.3: $\hat{\epsilon}_{10cv}$ 4.14.4: $\hat{\epsilon}_{B.632+}$

Figure 4.14: The test performance of error rate-based wrapper methods on the colon cancer dataset using SVM as classification algorithm and SFFS as search procedure. The four figures show the test performance for the wrapper methods using $\hat{\epsilon}_{resub}$, $\hat{\epsilon}_{loo}$, $\hat{\epsilon}_{10cv}$ and $\hat{\epsilon}_{B.632+}$, respectively, as evaluation criterion. Each figure includes 10 curves. Every curve corresponds to the test performance when carrying out wrapper method on one newly generated dataset.

4.15.1: leukaemia dataset

4.15.2: breast cancer dataset

4.15.3: colon cancer dataset

Figure 4.15: Comparison of SVM-based wrapper methods using four error estimators and generalized $\|w\|^2$ measure, respectively, as evaluation criterion and using SFFS as search procedure. The curves for the four error estimators were obtained by averaging the test errors across 10 newly generated datasets.

4.16.1: $\hat{\epsilon}_{resub}$ 4.16.2: $\hat{\epsilon}_{loo}$ 4.16.3: $\hat{\epsilon}_{10cv}$ 4.16.4: $\hat{\epsilon}_{B.632+}$

Figure 4.16: The test performance of error rate-based wrapper methods on the leukaemia dataset using KNN as classification algorithm and SFFS as search procedure. The four figures show the test performance for the wrapper methods using $\hat{\epsilon}_{resub}$, $\hat{\epsilon}_{loo}$, $\hat{\epsilon}_{10cv}$ and $\hat{\epsilon}_{B.632+}$, respectively, as evaluation criterion. Each figure includes 10 curves. Every curve corresponds to the test performance when carrying out wrapper method on one newly generated dataset.

4.17.1: $\hat{\epsilon}_{resub}$ 4.17.2: $\hat{\epsilon}_{loo}$ 4.17.3: $\hat{\epsilon}_{10cv}$ 4.17.4: $\hat{\epsilon}_{B.632+}$

Figure 4.17: The test performance of error rate-based wrapper methods on the breast cancer dataset using KNN as classification algorithm and SFBS as search procedure. The four figures show the test performance for the wrapper methods using $\hat{\epsilon}_{resub}$, $\hat{\epsilon}_{loo}$, $\hat{\epsilon}_{10cv}$ and $\hat{\epsilon}_{B.632+}$, respectively, as evaluation criterion. Each figure includes 10 curves. Every curve corresponds to the test performance when carrying out wrapper method on one newly generated dataset.

Figure 4.18: The test performance of error rate-based wrapper methods on the colon cancer dataset using KNN as classification algorithm and SFFS as search procedure. The four figures show the test performance for the wrapper methods using $\hat{\epsilon}_{resub}$, $\hat{\epsilon}_{loo}$, $\hat{\epsilon}_{10cv}$ and $\hat{\epsilon}_{B.632+}$, respectively, as evaluation criterion. Each figure includes 10 curves. Every curve corresponds to the test performance when carrying out wrapper method on one newly generated dataset.

4.19.1: leukaemia dataset

4.19.2: breast cancer dataset

4.19.3: colon cancer dataset

Figure 4.19: Comparison of KNN-based wrapper methods using four error estimators and modified Relief's measure, respectively, as evaluation criterion and SFBS as search procedure. The curves for the four error estimators were obtained by averaging the test errors across 10 newly generated datasets.

4.6 Experimental Studies on UCI Data

We also tested the performance of various wrapper methods, which were listed in the Tab. 4.1 (Section 4.4.2), on the sonar dataset. The sonar dataset is from UCI Repository [80], and it is available at <http://www.ics.uci.edu/~mllearn/MLRepository.html>. There are 208 instances with 60 features in the dataset. The dimensionality is much lower than that of microarray data. For the SVM classification, RBF kernel, $k(\mathbf{x}_1, \mathbf{x}_2) = e^{-g\|\mathbf{x}_1 - \mathbf{x}_2\|^2}$, was employed. In the experiments, $g = 0.02$ and $C = 20$. For the KNN classification, the parameter k was set to 3.

We performed the test procedure illustrated in Fig. 4.3 (Section 4.5) to assess the feature selection performance. The results are shown in Fig. 4.20~4.22. Although, for the resubstitution error and leave-one-out error, the test errors for the wrapper methods vary in a range as well, the variance is smaller than that on the microarray datasets. While for the wrapper methods using cross-validation error and bootstrap error as evaluation criteria, the performance is little affected by the ties problem we found before. Moreover, as shown in Fig. 4.22, the wrapper methods using counting-based error estimators as evaluation criteria achieves comparable performance to those using continuous evaluation criteria. For conventional data (low dimensionality and large or medium sample size), the impact of ties problem is not very significant. We believe it is the reason that this problem has not received many attentions in the literature of feature selection. However, for microarray data analysis, the impact of ties is more significant than that on conventional data due to high dimensionality and small sample size of microarray data.

4.20.1: $\hat{\epsilon}_{resub}$ 4.20.2: $\hat{\epsilon}_{loo}$ 4.20.3: $\hat{\epsilon}_{10cv}$ 4.20.4: $\hat{\epsilon}_{B.632+}$

Figure 4.20: The test performance of error rate-based wrapper methods on the sonar dataset using SVM as classification algorithm. The four figures show the test performance for the wrapper methods using $\hat{\epsilon}_{resub}$, $\hat{\epsilon}_{loo}$, $\hat{\epsilon}_{10cv}$ and $\hat{\epsilon}_{B.632+}$, respectively, as evaluation criterion. Each figure includes 10 curves. Every curve corresponds to the test performance when carrying out wrapper method on one newly generated dataset.

4.21.1: $\hat{\epsilon}_{resub}$ 4.21.2: $\hat{\epsilon}_{loo}$ 4.21.3: $\hat{\epsilon}_{10cv}$ 4.21.4: $\hat{\epsilon}_{B.632+}$

Figure 4.21: The test performance of error rate-based wrapper methods on the sonar dataset using KNN as classification algorithm. The four figures show the test performance for the wrapper methods using $\hat{\epsilon}_{resub}$, $\hat{\epsilon}_{loo}$, $\hat{\epsilon}_{10cv}$ and $\hat{\epsilon}_{B.632+}$, respectively, as evaluation criterion. Each figure includes 10 curves. Every curve corresponds to the test performance when carrying out wrapper method on one newly generated dataset.

4.22.1: SVM

4.22.2: KNN

Figure 4.22: Comparison of wrapper methods using four error estimators and continuous measure, respectively, as evaluation criterion on the sonar dataset. The curves for the four error estimators were obtained by averaging the test errors across 10 newly generated datasets.

4.7 Experimental Studies for Bolstered Error Estimator

The four error estimators we investigated are commonly used in the literature. Recently, Braga-Neto *et al.* [20] proposed bolstered error estimation to improve the performance of resubstitution and cross-validation estimators, particularly in small-sized settings. In the general case, the bolstered resubstitution estimation may be computed by Monte-Carlo sampling,

$$\hat{\epsilon}_{bolster}^R = \frac{1}{nM} \sum_{i=1}^n \sum_{j=1}^M I(\mathcal{C}_S(\mathbf{x}_i^j) \neq y_i), \quad (4.28)$$

where $\mathbf{x}_i^j$, $j = 1, \dots, M$, are Monte-Carlo samples drawn from the sample $\mathbf{x}_i$. In some cases, for a linear classification algorithm with a Gaussian bolstering kernel, the bolstered error estimation can be exactly computed. For example, for linear SVM classification, the bolstered resubstitution estimation is given by,

$$\hat{\epsilon}_{bolster}^R = \frac{1}{n} \sum_{i=1}^n \Phi_{\sigma_i}(-y_i W(\mathbf{x}_i)), \quad (4.29)$$

where Φ_{σ_i} is the cumulative distribution function of a zero-mean Gaussian random variable with variance σ_i , σ_i is a parameter that can be determined by the method proposed in [20], and W is given by $W(\mathbf{x}) = (\mathbf{w}^T \mathbf{x} + b) / \|\mathbf{w}\|$.

Note that there are two approaches to computing the bolstered resubstitution error if the linear SVM is employed. From our point of view, the former, Eq. (4.28), is based on the error-counting mechanism, while the latter, Eq. (4.29), is a continuous function.

We repeated the training-test splits as illustrated in Fig. 4.3 to test the performance of bolstered error based wrapper methods on 10 newly generated datasets with dif-

ferent gene order. In the experiment, the two approaches to computing the bolstered error were compared with applications to wrapper-based gene selection. Note that Monte-Carlo bolstered error estimation is affected by the Monte-Carlo sampling, and then produces internal variance. To eliminate the impact of such internal variance during comparison across the 10 newly generated datasets, we employed the following operations. We generated a sequence of random numbers in advance, and kept the set of random numbers unchanged across the simulations on the 10 newly generated datasets. Under such experimental settings, the selection performance on the 10 newly generated datasets would be of no difference if there were no ties. Hence if different performances are obtained, such variance mostly results from the ties problem.

The results are shown in Fig. 4.23~4.25. Similar with other counting-based error estimators, the test performance for wrapper method using Monte-Carlo bolstered error varies in a large range across the 10 newly generated datasets. While the continuous bolstered estimation presents consistent performance, and outperforms the average result of Monte-Carlo bolstered error in gene selection. From this experiment we see that even for the same error estimator, the continuous approach is better, for gene selection on high dimensional and small-sized microarray data, than error-counting approach. This experiment again demonstrates the advantage of well-designed continuous evaluation criteria over discrete criteria.

4.23.1: Monte-Carlo bolstered error

4.23.2: Comparison

Figure 4.23: Performance of wrapper-based gene selection using bolstered error as evaluation criterion and SVM as classification algorithm on the leukaemia dataset. (1) Test performance of Monte-Carlo bolstered error-based wrapper method on the leukaemia dataset. The figure includes 10 curves. Every curve corresponds to the test performance when carrying out wrapper method on one newly generated dataset. (2) Comparison of wrapper methods using Monte-Carlo bolstered error and continuous bolstered error, respectively, as evaluation criterion on the leukaemia dataset.

4.24.1: Monte-Carlo bolstered error

4.24.2: Comparison

Figure 4.24: Performance of wrapper-based gene selection using bolstered error as evaluation criterion and SVM as classification algorithm on the breast cancer dataset. (1) Test performance of Monte-Carlo bolstered error-based wrapper method on the breast cancer dataset. The figure includes 10 curves. Every curve corresponds to the test performance when carrying out wrapper method on one newly generated dataset. (2) Comparison of wrapper methods using Monte-Carlo bolstered error and continuous bolstered error, respectively, as evaluation criterion on the breast cancer dataset.

4.25.1: Monte-Carlo bolstered error

4.25.2: Comparison

Figure 4.25: Performance of wrapper-based gene selection using bolstered error as evaluation criterion and SVM as classification algorithm on the colon cancer dataset. (1) Test performance of Monte-Carlo bolstered error-based wrapper method on the colon cancer dataset. The figure includes 10 curves. Every curve corresponds to the test performance when carrying out wrapper method on one newly generated dataset. (2) Comparison of wrapper methods using Monte-Carlo bolstered error and continuous bolstered error, respectively, as evaluation criterion on the colon cancer dataset.

Chapter 5

LS Bound-based Gene Selection for DNA microarray data

5.1 Introduction

In Chapter 4, we have investigated the impact of ties problem resulting from counting-based error estimators on gene selection. Our analysis shows that the main cause of the ties problem lies in the discrete error estimators which lack sufficient resolution to differentiate gene subsets. The impact of the ties problem is two-fold. Firstly, the occurrence of ties would introduce uncertainty into the selection procedure. Secondly, less significant genes could have more chance to be selected while more significant genes could be discarded. Hence, the ties problem may deteriorate the gene selection performance. Our experiments show that well-designed continuous criterion outperforms the discrete error estimator in gene selection. This inspires us to focus on designing novel gene selection algorithms using continuous evaluation criteria.

In this chapter, a new continuous criterion, called LS Bound measure, is proposed

to address the gene selection problem. The new criterion has the advantages of both filter and wrapper methods. Firstly, the criterion is derived from the leave-one-out cross validation (LOOCV) procedure of Least Squares Support Vector Machines (LS-SVM), and is closely related to an upper bound of LOOCV classification results. Therefore, the criterion provides genes leading to accurate classification. Secondly, the estimation of the upper bound implicitly involves the training of the classifier only once, without the repeated training in the cross validation. As a result, the computational complexity is significantly reduced compared with the classical wrapper method. Our experiments show that the computational complexity of our method is at the level of the filter method.

This chapter is organized as follows. The Least Square SVM is first briefly reviewed, and a new gene selection criterion, LS Bound measure, based on the upper bound for the LOOCV of LS-SVMs is then proposed. The performance of the criterion is finally tested with three benchmark microarray datasets, i.e. colon cancer dataset [9], leukaemia dataset [45], and breast cancer dataset [114].

5.2 Least Squares Support Vector Machines

In the past few years support vector machines (SVMs) [112] have been introduced to solve pattern recognition problems. SVM classification separates one class from the other with a hyperplane that maximizes the distance between the hyperplane and the nearest sample of each class. The determination of the hyperplane involves solving a quadratic programming (QP) problem, which requires expensive computations. To alleviate this problem, Suykens and Vandewalle [101] proposed a least squares version of SVMs for classification problems. Instead of considering inequality constraints in the classical SVM approach, LS-SVM employs equality constraints.

Consider n training data pairs: $\{(\mathbf{x}_i, y_i)\}$, $i = 1, \dots, n$, where $\mathbf{x}_i$ is a p -dimensional

vector representing the i 'th sample, and y_i is the class label of $\mathbf{x}_i$, which is either +1 or -1. The linear decision boundary is described as:

$$\mathbf{w}^T \mathbf{x} + b = 0 \quad (5.1)$$

where $\mathbf{w} = [w_1, w_2, \dots, w_p]^T$, and b is a scalar. For LS-SVM, the optimization problem is formulated as:

$$\begin{aligned} &\text{minimize} \quad \mathcal{P}(\mathbf{w}, b, \mathbf{e}) = \frac{1}{2} \mathbf{w}^T \mathbf{w} + \frac{\gamma}{2} \mathbf{e}^T \mathbf{e}, \\ &\text{subject to} \quad y_i [\mathbf{w}^T \mathbf{x}_i + b] = 1 - e_i, \quad i = 1, \dots, n, \end{aligned} \quad (5.2)$$

where e_i denotes regression error for sample $\mathbf{x}_i$, $\mathbf{e} = [e_1, e_2, \dots, e_n]^T$, and γ is a given positive value assigned to penalize errors. The role of γ , just as that of the C in classical SVMs, is to adjust the compromise between generalization and training accuracy. The solution to the optimization problem is given by the saddle point of the Lagrangian (please refer to Appendix A for more details about Lagrangian Optimization Theory):

$$\mathcal{L}(\mathbf{w}, b, \mathbf{e}, \boldsymbol{\alpha}) = \frac{1}{2} \mathbf{w}^T \mathbf{w} + \frac{\gamma}{2} \mathbf{e}^T \mathbf{e} - \sum_{i=1}^n \alpha_i [y_i (\mathbf{w}^T \mathbf{x}_i + b) - 1 + e_i] \quad (5.3)$$

with Lagrange multipliers α_i .

The conditions for the optimality

$$\begin{aligned} \frac{\partial \mathcal{L}(\mathbf{w}, b, \mathbf{e}, \boldsymbol{\alpha})}{\partial \mathbf{w}} &= 0, \\ \frac{\partial \mathcal{L}(\mathbf{w}, b, \mathbf{e}, \boldsymbol{\alpha})}{\partial b} &= 0, \\ \frac{\partial \mathcal{L}(\mathbf{w}, b, \mathbf{e}, \boldsymbol{\alpha})}{\partial \alpha_i} &= 0, \quad i = 1, \dots, n, \\ \frac{\partial \mathcal{L}(\mathbf{w}, b, \mathbf{e}, \boldsymbol{\alpha})}{\partial e_i} &= 0, \quad i = 1, \dots, n, \end{aligned}$$

give

$$\mathbf{w} = \sum_{i=1}^n \alpha_i y_i \mathbf{x}_i, \quad (5.4)$$

$$\sum_{i=1}^n \alpha_i y_i = 0, \quad (5.5)$$

$$y_i [\mathbf{w}^T \mathbf{x}_i + b] = 1 - e_i, \quad i = 1, \dots, n, \quad (5.6)$$

$$\alpha_i = \gamma e_i, \quad i = 1, \dots, n. \quad (5.7)$$

Eqs. (5.4)~(5.7) can be written together in matrix form as:

$$\begin{bmatrix} 0 & -\mathbf{y}^T \\ \mathbf{y} & \mathbf{\Omega} + \gamma^{-1} \mathbf{I} \end{bmatrix} \begin{bmatrix} b \\ \boldsymbol{\alpha} \end{bmatrix} = \begin{bmatrix} 0 \\ \mathbf{1} \end{bmatrix} \quad (5.8)$$

where $\mathbf{y} = [y_1, \dots, y_n]^T$, $\mathbf{1} = [1, \dots, 1]^T$, $\boldsymbol{\alpha} = [\alpha_1, \dots, \alpha_n]^T$, $\mathbf{I}$ is the identity matrix, and $\mathbf{\Omega} = \{y_i y_j \mathbf{x}_i^T \mathbf{x}_j\}$.

Hence the classifier (5.1) is obtained by solving the linear set of Eqs. (5.8) instead of quadratic programming (QP) problem in SVMs. Once the coefficients $\boldsymbol{\alpha}$, b are determined, we assign the corresponding class label of the test sample $\mathbf{x}$ by the sign of the function $f(\mathbf{x})$:

$$f(\mathbf{x}) = \sum_{i=1}^n \alpha_i y_i \mathbf{x}_i^T \mathbf{x} + b \quad (5.9)$$

5.3 LS Bound Measure for Gene Selection

As we mentioned before, in most pattern classification applications the wrapper method provides feature subset with better predictive accuracy than the filter method. For better generalization, the leave-one-out cross validation error is often employed to guide the selection in the wrapper method [62]. The leave-one-out procedure gives an almost unbiased estimate of the probability of test error [73,120]. However,

it is a burdensome task to obtain the leave-one-out estimate of each training sample for every candidate gene subsets. Several bounds on the expectation of SVMs from the leave-one-out estimator were introduced to reduce the high computational complexity in the leave-one-out procedure [111]. SVMs are well suited to work with high dimensional data, such as microarray data [41]. But the determination of the hyperplane of SVMs involves solving a quadratic programming (QP) problem, which requires expensive computations. Our present work focuses on LS-SVMs [101]. The solution of LS-SVM can be obtained from solving a set of linear equations. At first glance, it seems that one should solve LS-SVM much more rapidly. However, the situation is more complicated because some SVM implementations, such as LIB-SVM [27] and SVM^{light} [56], use a decomposition approach, which is much faster, while LS-SVM does not permit such approach. Nevertheless, LS-SVM is still a viable alternative to the SVM.

The decision function of a linear classifier can be written as:

$$f(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + b \quad (5.10)$$

The classification is based on the sign of the decision function $f(\mathbf{x})$ in Eq. (5.10). In the present work, we were concentrated on the value of $f(\mathbf{x})$ rather than the sign of $f(\mathbf{x})$ to avoid the ties problem we discussed in Chapter 4. The value of $f(\mathbf{x})$ not only indicates the information of the class label of the sample $\mathbf{x}$, but also indicates the distance of sample $\mathbf{x}$ to the decision boundary. In fact, this value will change whenever a feature is added to the feature subset or deleted from the feature subset. Even if different feature subsets lead to the same classification results, the value $f(\mathbf{x})$ also provides a discriminant power to identify which subset is the best.

Next we present a leave-one-out bound of $f(\mathbf{x})$ by training once on the entire training set to avoid the repeated training during the leave-one-out procedure.

Theorem 5.1. *When an LS-SVM classifier is trained on the entire training set, if the corresponding Lagrangian multiplier (α_l^*) of the training sample $\mathbf{x}_l$ is positive, the following inequality holds in the leave-one-out procedure.*

$$-y_l f^l(\mathbf{x}_l) \leq \alpha_l^* [(D_{\min}^l)^2 + 2/\gamma] - 1 \quad (5.11)$$

where f^l is the decision function given by LS-SVM after the sample $\mathbf{x}_l$ has been removed, α_l^* is the corresponding Lagrangian multiplier of $\mathbf{x}_l$ when the LS-SVM classifier is trained on the entire training set, and $D_{\min}^l$ is the distance between $\mathbf{x}_l$ and its nearest neighbor.

The theorem is proved in Section 6.6.

$f^l(\mathbf{x}_l)$ is the validation result for the sample $\mathbf{x}_l$ in the leave-one-out procedure. If $y_l f^l(\mathbf{x}_l)$ is negative the sample $\mathbf{x}_l$ is considered as an leave-one-out error, and if $y_l f^l(\mathbf{x}_l)$ is positive $\mathbf{x}_l$ is correctly classified in the leave-one-out procedure.

In LS-SVM, we have $\alpha_l^* = \gamma e_l^*$ and $y_l f^*(\mathbf{x}_l) = 1 - e_l^*$, (See Eq. (5.7) and Eq. (5.6)). According to the two equations, we group the training data into three categories based on the values of Lagrange multiplier α^* . The first category includes the samples with $\alpha_i^* \leq 0$, which are far away from the decision boundary. They can be correctly classified not only when the decision function is trained on the basis of the entire data set, but also during the leave-one-out cross validation procedure in normal case. The samples with $0 < \alpha_i^* < \gamma$ are in the second category. Although these samples are correctly classified in the training stage, they might be misclassified during the leave-one-out procedure because they are close to the decision boundary. The third category is comprised of samples with $\alpha_i^* \geq \gamma$, which are misclassified not only in the training procedure but also in the leave-one-out procedure. The bound on the right side of inequality (5.11) is especially useful for the training data in the second category. The upper bound is related to both the corresponding training

result and the nearest neighbor. If the bound is negative, the sample must be correctly classified in the leave-one-out procedure. If the bound is positive, although we are uncertain whether it will be misclassified in the leave-one-out procedure, the bound indicates to some extent the probability of misclassification. Therefore, the bound can be used to evaluate the goodness of the feature (gene) set.

Combining the bounds for all training data together, we propose the following measure, called LS Bound measure, as the evaluation criterion for gene selection:

$$M = \sum_{l=1}^n (\alpha_l^* [(D_{min}^l)^2 + 2/\gamma] - 1)_+ \quad (5.12)$$

where $(x)_+ = \max(0, x)$. α^* can be obtained from solving a set of linear equations in LS-SVM. The solution of α also can be simply formulated as:

$$\alpha^* = \mathbf{H}^{-1} \mathbf{1} - (\mathbf{y}^T \mathbf{H}^{-1} \mathbf{1})(\mathbf{H}^{-1} \mathbf{y})(\mathbf{y}^T \mathbf{H}^{-1} \mathbf{y}) \quad (5.13)$$

where $\mathbf{H} = \mathbf{\Omega} + \gamma^{-1} \mathbf{I}$. Note that the matrix $\mathbf{H}$ is positive definite and therefore invertible. The measure M gives a bound on error expectation, and can be considered as an estimate of the generalization performance. The feature (gene) subset which minimizes the measure M is preferred.

The LS Bound measure M in Eq. (5.12) can be combined with any search algorithm, such as the sequential forward selection (SFS), to form a gene selection algorithm. The SFS algorithm is a simple greedy heuristic search algorithm. For better performance, other complex search algorithms, such as sequential floating forward selection (SFFS) [88], can be used but at the cost of increasing the computational complexity.

The pseudo code of the sequential forward gene selection algorithm can be summarized as follows:

The stopping criterion in our gene selection algorithm is whether a predefined num-

Algorithm 5: LS Bound-based sequential forward selection algorithm

Input: n data pairs $\{\langle \mathbf{x}_i, y_i \rangle\}$, m (predefined number of genes to be selected), γ **Output:** indices of m selected genesInitialize S to an empty set;// S is the set of selected genesInitialize C to the full gene set;// C is the set of candidate genes for selection**for** $i \leftarrow 1$ **to** m **do** p = number of genes in set C ; **for** $j \leftarrow 1$ **to** p **do** Take the j 'th gene in set C and temporarily put into set S ; Calculate the criterion M using all genes in set S ; // evaluate the goodness of temporary set S **end** Select the gene with the minimal M ; Remove the selected gene from set C ; Put the selected gene into set S ;**end****return** gene set S

ber of features are selected. In our experiments, we first selected 50 genes from the original gene set. However, the 50 genes are not the final selection result because few of them can get the same or satisfactory performance. In our opinion, the LS Bound measure reflects the generalization performance of selected gene subset. The decrease of the LS Bound measure indicates the improvement of the generalization performance. The decision of the final gene subset can be made based on whether the LS Bound measure approaches the minimum, or whether adding more genes results in insignificant change.

5.4 Efficient Computation of LS Bound Measure

In the gene selection algorithm described above, the LS Bound measure M needs to be computed for every candidate gene set. For a high dimensional dataset, such as microarray data, the repeated computation of M involving the inverse of an n -by- n

matrix $\mathbf{H}$ in Eq. (5.13) is actually a computationally intensive task. However, in the SFS search procedure we can express the inverse by using the Sherman-Morrison-Woodbury formula [44]:

$$(M + BCD^T)^{-1} = M^{-1} - M^{-1}B(C^{-1} + D^T M^{-1}B)^{-1}D^T M^{-1} \quad (5.14)$$

Suppose that we have selected a particular gene set S and the matrix H for gene set S is denoted by $\mathbf{H}_S$. During the SFS search procedure, we have to temporarily put each of the candidate genes into gene set S to test the performance of each candidate gene. When gene q is added into gene set S to form a temporary set $S' = S \cup \{x_q\}$, the matrix H for gene set S' , denoted by $\mathbf{H}_{S'}$, can be formulated as

$$\mathbf{H}_{S'} = \mathbf{H}_S + \mathbf{z}\mathbf{z}^T \quad (5.15)$$

where vector $\mathbf{z} = (y_1 x_{1q}, \dots, y_n x_{nq})^T$. According to the Sherman-Morrison-Woodbury formula, we have:

$$\mathbf{H}_{S'}^{-1} = \mathbf{H}_S^{-1} - \frac{\mathbf{H}_S^{-1} \mathbf{z} \mathbf{z}^T \mathbf{H}_S^{-1}}{1 + \mathbf{z}^T \mathbf{H}_S^{-1} \mathbf{z}} \quad (5.16)$$

So the inverse of $\mathbf{H}_{S'}$ corresponding to a new subset $S' = S \cup \{x_q\}$ can be recursively computed from the inverse of $\mathbf{H}_S$ using simple matrix operations. Thus, in the SFS procedure, the complex inverse operation is no longer needed by using the trick of Eq. (5.16). The computational cost of our measure, as well as our algorithm, is at the level of the filter method. For example, 65 seconds are needed to select 50 genes from 1000 candidate genes using our LS Bound-based gene selection algorithm on the colon cancer dataset in the MATLAB 7.0 environment (2.5GHz P4 CPU with 512MB RAM), while 62 seconds are needed in the Mahalanobis class separability measure-based sequential forward gene selection. The Mahalanobis class separability measure is a well-known feature subset evaluation criterion for the filter method [34].

5.5 Experimental Results

In this section, we report the performance of our proposed LS Bound-based gene selection algorithm on three publicly available microarray datasets: colon cancer dataset [9], breast cancer dataset [114] and leukaemia dataset [45]. The datasets are described in Section 4.4.1. The techniques in Section 4.4.1 were also employed to pre-process the datasets and to pre-select 1000 genes. All the simulations and comparisons in this work are based on the pre-processed and pre-selected data.

There are two goals in this section. One is to identify (select) genes which are significantly more important than others for the classification purpose. Because of the small sample size of microarray datasets, the goal can be fulfilled by selecting gene subset on the basis of the entire dataset to make the selection more reliable.

The other goal is to evaluate the performance of our gene selection criterion and algorithm, and to compare the performance with other gene selection algorithms. To assess the performance of gene selection algorithm, some researchers [45, 47] split the original dataset into two sets: a training set and a test set; and the gene selection procedure was carried out based on the training set while the performance of selected genes was assessed from the test set. Due to the small sample size of microarray dataset, we do not suggest the approach. Some other researchers employed the entire dataset for gene selection, and the performance of selected genes was tested using k -fold cross validation on the same dataset. This kind of cross validation, called internal cross validation, produces biased estimate, as we discussed in Chapter 3. Ambroise and McLachlan [11] suggested techniques of external (10-fold) cross validation and external .632+ bootstrap, in which the gene selection and validation are performed on the different parts of sample set, to obtain an unbiased estimate. Considering the high variance problem of cross validation, especially for the small-sized microarray data [21], we employed the external .632+ bootstrap [11, 37] to

evaluate the performance of our gene selection algorithm. The .632+ bootstrap estimate was introduced in Chapter 4. Here we summarize the procedure of external .632+ bootstrap for evaluating gene selection algorithms, as shown in Algorithm 6.

Algorithm 6: External .632+ bootstrap

Input: Dataset D , B (number of bootstrap sample sets)

Output: error rate

```

Perform gene selection on the dataset  $D$  to identify gene subset  $S$ ;
//  $D(S)$  represents the dataset only using selected genes  $S$ 
Train the classifier on the dataset  $D(S)$ ;
Test the trained classifier on  $D(S)$  to obtain the resubstitution error  $\hat{e}_{resub}$ ;
Generate  $B$  bootstrap sample sets,  $D_i, i = 1, \dots, B$ ;
for  $i \leftarrow 1$  to  $B$  do
    Perform gene selection on the dataset  $D_i$  to identify gene subset  $S_i$ ;
    //  $D_i(S_i)$  represents the corresponding bootstrap sample set only
    using selected genes  $S_i$ 
    Train the classifier on the dataset  $D_i(S_i)$ , and test on  $D(S_i)$ ;
end
Calculate the bootstrap error  $\hat{e}_{B1}$  using Eq. (4.25);
Calculate B.632+ error  $\hat{e}_{B.632+}$  using Eq. (4.26);

```

Because there are no consistent approaches for evaluation of gene selection algorithms in the literature, it is not easy to compare different algorithms only using the published results. Therefore, we employed the same technique, external B.632+, to assess the performance of other gene selection algorithms for comparison purpose in our study (the gene selection algorithms used for comparison are described below). The bootstrap samples were generated by resampling with replacement from the original dataset. In the present study, the balanced bootstrap sample sets with $B = 200$ replicates were employed to reduce the variance. Each sample in the original sample set is made to appear exactly B times in the balanced bootstrap samples. SVM was employed as the classification algorithm in the validation procedure to estimate the error rates of different gene selection algorithms.

In the present study, we compared our LS Bound-based gene selection algorithm with other published gene selection algorithms, including Weighting factor [45] and SVM

Recursive Feature Elimination (RFE) [47]. The Weighting factor ($W = \frac{|\mu_1 - \mu_2|}{\sigma_1 + \sigma_2}$) is a minor variant of Fisher's ratio, and is also commonly used in the literature of microarray data analysis. Benefiting from the good performance of SVMs in high dimensional microarray data, SVM RFE is often considered as one of the best gene selection algorithms in the literature. The Weighting factor is introduced in Chapter 3, and details about SVM RFE are provided in Chapter 7.

The penalty parameter γ is fixed during the gene selection procedure. Appropriate value of γ is determined by the external .632+ bootstrap estimate. We employed the external B.632+ to evaluate the performance of our LS Bound-based gene selection algorithm with a sequence of given values of γ . The optimal value of γ is chosen to be the one with the minimal estimated error. The tuning procedure on the leukaemia dataset is shown in Fig. 5.1. For the three datasets, the value of γ is set to 0.006. And the parameter C in the SVM is set to 0.1 for all experiments.

Figure 5.1: The external .632 bootstrap error rate when conducting gene selection with different γ 's on the leukaemia dataset.

5.5.1 Colon Cancer Dataset

To identify the genes that can distinguish colon cancer from normal tissue, we first identified 50 genes by employing the LS Bound-based gene selection on the entire dataset. According to the LS Bound measure on the selected gene subset, we finally selected the first 30 genes listed in Tab. 5.1, because adding more genes would not result in significant change on the LS Bound measure, as shown in Fig. 5.2.

Figure 5.2: LS Bound measure during the SFS procedure on the colon cancer dataset.

We assessed the performance of our LS Bound-based gene selection algorithm using external .632+ bootstrap technique and compared it with Weighting factor and SVM RFE. As shown in Fig. 5.3, the LS Bound-based gene selection is better than others.

5.5.2 Breast Cancer Dataset

By employing the LS Bound-based gene selection on the entire dataset, we first selected 50 most important genes for the classification between ER+ and ER- tissues. The corresponding LS Bound measure during the selection procedure was shown in Fig. 5.4. After 25 genes were selected, the LS Bound measure is very close to zero,

No.	Gene No.	Access No.	Gene Description
1	493	R87126	Myosin heavy chain, nonmuscle [Gallus gallus]
2	625	X12671	hnRNP A1 protein
3	249	M63391	Desmin gene
4	1042	R36977	P03001 Transcription factor IIIA
5	377	Z50753	GCAP-II/uroguanylin
6	175	T94579	Chitotriosidase
7	765	M76378	Cysteine-rich protein
8	1325	T47377	S-100P protein
9	228	J03040	SPARC
10	1873	L07648	MXI1
11	1221	R62549	Putative serine
12	1567	T61661	Inorganic Pyrophosphatase [Bos taurus]
13	419	R44418	EBNA-2 nuclear protein [Epstein-barr virus]
14	14	H20709	Myosin light chain alkali
15	1060	U09564	Serine kinase
16	788	X02492	Interferon-induced protein
17	187	T51023	Heat shock 90kDa protein 1
18	734	X66839	Carbonic anhydrase IX
19	792	R88740	ATP synthase coupling factor 6
20	353	T57882	Myosin heavy chain, nonmuscle (MYH9)
21	1976	K03474	Mullerian inhibiting substance (MIS, AMH)
22	576	D14812	KIAA0026
23	1623	T94993	Fibroblast growth factor receptor 2
24	1442	M34175	Beta adaptin
25	1372	X75208	HEK2
26	1102	T51558	Procollagen alpha 1(I) chain precursor
27	70	T61661	Profilin I
28	1440	M35878	Insulin-like growth factor-binding protein-3
29	812	Z49269	Chemokine HCC-1
30	365	X14958	High mobility group protein Y

Table 5.1: The genes selected by LS Bound-based selection algorithm for the colon cancer dataset. The genes and ESTs without indicating the source are all from Homo sapiens.

5.3.1: Using SVM as classification algorithm

5.3.2: Using LS-SVM as classification algorithm

Figure 5.3: The external B.632+ error for the colon cancer dataset, shown as the number of selected genes, when SVM and RN were employed as the classification algorithm, respectively, to estimate the error rates. The three curves in each figure are obtained from three gene selection algorithms: Weighting factor, SVM RFE and LS Bound-based gene selection algorithm. The LS Bound-based selection we proposed results in better performance than others.

so we finally selected 25 genes. They were listed in Tab. 5.2.

Figure 5.4: LS Bound measure during the SFS procedure on the breast cancer dataset.

Note that when more than 25 genes were selected, the LS Bound measure was reduced to zero, which is the minimum for LS Bound measure. For well separated dataset, such as breast cancer dataset and leukaemia dataset, it is possible for the LS Bound measure to reach zero. The gene selection may be terminated when LS Bound measure approximates zero. However, in order to compare the performance with other gene selection algorithms, we had to select more genes even if the LS Bound measure reached zero. Under such circumstance, several candidate gene subsets may result in an identical value (that is, zero) on the LS Bound measure. It would make the selection procedure arbitrary just as we discussed in Chapter 4. In this work we employed Fisher’s ratio as the secondary measure to avoid such arbitrary selection. At each step of SFS, every unselected gene is temporarily added into the selected gene subset to build a new subset (that is, each new gene subset links to an unselected gene; in the following part we just use the term “linked gene” to represent the temporarily added gene in a new gene subset). The final selected gene at this step is the linked gene with minimal LS Bound in the new subset. But if several new gene subsets lead to the same LS Bound value, say 0, the LS

Bound measure will lose the selection power on those gene subsets. In such case, Fisher's ratio is employed to identify the best one among the new subsets with zero LS Bound measure. In other words, the final selected gene is the linked gene with maximal Fisher's ratio.

We compared the performance of our LS Bound-based gene selection algorithm with the Weighting factor and SVM RFE. The comparison results are shown in Fig 5.5. Our LS Bound-based gene selection algorithm achieves similar performance with SVM RFE, but it is much better than the Weighting factor.

5.5.3 Leukaemia Dataset

We employed the LS Bound-based gene selection on the entire dataset to select 50 most important genes. The corresponding LS Bound measure during the selection procedure was shown in Fig. 5.6. As the LS Bound measure is reduced to 0 after 20 genes were selected, we finally selected 20 genes listed in Tab. 5.3.

Again, we assessed the performance of our LS Bound-based gene selection algorithm using external .632+ bootstrap technique and compared it with Weighting factor and SVM RFE. As shown in Fig. 5.7, the LS Bound-based gene selection algorithm results in similar performance with SVM RFE, but it outperforms the Weighting factor.

5.6 Proof of Theorem 5.1

Let us first introduce a simple lemma.

Lemma 5.1. *The solution of α for LS-SVM in Eq. (5.8) is to maximize the fol-*

No.	Gene No.	Access No.	Gene Description
1	5524	X03635	Estrogen receptor
2	6484	X87212	Cathepsin C
3	4473	U42408	Ladinin
4	1505	J03778	Microtubule-associated protein
5	6951	Z29083	5T4 oncofoetal antigen
6	5884	X56667	Calretinin
7	2593	M26311	Cystic fibrosis antigen
8	5280	U84011	Glycogen debranching enzyme isoform 6
9	479	D38437	Human DNA mismatch repair
10	878	D87468	KIAA0278
11	2577	M25269	Tyrosine kinase (ELK1) oncogene
12	6476	X86693	Hevin like protein
13	2322	M13241	N-myc gene
14	5908	X57809	Rearranged immunoglobulin lambda light chain
15	3749	U07151	GTP binding protein (ARL3)
16	2364	M14636	Liver glycogen phosphorylase
17	1462	J02871	Lung cytochrome P450 (IV subfamily) BI protein
18	3189	M83186	Cytochrome c oxidase
19	6103	X66945	Fibroblast growth factor receptor
20	2945	M60748	Histone H1 (H1F4)
21	5890	X56807	Desmocollins type 2a and 2b
22	2133	L39061	Transcription factor SL1
23	2911	M59807	NK4
24	3624	U00115	Zinc-finger protein (bcl-6)
25	5238	U82169	Frizzled homolog (FZD3)

Table 5.2: The selected genes by LS Bound-based selection algorithm for the breast cancer dataset.

5.5.1: Using SVM as classification algorithm

5.5.2: Using LS-SVM as classification algorithm

Figure 5.5: The external B.632+ error for the breast cancer dataset, shown as the number of selected genes, when SVM and LS-SVM were employed as the classification algorithm, respectively, to estimate the error rates. The three curves in each figure are obtained from three gene selection algorithms: Weighting factor, SVM RFE and LS Bound-based gene selection algorithm. The LS Bound-based selection we proposed achieves similar performance with SVM RFE, and outperforms the Weighting factor.

No.	Gene No.	Access No.	Gene Description
1	2288	M84526	Adipsin (D component of complement)
2	1882	M27891	Cystatin C
3	1834	M23197	CD33 antigen
4	1144	J05243	Nonerythroid alpha-spectrin
5	2354	M92287	Cyclin D3 (CCND3)
6	6277	M30703	Amphiregulin
7	412	D42043	KIAA0084 gene
8	3549	U65011	Antigen of melanoma
9	3847	U82759	GB DEF
10	1239	L07633	Interferon-gamma IEF SSP 5111
11	4847	X95735	Zyxin
12	4951	Y07604	Nucleoside-diphosphate kinase
13	3900	U88667	ATP binding cassette transporter
14	5657	D63479	Diacylglycerol kinase delta
15	4079	X05409	Aldehyde dehydrogenase 2 (ALDH2)
16	1941	M31994	Aldehyde dehydrogenase 1 (ALDH1)
17	668	D86967	KIAA0212
18	6079	U59632	Cell division control related protein
19	2422	M98045	Folylpolyglutamate synthetase
20	1217	L06132	Voltage-dependent anion channel 1 (VDAC1)

Table 5.3: The selected genes by LS Bound-based selection algorithm for the leukaemia dataset.

Figure 5.6: LS Bound measure during the SFS procedure on the leukaemia dataset.

lowing optimization problem:

$$\begin{aligned}
 & \text{maximize } \mathcal{W}(\boldsymbol{\alpha}) = \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i,j=1}^n \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j - \frac{1}{2} \sum_{i=1}^n \alpha_i^2 / \gamma, \\
 & \text{subject to } \sum_{i=1}^n \alpha_i y_i = 0.
 \end{aligned} \tag{5.17}$$

Proof. According to the optimization theory, the lemma can be easily proved by resubstituting Eqs. (5.4)~(5.7) into function (5.3). The optimization problem (5.17) is just the dual problem for the primal problem (5.2). $\square$

Theorem. When an LS-SVM classifier is trained on the entire training set, if the corresponding Lagrangian multiplier (α_i^*) of the training sample $\mathbf{x}_i$ is positive, the following inequality holds in the leave-one-out procedure.

$$-y_i f^l(\mathbf{x}_i) \leq \alpha_i^* [(D_{min}^l)^2 + 2/\gamma] - 1 \tag{5.18}$$

where f^l is the decision function given by the LS-SVM after the sample $\mathbf{x}_i$ has been removed, α_i^* is the corresponding Lagrangian multiplier of $\mathbf{x}_i$ when the LS-SVM classifier is trained on the entire training set, and D_{min}^l is the distance between $\mathbf{x}_i$ and its nearest neighbor.

5.7.1: Using SVM as classification algorithm

5.7.2: Using LS-SVM as classification algorithm

Figure 5.7: The external B.632+ error for the leukaemia dataset, shown as the number of selected genes, when the SVM and LS-SVM were employed as the classification algorithm, respectively, to estimate the error rates. The three curves in each figure are obtained from three gene selection algorithms: Weighting factor, SVM RFE and LS Bound-based gene selection algorithm. The LS Bound-based gene selection algorithm is a bit inferior to SVM RFE when the SVM classification algorithm is used, but it is a slight better than SVM RFE if LS-SVM is used. However, it outperforms the Weighting factor when either classification algorithm is used.

Proof. For the whole set, the vector $\boldsymbol{\alpha}^* = (\alpha_1^*, \dots, \alpha_n^*)^T$ maximizes the optimization problem (5.17). The decision function is $\mathbf{w}^{*T} \mathbf{x} + b^* = 0$, where $\mathbf{w}^* = \sum_{i=1}^n \alpha_i^* y_i \mathbf{x}_i$.

Let us consider the result of the leave-one-out procedure on the training data $\mathbf{x}_l$. The solution $\boldsymbol{\alpha}^l = (\alpha_1^l, \dots, \alpha_n^l)^T$ for the procedure is just to maximize the optimization problem (5.17) subject to the additional constraint

$$\alpha_l^l = 0. \quad (5.19)$$

Using the solution, we construct the separating hyperplane $\mathbf{w}_l^T \mathbf{x} + b_l = 0$, where $\mathbf{w}_l = \sum_{i=1}^n \alpha_i^l y_i \mathbf{x}_i$.

Since $\boldsymbol{\alpha}^l$ maximizes the optimization problem (5.17) under additional constraint (5.19), the following inequality holds.

$$\mathcal{W}(\boldsymbol{\alpha}^l) \geq \mathcal{W}(\boldsymbol{\alpha}^* - \boldsymbol{\delta}) \quad (5.20)$$

where the vector $\boldsymbol{\delta} = (\delta_1, \dots, \delta_n)$ satisfies the following conditions

$$\delta_l = \alpha_l^*, \quad \sum_{i=1}^n \delta_i y_i = 0 \quad (5.21)$$

Similarly since $\boldsymbol{\alpha}^*$ maximizes the optimization problem (5.17), the following inequality holds.

$$\mathcal{W}(\boldsymbol{\alpha}^*) \geq \mathcal{W}(\boldsymbol{\alpha}^l + \boldsymbol{\beta}) \quad (5.22)$$

where $\boldsymbol{\beta} = (\beta_1, \dots, \beta_n)$ is a vector satisfying the constraint

$$\sum_{i=1}^n \beta_i y_i = 0. \quad (5.23)$$

We obtain

$$\mathcal{W}(\alpha^l + \beta) - \mathcal{W}(\alpha^l) \leq \mathcal{W}(\alpha^*) - \mathcal{W}(\alpha^l) \leq \mathcal{W}(\alpha^*) - \mathcal{W}(\alpha^* - \delta) \quad (5.24)$$

We denote $I_1 = \mathcal{W}(\alpha^l + \beta) - \mathcal{W}(\alpha^l)$ and $I_2 = \mathcal{W}(\alpha^*) - \mathcal{W}(\alpha^* - \delta)$. The maximum of I_1 is equal to the minimum of I_2 , and both of them are $\mathcal{W}(\alpha^*) - \mathcal{W}(\alpha^l)$.

We first calculate the left hand side, I_1 , of inequality (5.24).

$$\begin{aligned} I_1 &= \mathcal{W}(\alpha^l + \beta) - \mathcal{W}(\alpha^l) \\ &= \sum_{i=1}^n \beta_i (1 - \sum_{j=1}^n \alpha_j^l y_i y_j \mathbf{x}_i^T \mathbf{x}_j - \alpha_i^l / \gamma) - \frac{1}{2} \left(\sum_{i,j=1}^n \beta_i \beta_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j + \sum_{i=1}^n \beta_i^2 / \gamma \right) \\ &= \sum_{i=1}^n \beta_i (1 - y_i (\mathbf{w}_l^T \mathbf{x}_i + b_l) - \alpha_i^l / \gamma) - \frac{1}{2} \left(\sum_{i,j=1}^n \beta_i \beta_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j + \sum_{i=1}^n \beta_i^2 / \gamma \right) \\ &= \sum_{i=1, i \neq l}^n \beta_i (1 - y_i (\mathbf{w}_l^T \mathbf{x}_i + b_l) - e_i^l) + \beta_l [1 - y_l (\mathbf{w}_l^T \mathbf{x}_l + b_l)] \\ &\quad - \frac{1}{2} \left(\sum_{i,j=1}^n \beta_i \beta_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j + \sum_{i=1}^n \beta_i^2 / \gamma \right) \\ &= \beta_l [1 - y_l f^l(\mathbf{x}_l)] - \frac{1}{2} \left(\sum_{i,j=1}^n \beta_i \beta_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j + \sum_{i=1}^n \beta_i^2 / \gamma \right) \end{aligned}$$

We define vector β as

$$\beta_l = -y_l y_k \beta_k = a, \quad \beta_i = 0, \quad i \notin \{l, k\}$$

where a is some constant and training data $\mathbf{x}_k$ is the nearest neighbor of $\mathbf{x}_l$. For the specific vector β , we obtain

$$I_1 = a[1 - y_l f^l(\mathbf{x}_l)] - \frac{1}{2} a^2 (||\mathbf{x}_l - \mathbf{x}_k||^2 + 2/\gamma) = a[1 - y_l f^l(\mathbf{x}_l)] - \frac{1}{2} a^2 [(D_{min}^l)^2 + 2/\gamma] \quad (5.25)$$

We choose the value

$$a = \frac{[1 - y_l f^l(\mathbf{x}_l)]}{(D_{min}^l)^2 + 2/\gamma}$$

to maximize the function (5.25). Putting the expression back into (5.25) we obtain

$$I_1 = \frac{[1 - y_l f^l(\mathbf{x}_l)]^2}{2[(D_{min}^l)^2 + 2/\gamma]} \quad (5.26)$$

Now we estimate the right hand side of inequality (5.24).

$$\begin{aligned} I_2 &= \mathcal{W}(\boldsymbol{\alpha}^*) - \mathcal{W}(\boldsymbol{\alpha}^* - \boldsymbol{\delta}) \\ &= \sum_{i=1}^n \delta_i (1 - \sum_{j=1}^n \alpha_j^* y_i y_j \mathbf{x}_i^T \mathbf{x}_j - \alpha_i^* / \gamma) + \frac{1}{2} \left(\sum_{i,j=1}^n \delta_i \delta_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j + \sum_{i=1}^n \delta_i^2 / \gamma \right) \\ &= \sum_{i=1}^n \delta_i (1 - y_i (\mathbf{w}^{*T} \mathbf{x}_i + b^*) - e_i^*) + \frac{1}{2} \left(\sum_{i,j=1}^n \delta_i \delta_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j + \sum_{i=1}^n \delta_i^2 / \gamma \right) \\ &= \frac{1}{2} \left(\sum_{i,j=1}^n \delta_i \delta_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j + \sum_{i=1}^n \delta_i^2 / \gamma \right) \end{aligned}$$

We define vector $\boldsymbol{\delta}$ as

$$\delta_l = -y_k y_l \delta_k = \alpha_l^*, \quad \delta_i = 0, \quad i \notin l, k$$

where the training data $\mathbf{x}_k$ is the nearest neighbor of $\mathbf{x}_l$. For the vector we obtain

$$I_2 = \frac{1}{2} (\alpha_l^*)^2 [(D_{min}^l)^2 + 2/\gamma] \quad (5.27)$$

Combining the Eq. (5.24), (5.26), (5.27) and $\alpha_l^* > 0$, we obtain

$$-y_l f^l(\mathbf{x}_l) \leq \alpha_l^* [(D_{min}^l)^2 + 2/\gamma] - 1 \quad (5.28)$$

□

Chapter 6

Regularization Network-based Gene Selection for Microarray Data Analysis

6.1 Introduction

The problem of approximating a multivariate function from sparse data is ill-posed. A classical remedy is to employ regularization theory [39, 107], which can alleviate the overfitting without performing dimensionality reduction [112]. The general regularization theory leads to the following regularization problem, known as the Tikhonov regularization:

$$\min_{f \in \mathcal{H}} \frac{1}{n} \sum_{i=1}^n V(y_i, f(\mathbf{x}_i)) + \lambda \|f\|_k^2. \quad (6.1)$$

where $V(\cdot, \cdot)$ is a loss function, $\|f\|_k^2$ is a norm in a Reproducing Kernel Hilbert Space $\mathcal{H}$ defined by the positive definite kernel function k , n is the number of training samples (the n pairs $\{\langle \mathbf{x}_i, y_i \rangle\}$), and $\lambda(> 0)$ is a fixed regularization parameter.

Support Vector Machines [112] and Regularization Networks [39] correspond to the solutions of the optimization problem (6.1) with different choices of loss function V . By choosing V to be the hinge loss function, $V(y_i, f(\mathbf{x}_i)) = (1 - y_i f(\mathbf{x}_i))_+$, where $(x)_+ = \max(0, x)$, the framework leads to Support Vector Machines (SVMs) in the case of 1-Norm Soft Margin optimization, while by choosing V as squared loss function, $V(y_i, f(\mathbf{x}_i)) = (y_i - f(\mathbf{x}_i))^2$, it leads to Regularization Networks (RNs). In the present study, Regularization Networks are employed for gene selection and classification of microarray data.

This chapter is organized as follows. The Regularization Network (RN) is first briefly introduced, and a new gene selection algorithm based on the leave-one-out cross validation of Regularization Networks is then proposed. The performance of the RN-based gene selection algorithm is finally tested on three benchmark microarray datasets, i.e. colon cancer dataset [9], breast cancer dataset [114] and leukaemia dataset [45].

6.2 Regularization Networks

Consider n training data pairs: $\{(\mathbf{x}_i, y_i)\}$, $i = 1, \dots, n$, where $\mathbf{x}_i$ is a p -dimensional vector representing the i 'th sample, and $y_i \in \{-1, +1\}$ is the class label of $\mathbf{x}_i$. The framework of Regularization Network can be formulated as a variational problem of finding the function f that minimizes the functional

$$\min_{f \in \mathcal{H}} \frac{1}{n} \sum_{i=1}^n (y_i - f(\mathbf{x}_i))^2 + \lambda \|f\|_k^2, \quad (6.2)$$

where $\|f\|_k^2$ is a norm in a Reproducing Kernel Hilbert Space $\mathcal{H}$ defined by the positive definite kernel function k , and $\lambda (> 0)$ is a fixed regularization parameter. According to the Representer Theorem [95], under rather general conditions the

solution to the above regularization problem has the form

$$f(\mathbf{x}) = \sum_{i=1}^n c_i k(\mathbf{x}, \mathbf{x}_i). \quad (6.3)$$

Substituting Eq. (6.3) into the regularization functional (6.2), we can rewrite the problem as

$$\min_{\mathbf{c} \in \mathbf{R}^n} \frac{1}{n} (\mathbf{y} - K\mathbf{c})^T (\mathbf{y} - K\mathbf{c}) + \lambda \mathbf{c}^T K \mathbf{c}, \quad (6.4)$$

where $\mathbf{c}$ and $\mathbf{y}$ are defined as $\mathbf{c} = [c_1, c_2, \dots, c_n]^T$ and $\mathbf{y} = [y_1, y_2, \dots, y_n]^T$ respectively, and symbol K refers to an n -by- n matrix defined by $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$. Note that the functional (6.4) is differentiable. After taking the derivative with respect to $\mathbf{c}$, the optimal $\mathbf{c}$ can be found by solving the following linear system of equations:

$$(K + \lambda n I) \mathbf{c} = \mathbf{y}, \quad (6.5)$$

where I is the identity matrix. The symmetric matrix $K + \lambda n I$ is strictly positive definite, and thus invertible, so

$$\mathbf{c} = (K + \lambda n I)^{-1} \mathbf{y}. \quad (6.6)$$

The solution f to the regularization problem (6.2) can be expressed as

$$f(\mathbf{x}) = \mathbf{K}^T(\mathbf{x}) (K + \lambda n I)^{-1} \mathbf{y}, \quad (6.7)$$

where $\mathbf{K}(\mathbf{x})$ is the vector of functions satisfying $(\mathbf{K}(\mathbf{x}))_i = k(\mathbf{x}, \mathbf{x}_i)$.

6.3 RN-based Gene Selection

6.3.1 Algorithm

As we mentioned in Chapter 5, it is proved that the leave-one-out procedure gives an almost unbiased estimate of the probability of test error [73, 120]. However, it is burdensome to use the leave-one-out procedure to compute cross validation result for every training sample. Fortunately, we have proved that under the framework of Regularization Networks, the leave-one-out cross validation result of every training sample can be exactly computed without repeated training in the leave-one-out procedure, just as shown in the following theorem.

Theorem 6.1. *For Regularization Networks, the following equation holds in the leave-one-out procedure:*

$$y_l f^l(\mathbf{x}_l) = \frac{y_l (G\mathbf{y})_l - G_{ll}}{1 - G_{ll}}, \quad (6.8)$$

where G is denoted as $G = K[K + \lambda(n-1)I]^{-1}$, G_{ll} is the l 'th item in the l 'th row of matrix G , $(G\mathbf{y})_l$ is l 'th item in the vector $G\mathbf{y}$, and f^l is the decision function given by the Regularization Network after the sample $\mathbf{x}_l$ has been removed.

Proof. The proof is divided into two steps. Firstly, we prove that Eq. (6.8) holds for $l = n$, i.e.

$$y_n f^n(\mathbf{x}_n) = \frac{y_n (G\mathbf{y})_n - G_{nn}}{1 - G_{nn}}. \quad (6.9)$$

According to Eq. (6.7),

$$f^n(\mathbf{x}) = \mathbf{K}_n^T(\mathbf{x})[K_n + \lambda(n-1)I]^{-1}\mathbf{y}^n, \quad (6.10)$$

where $\mathbf{K}_n(\mathbf{x})$ is a vector of functions such that $(\mathbf{K}_n(\mathbf{x}))_i = k(\mathbf{x}, \mathbf{x}_i)$, $i = 1, \dots, n-1$, K_n is an $(n-1)$ by $(n-1)$ matrix with $(K_n)_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$, $i, j = 1, \dots, n-1$, and $\mathbf{y}^n = [y_1, \dots, y_{n-1}]^T$.

Let $H = K + \lambda(n-1)I$, and $H_n = K_n + \lambda(n-1)I$. Eq. (6.10) can be expressed as:

$$f^n(\mathbf{x}_n) = \mathbf{K}_n^T(\mathbf{x}_n)H_n^{-1}\mathbf{y}^n. \quad (6.11)$$

Let $g = k(\mathbf{x}_n, \mathbf{x}_n) + \lambda(n-1)$. The matrix H may be present in the form of partitioned matrix,

$$H = \begin{pmatrix} H_n & \mathbf{K}_n(\mathbf{x}_n) \\ \mathbf{K}_n^T(\mathbf{x}_n) & g \end{pmatrix}.$$

According to the theorem related to the inverse of a partitioned matrix in [49, page 99], the inverse of H can be expressed as follows,

$$H^{-1} = \begin{pmatrix} H_n^{-1} & 0 \\ 0 & 0 \end{pmatrix} + \begin{pmatrix} H_n^{-1}\mathbf{K}_n(\mathbf{x}_n)\mathbf{K}_n^T(\mathbf{x}_n)H_n^{-1} & -H_n^{-1}\mathbf{K}_n(\mathbf{x}_n) \\ -\mathbf{K}_n^T(\mathbf{x}_n)H_n^{-1} & 1 \end{pmatrix} / q,$$

where $q = g - \mathbf{K}_n^T(\mathbf{x}_n)H_n^{-1}\mathbf{K}_n(\mathbf{x}_n)$.

Then we have,

$$\begin{aligned} G &= KH^{-1} \\ &= \begin{pmatrix} K_n & \mathbf{K}_n(\mathbf{x}_n) \\ \mathbf{K}_n^T(\mathbf{x}_n) & k(\mathbf{x}_n, \mathbf{x}_n) \end{pmatrix} H^{-1} \\ &= \begin{pmatrix} K_n H_n^{-1} & 0 \\ \mathbf{K}_n^T(\mathbf{x}_n) H_n^{-1} & 0 \end{pmatrix} + \\ &\quad \begin{pmatrix} (K_n H_n^{-1} - I)\mathbf{K}_n(\mathbf{x}_n)\mathbf{K}_n^T(\mathbf{x}_n)H_n^{-1} & -(K_n H_n^{-1} - I)\mathbf{K}_n(\mathbf{x}_n) \\ [\mathbf{K}_n^T(\mathbf{x}_n)H_n^{-1}\mathbf{K}_n(\mathbf{x}_n) - k(\mathbf{x}_n, \mathbf{x}_n)]\mathbf{K}_n^T(\mathbf{x}_n)H_n^{-1} & -[\mathbf{K}_n^T(\mathbf{x}_n)H_n^{-1}\mathbf{K}_n(\mathbf{x}_n) - k(\mathbf{x}_n, \mathbf{x}_n)] \end{pmatrix} / q \end{aligned}$$

It is easy to get,

$$G_{nn} = -[\mathbf{K}_n^T(\mathbf{x}_n)H_n^{-1}\mathbf{K}_n(\mathbf{x}_n) - k(\mathbf{x}_n, \mathbf{x}_n)]/q \quad (6.12)$$

Note that $G\mathbf{y} = G \begin{pmatrix} \mathbf{y}^n \\ y_n \end{pmatrix}$, and we have,

$$(G\mathbf{y})_n = \mathbf{K}_n^T(\mathbf{x}_n)H_n^{-1}\mathbf{y}^n + [\mathbf{K}_n^T(\mathbf{x}_n)H_n^{-1}\mathbf{K}_n(\mathbf{x}_n) - k(\mathbf{x}_n, \mathbf{x}_n)](\mathbf{K}_n^T(\mathbf{x}_n)H_n^{-1}\mathbf{y}^n - y_n)/q \quad (6.13)$$

Substituting Eqs. (6.11) and (6.12) into Eq. (6.13),

$$(G\mathbf{y})_n = f^n(\mathbf{x}_n) - G_{nn}(f^n(\mathbf{x}_n) - y_n).$$

It follows at once that Eq. (6.9) holds.

In the second step, we prove that Eq. (6.8) holds for $l \neq n$. We construct a new dataset $\{\mathbf{z}_i, y_i^z\}$, where

$$\begin{cases} \mathbf{z}_i = \mathbf{x}_i, & y_i^z = y_i & \text{if } i \neq l, n \\ \mathbf{z}_l = \mathbf{x}_n, & y_l^z = y_n \\ \mathbf{z}_n = \mathbf{x}_l, & y_n^z = y_l \end{cases}.$$

The kernel matrix on the new dataset

$$K^z = JKJ,$$

where J is obtained by exchanging the l 'th and n 'th rows of identity matrix. Noted that $JJ = I$. Similarly, we also have,

$$\mathbf{y}^z = J\mathbf{y},$$

where $\mathbf{y}^z = \{y_1^z, y_2^z, \dots, y_n^z\}^T$.

Easily obtained,

$$\begin{aligned}
 G^z &= K^z[K^z + \lambda(n-1)I]^{-1} \\
 &= JKJ[JKJ + \lambda(n-1)I]^{-1} \\
 &= JKJ[JKJ + J\lambda(n-1)IJ]^{-1} \\
 &= JKJJ[K + \lambda(n-1)I]^{-1}J \\
 &= JK[K + \lambda(n-1)I]^{-1}J \\
 &= JGJ.
 \end{aligned} \tag{6.14}$$

Then

$$G^z \mathbf{y}^z = JGJJ\mathbf{y} = JG\mathbf{y}. \tag{6.15}$$

According to Eq. (6.14),

$$G_{nn}^z = G_{ll}, \tag{6.16}$$

and according to Eq. (6.15),

$$(G^z \mathbf{y}^z)_n = (G\mathbf{y})_l. \tag{6.17}$$

Combining Eqns. (6.9), (6.16) and (6.17),

$$\begin{aligned}
 y_l f^l(\mathbf{x}_l) &= y_n^z f^n(\mathbf{z}_n) \\
 &= \frac{y_n^z (G^z \mathbf{y}^z)_n - G_{nn}^z}{1 - G_{nn}^z} \\
 &= \frac{y_l (G\mathbf{y})_l - G_{ll}}{1 - G_{ll}}.
 \end{aligned}$$

Considering the two steps together, Eq. (6.8) is proved. $\square$

The Eq. (6.8) demonstrates that the computation of cross validation results only demands matrix G , without involving any repeated training of the n classifiers in

the leave-one-out procedure. $f^l(\mathbf{x}_l)$ is just the validation result for the sample $\mathbf{x}_l$ in the leave-one-out procedure. If $y_l f^l(\mathbf{x}_l)$ is negative the sample $\mathbf{x}_l$ is considered as a leave-one-out error, otherwise $\mathbf{x}_l$ is correctly classified in the leave-one-out procedure.

One way of utilizing the cross validation results provided by Eq. (6.8) for feature selection is to compute the leave-one-out error, and to take it as the evaluation criterion. However, as we discussed in Chapter 4, the leave-one-out error is not a desirable gene selection criterion for the high dimensional and small-sized microarray data, because different candidate gene subsets might lead to the same leave-one-out errors, which could result in an arbitrary selection. In this work, we designed a continuous criterion to replace the discrete leave-one-out error.

If $p(y|\mathbf{x})$ is defined as the probability that class label of $\mathbf{x}$ is y , Lin [71] proved that the solution of RN, $f(\mathbf{x})$, is convergent to $2p(y = 1|\mathbf{x}) - 1$ when $\lambda \rightarrow 0$. So in the present study, we employed the following function h to estimate the probability that a specific sample $\mathbf{x}_l$ is correctly classified in the leave-one-out procedure,

$$h(y_l f^l(\mathbf{x}_l)) = \left(\frac{y_l f^l(\mathbf{x}_l) + 1}{2} \right)_{[0,1]}, \quad \text{where } (x)_{[0,1]} = \begin{cases} 1 & \text{if } x > 1 \\ x & \text{if } 1 \geq x \geq 0 \\ 0 & \text{if } x < 0 \end{cases} \quad (6.18)$$

Combining all the estimates of training data together, we proposed the following evaluation criterion for gene selection:

$$M = \sum_{l=1}^n \left(\frac{y_l f^l(\mathbf{x}_l) + 1}{2} \right)_{[0,1]} \quad (6.19)$$

where $y_l f^l(\mathbf{x}_l)$ is computed by using Eq. (6.8). The criterion M gives an estimate on the generalisation performance. The feature (gene) subset which maximizes the criterion M is preferred. In the present study, the criterion M was combined with sequential forward selection (SFS) to form a gene selection algorithm. To get

better performance, other search algorithms, such as sequential floating forward selection (SFFS) [88], can be used but at the cost of increasing the computational complexity. The pseudo code of the sequential forward gene selection algorithm can be summarized in Algorithm 7.

Algorithm 7: Combining RN criterion M with SFS for gene selection

Input: n data pairs $\{\langle \mathbf{x}_i, y_i \rangle\}$, m (predefined number of genes to be selected), λ

Output: indices of m selected genes

Initialize S to an empty set;

// S is the set of selected genes

Initialize C to the full gene set;

// C is the set of candidate genes for selection

for $i \leftarrow 1$ **to** m **do**

p = number of genes in set C ;

for $j \leftarrow 1$ **to** p **do**

 Take the j 'th gene in set C and temporarily put into set S ;

 Calculate the criterion M using all genes in set S ;

 // evaluate the goodness of temporary set S

end

 Select the gene with the maximal M ;

 Remove the selected gene from set C ;

 Put the selected gene into set S ;

end

return gene set S

The gene selection algorithm can be considered as a modification of the leave-one-out error based wrapper method, in which the 0-1 function for computing the classification error rate is replaced by the function h in Eq. (6.18).

6.3.2 Selection of the Kernel

Regularization Networks, just as SVMs, can perform linear or non-linear classification by using the trick of kernel. Which particular kernel should be used depends on the application being investigated. In the machine learning area, selecting the appropriate kernel can be addressed as a model selection problem. From the viewpoint of model selection, the appropriate model should be with “the smallest possible num-

ber of parameters for adequate representation of the data” according to the *principle of parsimony* [19]. Normally a microarray dataset contains a large number of genes (usually more than 1000) and a relatively small number of samples (usually fewer than 100). In such a sparse space a linear classifier is usually adequate for separation between classes. Therefore to avoid overfitting, we prefer the linear classifier, the simplest classification algorithm, for better generalization.

Note that there is no bias term b in the decision function (6.7) when the linear kernel, $k(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{x}_1^T \mathbf{x}_2$, is employed, compared with typical linear classifier, $f(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + b$. The absence of b in linear classification might deteriorate the classification performance. In fact, the issue about using or not using b in Eq. (6.3) is a controversial topic in the Representer Theorem, and this is beyond our research interest. In this work, the polynomial kernel of degree 1, $k'(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{x}_1^T \mathbf{x}_2 + 1$, is used to alleviate the problem. The corresponding Reproducing Kernel Hilbert Space $\mathcal{H}$ (also called the feature space) induced by k' is $(p + 1)$ -dimensional, where p is the dimensionality of the original space. The feature space is generated by just adding a constant unit variable to the original space, so the kernel k' preserves all distance relationships among samples in the feature space. According to Eq. (6.7), the decision function induced by kernel k' is linear as we expect. The solution of RN with the kernel k' is identical with the linear function, $f(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + b$, which minimizes the functional

$$\min_f \frac{1}{n} \sum_{i=1}^n (y_i - f(\mathbf{x}_i))^2 + \lambda(\mathbf{w}^T \mathbf{w} + b^2).$$

In the present study, the polynomial kernel, $k'(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{x}_1^T \mathbf{x}_2 + 1$, is employed both in the classification and in the gene selection.

6.3.3 Efficient Computation of RN-based Criterion

Similar to the LS Bound measure we described in Section 5.4, to compute the RN-based criterion, the inverse of an n -by- n positive definite matrix $K + \lambda(n-1)I$ (for clarity, denote $H = K + \lambda(n-1)I$) in Eq. (6.8) is required to be computed for every gene subset. For a high dimensional dataset, it is a quite intensive task to repeatedly compute the inverse of the corresponding matrix H .

We can employ the same trick as demonstrated in Chapter 5 to solve the problem. Suppose that there are two gene sets S and S' , where $S' = S \cup \{x_q\}$ and x_q represents the q 'th gene. The corresponding matrix H for the two gene sets are denoted as H_S and $H_{S'}$, respective. Then, $H_{S'}$ can be formulated as $H_{S'} = H_S + \mathbf{z}\mathbf{z}^T$, where vector $\mathbf{z} = (x_{1q}, \dots, x_{nq})^T$, if the kernel function $k'(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{x}_1^T \mathbf{x}_2 + 1$ is used. According to the Sherman-Morrison-Woodbury formula, Eq. (5.14), we have:

$$H_{S'}^{-1} = H_S^{-1} - \frac{H_S^{-1} \mathbf{z} \mathbf{z}^T H_S^{-1}}{1 + \mathbf{z}^T H_S^{-1} \mathbf{z}} \quad (6.20)$$

Eq. (6.20) reveals that the inverse of $H_{S'}$ can be recursively computed from inverse of H_S using simple matrix operations. Hence the complexity in computing the RN-based criterion M can be efficiently reduced.

6.4 Experimental Results

In this section, we report the performance of our proposed RN-based gene selection algorithm on three publicly available microarray datasets: colon cancer dataset [9], breast cancer dataset [114] and leukaemia dataset [45]. The datasets are described in Section 4.4.1. The techniques in Section 4.4.1 were also employed to pre-process the datasets and to pre-select 1000 genes. All the simulations and comparisons in this work are based on the pre-processed and pre-selected data.

To identify (select) genes which are significantly more important than others for the classification, RN-based gene selection algorithm was carried out on the entire dataset, which makes the selection more reliable. In order to evaluate the performance of our gene selection algorithm, and to compare with other gene selection algorithms, we employed external .632+ bootstrap [11] (see Algorithm 6 in Chapter 5). The balanced bootstrap samples, with $B = 200$ replicates, were employed in our work. We compared the performance of our RN-based gene selection algorithm with other two gene selection algorithms, including Weighting factor [45], which is commonly used in microarray data analysis, and SVM RFE [47], which is often considered as one of the best gene selection algorithm in the literature.

Our gene selection algorithm is based on RNs, while the RFE is related to SVMs. For fair comparison of the different gene selection algorithms, both RN and SVM were used as the classification algorithm to estimate the error rates. The kernel used in RN was $k(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{x}_1^T \mathbf{x}_2 + 1$, and the linear kernel, $k(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{x}_1^T \mathbf{x}_2$ was employed in SVM.

To select the informative genes by using our RN-based selection algorithm, it is important to determine the appropriate value of the regularization parameter λ . We employed the external .632+ bootstrap to estimate the performance of our RN-based selection algorithm with a sequence of given values of λ . The optimal value of λ was chosen to be the one with the minimal estimated error. The tuning procedure on the leukaemia dataset is shown in Fig. 6.1. For the leukaemia dataset, the value of λ is set to 0.2 for both gene selection and classification, and for the other two datasets, λ is set to 0.5. And the parameter C in the SVM is set to 0.1 for all experiments.

Figure 6.1: The external .632 bootstrap error rate when conducting gene selection with different λ 's on the leukaemia dataset.

6.4.1 Colon Cancer Dataset

We compared the performance of our RN-based gene selection algorithm with those of Weighting factor and SVM RFE. The comparison results on the colon dataset are shown in Fig. 6.2. The difference between the two figures is that Fig. 6.2.1 showed the classification performance when SVM was employed as the classification algorithm, while for Fig. 6.2.2 RN was the classification algorithm. Obviously, our gene selection algorithm outperforms the other two algorithms.

By employing the RN-based gene selection on the entire dataset, we selected 20 most important genes for the classification between normal and tumor tissues. They are listed in Tab 6.1.

6.4.2 Breast Cancer Dataset

We compared the performance of our RN-based gene selection algorithm with those of Weighting factor and SVM RFE. The comparison results on the breast cancer dataset are shown in Fig. 6.3. Our gene selection algorithm shows a better perfor-

6.2.1: Using SVM as classification algorithm

6.2.2: Using RN as classification algorithm

Figure 6.2: The external B.632+ error for the colon cancer dataset, shown as the number of selected genes, when SVM and RN were employed as the classification algorithm, respectively, to estimate the error rates. The three curves in each figure were obtained from three gene selection algorithms: Weighting factor, SVM RFE and RN-based gene selection algorithm. The RN-based selection we proposed results in better performance than others.

No.	Gene No.	Access No.	Gene Description
1	493	R87126	Myosin heavy chain, nonmuscle [Gallus gallus]
2	625	X12671	hnRNP A1 protein
3	377	Z50753	GCAP-II/uroguanylin
4	1567	T61661	Inorganic Pyrophosphatase [Bos taurus]
5	1221	R62549	Putative serine/threonine-protein kinase B0464.5 [Caenorhabditis elegans]
6	391	D31885	KIAA0069
7	14	H20709	Myosin light chain alkali
8	47	T58861	60s ribosomal protein L30E [Kluyveromyces lactis]
9	1976	K03474	Mullerian inhibiting substance (MIS, AMH)
10	799	J03210	Collagenase type IV
11	765	M76378	Cysteine-rich protein
12	1372	X75208	HEK2
13	792	R88740	ATP synthase, H+ transporting, mitochondrial F0 complex, subunit F6 (ATP5J)
14	353	T57882	Myosin heavy chain, nonmuscle (MYH9)
15	652	R10066	Prohibitin
16	788	X02492	Interferon-induced protein
17	1042	R36977	P03001 Transcription factor IIIA
18	427	L10284	Integral membrane protein, calnexin
19	1609	L37792	Syntaxin 1A
20	576	D14812	KIAA0026

Table 6.1: The top 20 genes selected by RN-based selection algorithm for the colon cancer dataset. The genes and ESTs without indicating the source are all from Homo sapiens.

mance than the other two algorithms. By employing the RN-based gene selection on the entire dataset, we selected 20 most important genes for the classification between ER+ and ER- tissues. They are listed in Tab. 6.2.

No.	Gene No.	Access No.	Gene Description
1	5524	X03635	Estrogen receptor
2	6484	X87212	Cathepsin C
3	1512	J03827	Y box binding protein-1
4	5884	X56667	Calretinin
5	1982	L26336	Heat shock protein HSPA2
6	2038	L33799	Procollagen C-proteinase enhancer
7	3558	S79048	Lacrimal gland
8	3749	U07151	GTP binding protein
9	2301	M11722	Terminal transferase
10	2368	M14745	bcl-2
11	2577	M25269	Tyrosine kinase (ELK1) oncogene
12	714	D80009	KIAA0187
13	217	D11327	Protein-tyrosine phosphatase
14	185	D00408	fetal liver cytochrome P-450
15	2322	M13241	N-myc gene
16	6476	X86693	Hevin like protein
17	1376	U10860	Guanosine 5'-monophosphate synthase
18	5280	U84011	Glycogen debranching enzyme isoform 6
19	1653	L02326	Lambda-like gene
20	3710	U04636	Cyclooxygenase-2 (hCox-2)

Table 6.2: The top 20 genes selected by RN-based selection algorithm for the breast cancer dataset.

6.4.3 Leukaemia Dataset

We also compared the performance of three gene selection algorithms on the leukaemia dataset, as shown in Fig. 6.4. Our gene selection algorithm was comparable to SVM RFE when SVM classification algorithm was used, but was a bit inferior to the SVM RFE when RN classification algorithm was used. However, it outperforms Weighting factor when either classification algorithm was used. By employing the RN-based gene selection on the entire dataset, we selected 20 most important genes

6.3.1: Using SVM as classification algorithm

6.3.2: Using RN as classification algorithm

Figure 6.3: The external B.632+ error for the breast cancer dataset, shown as the number of selected genes, when SVM and RN were employed as the classification algorithm, respectively, to estimate the error rates. The three curves in each figure were obtained from three gene selection algorithms: Weighting factor, SVM RFE and RN-based gene selection algorithm. The RN-based selection we proposed results in better performance than others.

for the classification between ALL and AML tissues. They are listed in Tab. 6.3.

No.	Gene No.	Access No.	Gene Description
1	2288	M84526	Adipsin (D component of complement)
2	1882	M27891	Cystatin C
3	2354	M92287	Cyclin D3 (CCND3)
4	4847	X95735	Zyxin
5	1144	J05243	Nonerythroid alpha-spectrin
6	2606	U03057	Actin bundling protein
7	1239	L07633	Interferon-gamma IEF SSP 5111
8	4664	X81479	EMR1 hormone receptor
9	343	D29643	KIAA0115
10	6277	M30703	Amphiregulin
11	3549	U65011	Antigen of melanoma
12	5935	V00594	Metallothionein isoform 2
13	5956	J03077	Co-beta glucosidase
14	2001	M37435	Colony-stimulating factor 1
15	5863	HG3914-HT4184	Cell Division Cycle Protein 2-Related Protein Kinase
16	5122	Z32765	CD36
17	1962	M33680	Cell surface protein TAPA-1
18	6281	M31211	Myosin light chain alkali
19	3169	U40434	Pre-pro-megakaryocyte potentiating factor
20	6923	X58399	L2-9 transcript of unrearranged immunoglobulin V(H)5 pseudogene

Table 6.3: The top 20 genes selected by RN-based selection algorithm for the leukaemia dataset.

6.4.1: Using SVM as classification algorithm

6.4.2: Using RN as classification algorithm

Figure 6.4: The external B.632+ error for the leukaemia dataset, shown as the number of selected genes, when SVM and RN are employed as the classification algorithm, respectively, to estimate the error rates. The three curves in each figure were obtained from three gene selection algorithms: Weighting factor, SVM RFE and RN-based gene selection algorithm. The RN-based gene selection algorithm is comparable to SVM RFE when SVM classification was used, but it is a bit inferior to the SVM RFE when RN was used. However, it outperforms the Weighting factor when either classification algorithm was used.

Chapter 7

Efficient Gene Selection Using Iterative Reduced Forward Search

7.1 Introduction

There are two key components in a feature selection algorithm. One is *evaluation criterion* that is used to measure the goodness of a feature subset, and the other is *search procedure* that generates feature subsets for evaluation. The feature selection can be considered as an optimization problem as follows,

$$A^* = \arg \min_{A \in \mathcal{A}} J(A) \quad (7.1)$$

where $J(\cdot)$ is the function of evaluation criterion, and $\mathcal{A}$ represents the set of all possible feature subsets. The performance of a feature selection algorithm should be evaluated in terms of predictive capacity and computational efficiency. Generally, predictive capacity of features receives more attentions than computational efficiency does. However, in gene selection problem for high dimensional microarray data, the computational efficiency is a must considered factor because of the huge number

of possible gene subsets that have to be evaluated. Computational efficiency is related to not only the evaluation criterion but also the search procedure. In the previous two chapters, we facilitate the gene selection by reducing computations on the evaluation criterion, although the main goal is to improve accuracy. In this chapter, we attempt to reduce computations involved in both evaluation criterion and search procedure. The work is based on the following property of the evaluation criterion, that for two feature subsets, A and A' ,

$$J(A') \geq h(J(A), A, A'). \quad (7.2)$$

The right hand side of inequality (7.2) is called the moving bound of criterion $J(\cdot)$. Note that in Eq. (7.1) the minimum of $J(\cdot)$ is preferred, hence the lower moving bound is required.

The chapter is organized as follows. The (2-Norm Soft Margin) support vector machine is first briefly reviewed in Section 7.2. Then two SVM-based criteria, generalized $\|\mathbf{w}\|^2$ measure and generalized radius/margin bound, are introduced in Section 7.3. Both criteria are applicable to separable and non-separable case. In Section 7.4, the moving bounds for SVM-based criteria are developed. Then Iterative Reduced Forward Search (IRFS) is proposed in Section 7.5. The well-known SVM RFE algorithm is described in Section 7.6. The performance of SVM-IRFS algorithms are tested and discussed in Section 7.7.

7.2 Support Vector Machines

Consider n training data pairs: $\{\langle \mathbf{x}_i, y_i \rangle\}$, $i = 1, \dots, n$, where $\mathbf{x}_i$ is a feature vector representing the i 'th sample, and $y_i \in \{-1, 1\}$ is the class label of $\mathbf{x}_i$.

The support vector machines are introduced in Chapter 4. In this section, we only

consider the 2-Norm Soft Margin SVMs [32]. The primal problem is as follows.

$$\begin{aligned} & \text{minimize} \quad \mathcal{P}(\mathbf{w}, b, \boldsymbol{\xi}) = \frac{1}{2} \|\mathbf{w}\|^2 + \frac{C}{2} \sum_{i=1}^n \xi_i^2, \\ & \text{subject to} \quad y_i [\mathbf{w}^T \mathbf{x}_i + b] \geq 1 - \xi_i, \quad i = 1, \dots, n, \\ & \quad \quad \quad \xi_i \geq 0, i = 1, \dots, n, \end{aligned} \quad (7.3)$$

where the parameter C is a trade-off between accuracy and regularization. The corresponding Lagrangian for the 2-norm soft margin optimization problem (7.3) is

$$\begin{aligned} \mathcal{L}(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}) &= \frac{1}{2} \|\mathbf{w}\|^2 + \frac{C}{2} \sum_{i=1}^n \xi_i^2 \\ &\quad - \sum_{i=1}^n \alpha_i [y_i (\mathbf{w}^T \mathbf{x}_i + b) - 1 + \xi_i], \end{aligned} \quad (7.4)$$

where $\alpha_i \geq 0, i = 1, \dots, n$, are the Lagrange multipliers. The corresponding dual problem is as follows,

$$\begin{aligned} & \text{maximize} \quad \mathcal{W}(\boldsymbol{\alpha}) = \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n y_i y_j \alpha_i \alpha_j (\mathbf{x}_i^T \mathbf{x}_j + \frac{1}{C} \delta_{ij}) \\ & \text{subject to} \quad \sum_{i=1}^n y_i \alpha_i = 0, \\ & \quad \quad \quad \alpha_i \geq 0, \quad i = 1, \dots, n. \end{aligned} \quad (7.5)$$

where δ_{ij} is the Kronecker δ . It is defined to 1 if $i = j$ and 0 otherwise. After finding the Lagrange multipliers α_i^* that maximize the optimization problem (7.5), we obtain the solution for the primal problem (7.3),

$$\begin{aligned} \mathbf{w}^* &= \sum_{i=1}^n y_i \alpha_i^* \mathbf{x}_i, \\ b^* &= y_i (1 - \alpha_i^*/C) - \mathbf{w}^{*T} \mathbf{x}_i, \quad \text{for any } i \text{ with } \alpha_i^* \neq 0. \end{aligned}$$

The dual problem is still a quadratic programming (QP) problem, and can be solved by the same methods as used for separable cases. The only difference is the addition of $\frac{1}{C}$ to the diagonal of the inner product matrix associated with the training set. This has the effect of adding $\frac{1}{C}$ to the eigenvalues of the inner product matrix, which makes the problem better conditioned.

For simplicity, the dual problem can be rewritten in a matrix format as the follows.

$$\begin{aligned} & \text{maximize} \quad \mathcal{W}(\boldsymbol{\alpha}) = \boldsymbol{\alpha}^T \mathbf{1} - \frac{1}{2} \boldsymbol{\alpha}^T \mathbf{H} \boldsymbol{\alpha} \\ & \text{subject to} \quad \boldsymbol{\alpha}^T \mathbf{y} = 0, \\ & \quad \quad \quad \boldsymbol{\alpha} \geqslant \mathbf{0}. \end{aligned} \tag{7.6}$$

where vector $\mathbf{1} = \{1, 1, \dots, 1\}^T$, vector $\mathbf{0} = \{0, 0, \dots, 0\}^T$, $\mathbf{y} = \{y_1, y_2, \dots, y_n\}^T$ and Hessian matrix $\mathbf{H}_{ij} = y_i y_j \mathbf{x}_i^T \mathbf{x}_j + \frac{1}{C} \delta_{ij}$.

7.3 SVM-based Criteria for gene selection

Several criteria derived from support vector machines (SVMs) were investigated for feature selection. For instance, Weston *et. al* [116] proposed a method based on finding the best feature subset which minimizes the radius/margin bound, one of the most common error bounds for SVMs. Rakotomamonjy [90] investigated the performance of several SVM-based criteria, including weight vector norm $\|\mathbf{w}\|^2$ measure, radius/margin bound, and span estimate. Note that these criteria are only applicable to separable cases. However, most of the practical cases are non-separable. Hence, the applications of these criteria are limited. In this section, we introduce two SVM-based criteria, including generalized $\|\mathbf{w}\|^2$ measure and generalized radius/margin bound, which are applicable to both separable and non-separable cases. The former is used in Chapter 4 as well, and we will reveal in Section 7.6

that it is the underlying evaluation criterion in the well-known feature selection algorithm, SVM RFE [47]. The latter is a new error bound that we propose in this work, and it can be considered as an extension of radius/margin bound on general cases.

7.3.1 Generalized $\|\mathbf{w}\|^2$ Measure

Usually, the good generalization performance of SVMs is explained by the existence of a large separating margin. The SVM, for separable cases, minimizes $\frac{1}{2}\|\mathbf{w}\|^2$ to maximize the separating margin between two classes. While for non-separable cases, the 2-norm soft margin SVM is generalized to minimize the functional $\mathcal{P}(\mathbf{w}, b, \boldsymbol{\xi})$ in optimization problem (7.3). Similar to section 4.2.2, the feature selection criterion, $\frac{1}{2}\|\mathbf{w}\|^2$, can also be generalized to $\frac{1}{2}\|\mathbf{w}\|^2 + \frac{C}{2} \sum_{i=1}^n \xi_i^2$. The criterion here is also called generalized $\|\mathbf{w}\|^2$ measure. The difference between two generalized $\|\mathbf{w}\|^2$ measures is that one is for 1-norm soft margin SVM, while the other is for 2-norm soft margin. Denote $\{\mathbf{w}^*, b^*, \boldsymbol{\xi}^*\}$, and $\boldsymbol{\alpha}^*$ as solutions for prime problem (7.3) and dual problem (7.5), respectively. The following equation holds for computing the generalized $\|\mathbf{w}\|^2$ measure according to strong duality theorem (refer to Appendix A).

$$\mathcal{P}(\mathbf{w}^*, b^*, \boldsymbol{\xi}^*) = \mathcal{W}(\boldsymbol{\alpha}^*). \quad (7.7)$$

Therefore the generalized $\|\mathbf{w}\|^2$ measure can be represented as $\mathcal{W}(\boldsymbol{\alpha}^*)$.

7.3.2 Generalized Radius/Margin Bound

The leave-one-out error is known to be an unbiased estimator of the generalization performance of a classifier [73]. In [28, 111, 112], several bounds on the leave-one-out error rate for a SVM classifier were obtained. The most commonly used is the radius/margin bound. Suppose that the size of the maximal margin is ρ , $\rho = 2/\|\mathbf{w}\|$,

and the training data are within a smallest sphere of radius R . If training data of size n are separable, then the leave-one-out error rate L has the bound [112]:

$$L \leq \frac{R^2}{n\rho^2} = \frac{4}{n} R^2 \|\mathbf{w}\|^2. \quad (7.8)$$

The right hand side of inequality (7.8) is the radius/margin bound.

The radius R of the smallest sphere can be computed by solving the following optimization problem,

$$\begin{aligned} & \text{minimize} \quad \mathcal{R}^2 \\ & \text{subject to} \quad \|\mathbf{x}_i - \mathbf{c}\|^2 \leq \mathcal{R}^2, \quad i = 1, \dots, n, \end{aligned} \quad (7.9)$$

where $\mathbf{c}$ is the center of the sphere which contains all samples. To solve the above problem, we can instead maximize the following dual optimization problem:

$$\begin{aligned} & \text{maximize} \quad \mathcal{R}^2(\boldsymbol{\beta}) = \sum_{i=1}^n \beta_i \mathbf{x}_i^T \mathbf{x}_i - \sum_{i,j=1}^n \beta_i \beta_j \mathbf{x}_i^T \mathbf{x}_j \\ & \text{subject to} \quad \sum_{i=1}^n \beta_i = 1 \quad \text{and} \quad \beta_i \geq 0, \quad i = 1, \dots, n. \end{aligned} \quad (7.10)$$

The complexity of the above problem (7.10) is similar with that in (7.5). Both of them are quadratic programming problems. If we denote $\boldsymbol{\beta}^*$ as the solution for the optimization problem (7.10), $\mathcal{R}(\boldsymbol{\beta}^*)$ is the radius of the smallest sphere which contains all the training samples, that is, $R = \mathcal{R}(\boldsymbol{\beta}^*)$, according to the strong duality theorem (refer to Appendix A). For convenience, the above optimization problem can be expressed in the following matrix format.

$$\begin{aligned} & \text{maximize} \quad \mathcal{R}^2(\boldsymbol{\beta}) = \boldsymbol{\beta}^T \mathbf{p} - \boldsymbol{\beta}^T \mathbf{Q} \boldsymbol{\beta} \\ & \text{subject to} \quad \boldsymbol{\beta}^T \mathbf{1} = 1 \quad \text{and} \quad \boldsymbol{\beta} \geq \mathbf{0}, \end{aligned} \quad (7.11)$$

where vector $\mathbf{1} = \{1, 1, \dots, 1\}^T$, vector $\mathbf{p} = \{\mathbf{x}_1^T \mathbf{x}_1, \mathbf{x}_2^T \mathbf{x}_2, \dots, \mathbf{x}_n^T \mathbf{x}_n\}^T$, and Hessian matrix $\mathbf{Q}_{ij} = \mathbf{x}_i^T \mathbf{x}_j$.

Note that the radius/margin bound is only applicable to separable cases. For general cases (either separable or non-separable), we proved the following theorem (proof is provided in Section 7.8):

Theorem 7.1. *For 2-norm soft margin SVM, the leave-one-out error L meets the following inequality,*

$$L \leq \frac{8}{n} \mathcal{W}(\boldsymbol{\alpha}^*) (R^2 + \frac{1}{2C}), \quad (7.12)$$

where n is the size of training samples, R is the radius of the smallest sphere that contains all the training samples, $\mathcal{W}(\cdot)$ is defined in the dual problem (7.5), and $\boldsymbol{\alpha}^*$ is the solution of problem (7.5).

The right hand side of inequality (7.12) is called the generalized radius/margin bound in this work. Let $C = +\infty$ and $\boldsymbol{\xi} = \mathbf{0}$, then the 2-Norm Soft Margin SVM is identical to the SVM in separable cases. Under such conditions, the generalized radius/margin bound is changed to the traditional radius/margin bound.

7.4 Moving Bounds for SVM-based Criteria

For simplicity, we denote $\mathcal{W}(\boldsymbol{\alpha}^*)$ as W^2 , and denote the generalized $\|\mathbf{w}\|^2$ measure and generalized Radius/Margin bound as J_1 and J_2 , respectively. That is,

$$J_1 = W^2, \quad (7.13)$$

$$J_2 = (R^2 + \frac{1}{2C})W^2. \quad (7.14)$$

In Eq. (7.14), we omit the coefficient $\frac{8}{n}$. If any one of the two criteria is used in feature selection, the feature subset with the minimal criterion is preferred. Note that

the computation of any criterion involves solving one or two quadratic programming (QP) problems. The computational cost of feature selection in high dimensional feature space would be very high, if the SVM-based criterion has to be calculated for all candidate feature subsets.

For a feature subset A , assume that we have known the evaluation criterion of this subset, $J(A)$. If the criterion for a candidate feature subset A' can be expressed as $J(A') = h(J(A), A, A')$, and at the same time if the computation of h is much easier than that of J , such equality relationship could be quite helpful to reduce the computational cost in the process of feature selection. The recursive calculation technique, which is often used in the feature selection algorithms, just works in the same way. In general, the search algorithms, such as SFS and SBS, construct the feature subsets at the i 'th step of the algorithm by the addition or subtraction of a feature from the optimal subset at the $(i - 1)$ 'th step. If the value of the criterion at step i can be exactly evaluated by updating its value that is already calculated at step $i - 1$ instead of computing the criterion from its definition, it can result in substantial reduction of computations.

The LS Bound measure and RN-based measure, which were proposed in the previous chapters, employ the similar trick to reduce the computational cost. The bottleneck for calculating both of the measures is the inverse of a positive definite matrix. During the feature selection procedure, the matrix inverse on the candidate feature subset can be expressed as a function that involves the inverse already calculated in previous step. Such technique is very useful, especially for the applications to high dimensional gene expression data, in saving computational costs.

However, for some complex evaluation criteria, such as the criteria derived from SVMs in this work, it is not easy to find such an equality relationship. From the experiments, we found that appropriate inequality relationship between two subsets, for example $J(A') \geq h(J(A), A, A')$, is quite helpful to speed up the feature

selection process as well. Under such circumstance, the function $h(J(A), A, A')$ can be considered as a sort of bound for $J(A')$. However, the bound is slightly different from traditional ones which are determinate for a specific subset A' , as the bound here is variant when different reference subset A is employed. This kind of bound is called moving bound in this work.

For a feature subset A , n training sample pairs are denoted as $\{\langle \mathbf{z}_i, y_i \rangle\}$, $i = 1, \dots, n$, where $\mathbf{z}_i$ is the feature vector representing the i 'th sample, and y_i is the class label of $\mathbf{z}_i$. Accordingly, for a different feature subset A' , the training sample pairs are $\{\mathbf{z}'_i, y_i\}$, $i = 1, \dots, n$. Assume that we have known the values of $J_1(A)$ and $J_2(A)$. The moving bounds for $J_1(A')$ and $J_2(A')$ are derived in the follows.

7.4.1 Moving Bound for Generalized $\|\mathbf{w}\|^2$ Measure

According to the dual problem (7.6), we denote

$$\boldsymbol{\alpha}_A = \arg \max_{\{\boldsymbol{\alpha} | \boldsymbol{\alpha}^T \mathbf{y} = 0, \boldsymbol{\alpha} \geq 0\}} \boldsymbol{\alpha}^T \mathbf{1} - \frac{1}{2} \boldsymbol{\alpha}^T \mathbf{H}^A \boldsymbol{\alpha}, \quad (7.15)$$

where $\mathbf{H}^A$ is the corresponding hessian matrix for feature subset A , that is, $\mathbf{H}^A_{ij} = y_i y_j \mathbf{z}_i^T \mathbf{z}_j + \frac{1}{C} \delta_{ij}$. Similarly, for feature subset A' , denote

$$\boldsymbol{\alpha}_{A'} = \arg \max_{\{\boldsymbol{\alpha} | \boldsymbol{\alpha}^T \mathbf{y} = 0, \boldsymbol{\alpha} \geq 0\}} \boldsymbol{\alpha}^T \mathbf{1} - \frac{1}{2} \boldsymbol{\alpha}^T \mathbf{H}^{A'} \boldsymbol{\alpha}, \quad (7.16)$$

where $\mathbf{H}^{A'}_{ij} = y_i y_j \mathbf{z}'_i{}^T \mathbf{z}'_j + \frac{1}{C} \delta_{ij}$.

Considering that $\alpha_{A'}$ is the solution for the optimization problem (7.16), we have,

$$\begin{aligned} J_1(A') &= \alpha_{A'}^T \mathbf{1} - \frac{1}{2} \alpha_{A'}^T \mathbf{H}^{A'} \alpha_{A'} \\ &\geq \alpha_A^T \mathbf{1} - \frac{1}{2} \alpha_A^T \mathbf{H}^{A'} \alpha_A \\ &= \alpha_A^T \mathbf{1} - \frac{1}{2} \alpha_A^T \mathbf{H}^A \alpha_A - \frac{1}{2} \alpha_A^T (\mathbf{H}^{A'} - \mathbf{H}^A) \alpha_A. \end{aligned} \quad (7.17)$$

Then we can obtain,

$$J_1(A') \geq J_1(A) - \frac{1}{2} \alpha_A^T (\mathbf{H}^{A'} - \mathbf{H}^A) \alpha_A. \quad (7.18)$$

The right hand side of inequality (7.18) is the moving bound for $J_1(A')$. Compared with the optimization problem (7.6) for computing $J_1(A')$, the computational cost on the moving bound is much lower if $J_1(A)$ and α_A are known beforehand.

7.4.2 Moving Bound for Generalized Radius/Margin Bound

The computation of the generalized Radius/Margin bound involves two quadratic programming problems, (7.6) and (7.11). We have obtained the moving bound related to problem (7.6). Similarly, if β_A is denoted as the solution of the optimization problem (7.11) on the reference feature subset A , the moving bound for the radius of the smallest sphere is shown as the follows,

$$R^2(A') \geq R^2(A) - \beta_A^T (\mathbf{p}^A - \mathbf{p}^{A'}) + \beta_A^T (\mathbf{Q}^A - \mathbf{Q}^{A'}) \beta_A, \quad (7.19)$$

where vector $\mathbf{p}^A = \{\mathbf{z}_1^T \mathbf{z}_1, \mathbf{z}_2^T \mathbf{z}_2, \dots, \mathbf{z}_n^T \mathbf{z}_n\}^T$, vector $\mathbf{p}^{A'} = \{\mathbf{z}_1'^T \mathbf{z}_1', \mathbf{z}_2'^T \mathbf{z}_2', \dots, \mathbf{z}_n'^T \mathbf{z}_n'\}^T$, hessian matrix $\mathbf{Q}_{ij}^A = \mathbf{z}_i^T \mathbf{z}_j$, and hessian matrix $\mathbf{Q}_{ij}^{A'} = \mathbf{z}_i'^T \mathbf{z}_j'$.

Considering that the generalized $\|\mathbf{w}\|^2$ measure and radius of the smallest sphere are non-negative according to their definitions, we rewrite the moving bounds, (7.18)

and (7.19), as follows:

$$W^2(A') \geq \left(W^2(A) - \frac{1}{2} \alpha_A^T (\mathbf{H}^{A'} - \mathbf{H}^A) \alpha_A \right)_+, \quad (7.20)$$

$$R^2(A') \geq \left(R^2(A) - \beta_A^T (\mathbf{p}^A - \mathbf{p}^{A'}) + \beta_A^T (\mathbf{Q}^A - \mathbf{Q}^{A'}) \beta_A \right)_+, \quad (7.21)$$

where $(x)_+ = \max(x, 0)$.

By substituting the inequalities (7.20) and (7.21) into Eq. (7.14), we obtain

$$\begin{aligned} J_2(A') \geq & \left(W^2(A) - \frac{1}{2} \alpha_A^T (\mathbf{H}^{A'} - \mathbf{H}^A) \alpha_A \right)_+ \\ & \times \left[\left(R^2(A) - \beta_A^T (\mathbf{p}^A - \mathbf{p}^{A'}) + \beta_A^T (\mathbf{Q}^A - \mathbf{Q}^{A'}) \beta_A \right)_+ + \frac{1}{2C} \right] \end{aligned} \quad (7.22)$$

The right hand side of the inequality (7.22) is the moving bound of $J_2(A')$. As $W^2(A)$, α_A , $R^2(A)$ and β_A are already known after the generalized radius/margin bound on the feature subset A is computed, the moving bound of J_2 on any feature subset can be easily computed, according to inequality (7.22).

7.5 Iterative Reduced Search Algorithms

Search procedure is an important component in feature selection method. As the running speed for feature selection largely depends on the search procedure, well-designed search algorithm is critical for successful applications to high dimensional data, such as microarray data. In this section, we first propose the Iterative Reduced Search (IRS) algorithm, then the Iterative Reduced Forward Search (IRFS) is developed by combining the IRS with SFS to address the feature selection problem.

7.5.1 Iterative Reduced Search (IRS)

Suppose there are m candidate feature subsets, $A_1, A_2, \dots, A_m$. To select the feature subset with minimal criterion J , we have to compute the evaluation criterion for every candidate subset. If m is very large, the computational cost can be very high. Normally, among the m subsets, many are irrelevant to the discriminant problem under investigation. If we could identify the irrelevant feature subsets before exactly computing their evaluation criteria, the computational cost might be greatly saved. This is the main idea of our Iterative Reduced Search (IRS) related algorithm. The moving bound we discussed in the last section is employed to identify the irrelevant candidate subsets, and to speed up the search procedure.

Denote the moving bound for criterion J as functional h , that is, $J(A') \geq h(J(A), A, A')$. In the IRS procedure, we first compute the evaluation criterion for one selected feature subset, then estimate the moving bound for every candidate subset with respect to the selected one. The feature subsets whose moving bounds are greater than the minimal criterion J^* that we have already computed are discarded, because their criteria are not necessary to be exactly evaluated. The procedure is repeated until all the candidate feature subsets are either evaluated or discarded, and the optimal subset is chosen from the evaluated ones. The IRS algorithm is described as follows,

Algorithm 8: Iterative Reduced Search algorithm

Input: n data pairs $\{\langle \mathbf{x}_i, y_i \rangle\}$, m feature subsets A_i **Output:** optimal feature subset from m candidatesInitialize $\mathcal{A}$ to the set of m candidate feature subsets;// $\mathcal{A} = \{A_1, A_2, \dots, A_m\}$ Initialize $J^* \leftarrow +\infty$;// J^* is the minimal criterion value from evaluated subsets $t \leftarrow 1$; // t is the index of current iteration in IRSRandom select one subset from $\mathcal{A}$, and denote as $A^{(1)}$; $h_i^{(0)} \leftarrow -\infty, i = 1, \dots, m$;// $h_i^{(t)}$ is the moving bound for the i 'th element from $\mathcal{A}$ in t 'th iteration**while** $m \neq 0$ **do** // m is the number of elements in Set $\mathcal{A}$ Calculate the criterion $J^{(t)}$ for the subset $A^{(t)}$;#1 $J^* \leftarrow \min(J^*, J^{(t)})$; Evaluate h_i for every subset in $\mathcal{A}$ based on $J^{(t)}$, and $A^{(t)}$;#2 $h_i^{(t)} \leftarrow \max(h_i^{(t-1)}, h_i)$;#3 Remove the subsets whose $h_i^{(t)} \geq J^*$ from $\mathcal{A}$; $m =$ number of subsets in $\mathcal{A}$; Rearrange the $h_i^{(t)}, i = 1, \dots, m$ to correspond to the subsets in $\mathcal{A}$; Select the subset with minimal $h_i^{(t)}$ as $A^{(t+1)}$; $t \leftarrow t + 1$;**end****return** the feature subset with the minimal criterion J^*

The word “reduced” here has two meanings. Firstly, the identified minimal criterion J^* is gradually reduced, according to the line #1 of Algorithm 8, during the iterations, and finally reach the global minimum. Secondly, the number of candidate subsets is reduced after many disqualified subsets are discarded in each iteration. The reduction in the search space is significant for saving computational cost. The two aspects are linked together in the line #3 of Algorithm 8, which describes that only the subsets with $h_i^{(t)} < J^*$ are remained for evaluation. Note that according to line #1 and #2, J^* keeps decreasing, while $h_i^{(t)}$'s keep increasing during iterations. Therefore the inequality $h_i^{(t)} < J^*$ becomes more strict, which ensures more subsets to be discarded.

At the beginning of the IRS algorithm, we randomly select a subset to start the iterative search. Although different initial subsets might make the algorithm to follow different paths to the optimal subset, this does not affect the identification of the optimal subset by the algorithm. However, an appropriate initial subset might lead to the optimal subset more quickly.

According to the line #3, a subset with $h_i^{(t)} < J^*$ cannot be discarded in the current iteration. However, if the moving bound $h_i^{(t)}$ is very close to J^* , the evaluation criterion for the corresponding subset is more likely to exceed J^* . Under such circumstance, the subset will be eventually discarded in the following iterations. Therefore if such subsets are eliminated in the preceding iterations, some unnecessary computational cost can be saved. For this purpose, we introduce a predefined tolerance, $\epsilon \geq 0$, in Algorithm 9.

In the line #1 of Algorithm 9, the subsets with $h_i^{(t)} \geq \frac{J^*}{1+\epsilon}$ are discarded in each iteration. Although it is possible to remove the optimal subset with criterion J_{opt} by mistake (in the case of $J^* > J_{opt} \geq \frac{J^*}{1+\epsilon}$), the minimal criterion J^* obtained by Algorithm 9 satisfies $0 \leq \frac{J^* - J_{opt}}{J_{opt}} \leq \epsilon$. If ϵ is small enough, its impact on the search performance is trivial. Our experimental studies show that, for the case of

Algorithm 9: Iterative Reduced Search algorithm with tolerance ϵ

Input: n data pairs $\{\langle \mathbf{x}_i, y_i \rangle\}$, m feature subsets A_i , tolerance ϵ

Output: optimal feature subset from m candidates

Initialize $\mathcal{A}$ to the set of m candidate feature subsets;

// $\mathcal{A} = \{A_1, A_2, \dots, A_m\}$

Initialize $J^* \leftarrow +\infty$;

// J^* is the minimal criterion value from evaluated subsets

$t \leftarrow 1$; // t is the index of current iteration in IRS

Random select one subset from $\mathcal{A}$, and denote as $A^{(1)}$;

$h_i^{(0)} \leftarrow -\infty, i = 1, \dots, m$;

// $h_i^{(t)}$ is the moving bound for the i 'th element from $\mathcal{A}$ in t 'th iteration

while $m \neq 0$ **do**

 // m is the number of elements in Set $\mathcal{A}$

 Calculate the criterion $J^{(t)}$ for the subset $A^{(t)}$;

$J^* \leftarrow \min(J^*, J^{(t)})$;

 Evaluate h_i for every subset in $\mathcal{A}$ based on $J^{(t)}$, and $A^{(t)}$;

$h_i^{(t)} \leftarrow \max(h_i^{(t-1)}, h_i)$;

#1 Remove the subsets whose $h_i^{(t)} \geq \frac{J^*}{1 + \epsilon}$ from $\mathcal{A}$;

$m =$ number of subsets in $\mathcal{A}$;

 Rearrange the $h_i^{(t)}, i = 1, \dots, m$ to correspond to the subsets in $\mathcal{A}$;

 Select the subset with minimal $h_i^{(t)}$ as $A^{(t+1)}$;

$t \leftarrow t + 1$;

end

return the feature subset with the minimal criterion J^*

$\epsilon = 1\%$, the classification performance is rarely degraded, while the computational cost is dramatically reduced, compared with non-tolerance case, *i.e.* $\epsilon = 0$.

7.5.2 Iterative Reduced Forward Search (IRFS)

The IRS algorithm can be performed in the manner of exhaustive search if the number of all possible candidate subsets is small or moderate. However, in high dimensional data, such as microarray data, exhaustive search is impractical. Generally, suboptimal search algorithms, such as Sequential Forward Selection (SFS) and Sequential Backward Selection (SBS), are employed to make the search procedure computationally feasible. In this work, our IRS algorithm is combined with Sequential Forward Selection (SFS) to form the Iterative Reduced Forward Search (IRFS), which is described in Algorithm 10.

Algorithm 10: Iterative Reduced Forward Search algorithm

Input: n data pairs $\{\langle \mathbf{x}_i, y_i \rangle\}$, tolerance ϵ , m (predefined number of genes to be selected)

Output: the index of m selected features

Initial S to an empty set;

// S is the set of selected features

Initial C to the full feature set;

// C is the set of candidate features for selecting

// C_i is the i 'th element in set C

for $i \leftarrow 1$ **to** m **do**

p = number of features in set C ;

for $j \leftarrow 1$ **to** p **do**

$A_j = S \cup C_j$;

end

 Select the optimal subset from $\{A_j\}, (j = 1, \dots, p)$, by IRS;

 Store the optimal subset in S ;

$C \leftarrow C - S$;

end

return feature set S

SFS is a bottom-up search procedure that adds new features to a feature set one at a step until the final feature set is reached. At the i 'th step of SFS procedure,

the selected subset S containing $i - 1$ features is determined at the previous steps, and the i 'th feature will be selected from all unselected ones. Each of unselected feature is temporarily moved into the selected subset S to form a candidate feature subset. Then the optimal subset for this step is selected from all candidate subsets. In the Algorithm 10, the IRS search is employed to expedite the selection procedure at each step of SFS algorithm.

As we described before that an appropriate initial subset helps to quicken the IRS procedure. In the IRFS algorithm, the following technique, instead of random selection, is employed to identify the appropriate initial subset. At the first step of IRFS procedure ($i = 1$ in the Algorithm 10), each candidate subset contains only one feature. The initial subset is selected as the one containing the feature with the greatest Fisher's ratio. At the following steps ($i > 1$ in the Algorithm 10), suppose that the selected subset at the previous step is denoted as S , and all candidate subsets are $A_1, A_2, \dots, A_p$. The moving bound for every candidate subsets, h_i , is computed based on the evaluation criterion $J(S)$, that is, $h_i = h(J(S), S, A_i)$. The initial subset to start the IRS search at this step is the one with the smallest moving bound, h_i .

The evaluation criterion and search procedure are two main components of a feature selection algorithm. The evaluation criterion in the IRFS algorithm can be generalized $\|\mathbf{w}\|^2$ measure or generalized Radius/Margin bound. For convenience, the selection algorithm using the former is called W^2 -IRFS, while the latter is called R^2W^2 -IRFS.

7.6 SVM Recursive Feature Elimination

The well-known SVM Recursive Feature Elimination (RFE) [47] method performs feature selection in a sequential backward elimination manner, which starts with

all the features and discards one feature each time. In the end, a small subset of features is obtained. At each step, the coefficients of the weight vector $\mathbf{w}$ of a linear SVM are used as the ranking criterion, and the feature with the smallest magnitude of the weight coefficient is removed. The RFE algorithm is described as follows:

Algorithm 11: Support Vector Machine Recursive Feature Elimination

Input: n data pairs $\{(\mathbf{x}_i, y_i)\}$, m (predefined number of features to be selected)

Output: the index of m selected features

Initialize S to the full feature set;

// S is the set of selected features

p = number of features in Set S ;

while $p \neq m$ **do**

 Train a linear SVM with the features in Set S ;

 // Weight vector $\mathbf{w} = [w_1, w_2, \dots, w_p]^T$

 Calculate the ranking criteria for features in Set S : $c_i = w_i^2$;

 Identify the feature with the smallest ranking criterion;

 Remove the identified feature from set S ;

p = number of features in set S ;

end

return feature set S

To reduce the computational cost, the algorithm can be generalized to discard more than one feature at each step, although discarding several features may degrade the selection performance.

As the authors of [47] assumed that the microarray data are linearly separable down to just a few features, they employed Optimum Brain Damage (OBD) algorithm [66] to minimize the evaluation criterion $J = \frac{1}{2} \|\mathbf{w}\|^2$. The magnitude of w_i^2 just corresponds to the approximate change in this criterion, when the i 'th feature is removed.

However, even for separable cases, it is preferable to take the soft margin into account to get better generalization. Considering the general case, the Lagrangian

in Eq. (7.4) can be expanded in Taylor series to second order,

$$\Delta \mathcal{L}(i) = \frac{\partial \mathcal{L}}{\partial w_i} \Delta w_i + \frac{1}{2} \frac{\partial^2 \mathcal{L}}{\partial^2 w_i} (\Delta w_i)^2 + O((\Delta w_i)^3) \quad (7.23)$$

According to the KKT conditions, the optimum of $\mathcal{L}$ is equal to the generalized $\|\mathbf{w}\|^2$ measure, J_1 . And at the optimum of $\mathcal{L}$, the first order term can be neglected, then Eq. (7.23) becomes,

$$\Delta J_1(i) \approx \frac{1}{2} (\Delta w_i)^2. \quad (7.24)$$

If we denote the value of J_1 after the i 'th feature is removed (by setting the corresponding weight to 0) as $J_1(i)$, we obtained,

$$J_1(i) \approx J_1 + \frac{1}{2} w_i^2. \quad (7.25)$$

Therefore removing the feature with smallest w_i^2 will cause the least increase of J_1 .

Rakotomamonjy [90] explained the reason of using w_i^2 as a ranking criterion from the sensitivity analysis of the evaluation criterion $J = \frac{1}{2} \|w\|^2$ with respect to a virtual variable. Here we employ the generalized $\|\mathbf{w}\|^2$ measure, J_1 , for general case, to replace the $\frac{1}{2} \|w\|^2$. To compute the gradient, a virtual scaling factor $\mathbf{v} = [v_1, v_2, \dots, v_p]^T$ is introduced to present the feature vector of samples. After scaled, the feature vector of sample $\mathbf{x} = [x_1, x_2, \dots, x_p]^T$ is changed to $\mathbf{v} \cdot \mathbf{x} = [v_1 x_1, v_2 x_2, \dots, v_p x_p]^T$, where $\cdot$ is the array multiplication. The generalized $\|\mathbf{w}\|^2$ measure on the scaled data can be expressed as,

$$J_1 = \sum_{i=1}^n \alpha_i^* - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n y_i y_j \alpha_i^* \alpha_j^* \left[(\mathbf{v} \cdot \mathbf{x}_i)^T (\mathbf{v} \cdot \mathbf{x}_j) + \frac{1}{C} \delta_{ij} \right].$$

According to the lemma in [28], the gradient of J_1 with respect to scaling factor v_k

can be computed as

$$\begin{aligned}
 \left. \frac{\partial J_1}{\partial v_k} \right|_{v_k=1} &= -\frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n y_i y_j \alpha_i^* \alpha_j^* \frac{\partial \left[(\mathbf{v} \cdot \mathbf{x}_i)^T (\mathbf{v} \cdot \mathbf{x}_j) \right]}{\partial v_k} \\
 &= -\frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n y_i y_j \alpha_i^* \alpha_j^* (2v_k x_{ik} x_{jk}) \\
 &= -\left(\sum_{i=1}^n y_i \alpha_i^* x_{ik} \right)^2 \\
 &= -w_k^2
 \end{aligned}$$

When v_k decreases to 0, J_1 would be approximately increased by the amount of w_k^2 . As $v_k = 0$ leads to removal of the k 'th feature, we come to the same conclusion that removing the feature with smallest w_k^2 will cause the least increase of J_1 .

Therefore, the evaluation criterion of SVM RFE is equivalent to the generalized $\|w\|^2$ measure. The feature subset with minimal generalized $\|w\|^2$ measure is preferred. Our W^2 -IRFS can be considered as an extension of the SVM RFE algorithm. Both of them aim to optimize the same evaluation criterion. The difference is that W^2 -IRFS employs forward search, while SVM RFE employs backward search. The experimental results in Section 7.7 also show that the two algorithms, W^2 -IRFS and SVM RFE, achieve similar performance.

7.7 Experimental Studies

In this section we reported the performance of the two proposed SVM-IRFS algorithms on three publicly available microarray datasets: breast cancer dataset [114], colon cancer dataset [9] and leukaemia dataset [45]. The specifications of the datasets are listed in Tab. 7.1.

We employed a simple pre-processing procedure on each dataset. The microarray

data were directly standardized to zero mean and unit standard variance across genes. No thresholding, filtering, log-transforming and pre-selecting were used in the pre-processing procedure, as we expect to test the performance of our algorithms with a higher dimensional data. .

Dataset	Genes	Samples
Breast cancer	7129	49 (25 ER+, 24 ER-)
Colon cancer	2000	62 (40 tumor, 22 normal)
Leukaemia	7129	72 (47 ALL, 25 AML)

Table 7.1: Description of the datasets

We employed the external .632+ bootstrap [11], which is described as Algorithm 6 in Chapter 5, to investigate the performance of our gene selection algorithms. The balanced bootstrap sample, with $B = 200$ replicates, was used in our experiment.

The parameter C is fixed during the SVM-IRFS gene selection. C is determined by employing the external .632+ bootstrap to estimate the performance of our SVM-IRFS algorithms with a sequence of given values of C , such as $C = 1, 0.1, 0.01, 0.001$. For examples, Fig. 7.1 shows the tuning procedure when R^2W^2 -IRFS ($\epsilon = 0$) is carried out on the breast cancer dataset. The optimal value was chosen to be the one with the minimal estimated error. For the leukaemia dataset, the parameter C is set to 0.1 for both SVM-IRFS algorithms, and for the other two datasets, C is set to 0.01.

Firstly, we analyzed the computational cost of our IRFS algorithm, and compared it with the sequential forward selection (SFS). According to the design of IRFS algorithm, IRFS (with $\epsilon = 0$) would select the same feature subset with SFS. Here we need to investigate how much computational cost the IRFS saves.

The generalized $\|w\|^2$ measure and generalized radius/margin bound are combined with the Sequential Forward Selection (SFS) for comparison. The programs were all executed on a PC (1.6GHz P4 CPU with 1G RAM). The running times (in minutes)

Figure 7.1: The external .632 bootstrap error rate when conducting R^2W^2 -IRFS ($\epsilon = 0$) gene selection with different C 's on the breast cancer dataset.

using gene selection algorithms, respectively, to identify 50 most important genes are listed in Tab. 7.2 and 7.3. The running times of W^2 -IRFS (with $\epsilon = 0$) are only 0.33%, 0.52% and 0.72% of those of W^2 -SFS on the three datasets respectively, and the running times of R^2W^2 -IRFS (with $\epsilon = 0$) are 0.48%, 1.15% and 1.08% of those of R^2W^2 -SFS. The computational cost of IRFS (with $\epsilon = 0$) is extremely lower than that of SFS, but they lead to the same selected feature (gene) subset. Taking W^2 -IRFS on the breast cancer dataset as an example, about 99.67% of computations in the SFS procedure is saved. SVM-IRFS algorithms are very efficient on saving unnecessary computations.

Dataset	W ² -IRFS					W ² -SFS	RFE
	ϵ = 0	ϵ = 1%	ϵ = 2%	ϵ = 5%	ϵ = 10%		
Breast cancer	9.11	1.40	0.68	0.41	0.40	2768.32	71.99
Colon cancer	7.08	0.88	0.81	0.69	0.69	1366.03	28.19
Leukaemia	52.34	25.38	16.66	8.66	5.45	7223.94	174.76

Table 7.2: Running times (in minutes) of the generalized $\|w\|^2$ measure related algorithms, including W^2 -IRFS, W^2 -SFS, and RFE, when 50 genes are selected on the three datasets.

Secondly, we investigated the impact of tolerance ϵ on the IRFS gene selection. If

Dataset	R^2W^2 -IRFS					R^2W^2 -SFS
	$\epsilon = 0$	$\epsilon = 1\%$	$\epsilon = 2\%$	$\epsilon = 5\%$	$\epsilon = 10\%$	
Breast cancer	23.25	8.42	3.43	1.02	0.68	4845.00
Colon cancer	26.77	8.18	4.59	1.89	1.20	2333.16
Leukaemia	140.59	85.65	58.38	30.38	15.18	12586.91

Table 7.3: Running times (in minutes) of the generalized radius/margin bound related algorithms, including R^2W^2 -IRFS and R^2W^2 -SFS, when 50 genes are selected on the three datasets.

the tolerance ϵ is used, much more computational costs can be saved compared with the IRFS algorithm without tolerance ($\epsilon = 0$) as shown in the Tab. 7.2 and 7.3. But how much does this computational saving deteriorate the selection performance? As shown in the Fig. 7.2~7.4, in some cases (W^2 -IRFS on three datasets), the use of ϵ ($\leq 10\%$) results in minor changes in the selection performance, while in other cases (R^2W^2 -IRFS on three datasets), the goodness of the selected feature (gene) subset is affected if greater ϵ was used in the IRFS algorithm. Normally, the selection performance deteriorates when the tolerance ϵ (> 0) is used, but R^2W^2 -IRFS algorithm on leukaemia dataset shows a strange phenomenon that the selection performance without tolerance ($\epsilon = 0$) is even not better than that with tolerance ($\epsilon \leq 5\%$). We will explain the phenomenon later.

To exactly evaluate the influence of the tolerance ϵ , we employed the relative difference, $\frac{J(\epsilon) - J(0)}{J(0)}$, where $J(\epsilon)$ is the value of the evaluation criterion on the selected feature (gene) subset with tolerance ϵ and $J(0)$ is the value for the case of $\epsilon = 0$, to test the selection performance between IRFS algorithms with tolerance ($\epsilon > 0$) and without tolerance ($\epsilon = 0$). The relative difference and corresponding computational saving are reported in Tab. 7.4 and 7.5.

According to the use of ϵ in the IRS (Algorithm 9), the evaluation criterion for selected feature set (J^*) and that for optimal feature set (J_{opt}) should satisfy: $0 \leq \frac{J^* - J_{opt}}{J_{opt}} \leq \epsilon$. However, it is might not be true for the IRFS algorithm which performs IRS several times (in the experiments, 50 times). The IRFS algorithm is

7.2.1: W^2 -IRFS

7.2.2: R^2W^2 -IRFS

Figure 7.2: (1) The value of generalized $\|w\|^2$ measure for W^2 -IRFS on the breast cancer dataset, shown as the number of selected genes. (2) The value of generalized radius/margin bound for R^2W^2 -IRFS on the breast cancer dataset, shown as the number of selected genes. The values of both measures are obtained from the averages across 200 simulation runs on the bootstrap sample sets.

7.3.1: W^2 -IRFS

7.3.2: R^2W^2 -IRFS

Figure 7.3: (1) The value of generalized $\|w\|^2$ measure for W^2 -IRFS on the colon cancer dataset, shown as the number of selected genes. (2) The value of generalized radius/margin bound for R^2W^2 -IRFS on the colon cancer dataset, shown as the number of selected genes. The values of both measures are obtained from the averages across 200 simulation runs on the bootstrap sample sets.

7.4.1: W^2 -IRFS

7.4.2: R^2W^2 -IRFS

Figure 7.4: (1) The value of generalized $\|w\|^2$ measure for W^2 -IRFS on the leukaemia dataset, shown as the number of selected genes. (2) The value of generalized radius/margin bound for R^2W^2 -IRFS on the leukaemia dataset, shown as the number of selected genes. The values of both measures are obtained from the averages across 200 simulation runs on the bootstrap sample sets.

Dataset	$\epsilon = 1\%$	$\epsilon = 2\%$	$\epsilon = 5\%$	$\epsilon = 10\%$
Breast	0.16% (84.63%)	0.21% (92.54%)	0.28% (95.50%)	0.29% (95.61%)
Colon	0.19% (87.57%)	0.20% (88.56%)	0.21% (90.25%)	0.22% (90.25%)
Leukaemia	-0.12% (51.51%)	0.94% (68.17%)	4.04% (83.45%)	6.18% (89.59%)

Table 7.4: The relative differences and computational savings using W^2 -IRFS algorithm at different settings of $\epsilon(> 0)$ on three datasets. Each value outside the parentheses is the relative difference between W^2 -IRFS with tolerance ($\epsilon > 0$) and without tolerance ($\epsilon = 0$). The value in the parentheses is the corresponding computational saving of W^2 -IRFS with tolerance ($\epsilon > 0$) compared with non-tolerance one ($\epsilon = 0$).

Dataset	$\epsilon = 1\%$	$\epsilon = 2\%$	$\epsilon = 5\%$	$\epsilon = 10\%$
Breast	-0.05% (63.78%)	2.62% (85.25%)	7.21% (95.61%)	7.52% (97.08%)
Colon	1.16% (69.44%)	3.58% (82.85%)	7.32% (92.94%)	9.60% (95.52%)
Leukaemia	-4.85% (39.08%)	-6.40% (58.47%)	-3.38% (78.39%)	11.67% (89.20%)

Table 7.5: The relative differences and computational savings using R^2W^2 -IRFS algorithm at different settings of $\epsilon(> 0)$ on three datasets. Each value outside the parentheses is the relative difference between R^2W^2 -IRFS with tolerance ($\epsilon > 0$) and without tolerance ($\epsilon = 0$). The value in the parentheses is the corresponding computational saving of R^2W^2 -IRFS with tolerance ($\epsilon > 0$) compared with non-tolerance one ($\epsilon = 0$).

derived from SFS, and both of them are suboptimal search algorithms. Therefore, it is hard to conclude any relationship between the selected feature set and the optimal one in IRFS algorithms (regardless of the use of tolerance). Even taking the selection performance of IRFS without tolerance ($\epsilon = 0$) as a reference, the inequality $0 \leq \frac{J(\epsilon) - J(0)}{J(0)} \leq \epsilon$ might not hold either, although the relative differences for most experiments fall in this region as shown in Tab. 7.4 and 7.5. There are two categories that need further investigations: (1) $\frac{J(\epsilon) - J(0)}{J(0)} > \epsilon$; and (2) $\frac{J(\epsilon) - J(0)}{J(0)} < 0$.

In the first category, the impact of tolerance on IRFS algorithm is beyond our expectation. However, if the decrease on the selection performance can be compensated with the saving on computations, it is also acceptable to employ tolerance in IRFS. We take R^2W^2 -IRFS with tolerance $\epsilon = 1\%$ on the colon cancer dataset as an example. The use of tolerance lowers the selection performance at 1.16%, which is not a significant change, but saves 66.31% computations, when compared with non-tolerance search.

For the second category, the search with tolerance is better than non-tolerance one. It is really strange. We take R^2W^2 -IRFS with $\epsilon = 1\%$ on the leukaemia dataset as an example to explain the occurrence of this phenomenon. We compared the selection performance of tolerance IRFS ($\epsilon = 1\%$) with that of non-tolerance one ($\epsilon = 0$) on each bootstrap sample set. Among 200 bootstrap sample sets, the tolerance search is better than non-tolerance one on 172 sample sets. The comparison results on some bootstrap sample sets are shown in Fig. 7.5. In Fig. 7.5.1, the first 25 selected genes are exactly the same for tolerance and non-tolerance IRFS algorithms. At the 26th step of IRFS, different gene is selected. The tolerance IRS does not find the best one, and at this time its performance is slightly worse than that of non-tolerance. After that, the IRFS algorithms with and without tolerance are down different routes to optimize the evaluation criterion (in this case, generalized radius/margin bound). At the 27th step, the R^2W^2 value obtained from IRFS with tolerance is already

better than that from non-tolerance one. The reason for this phenomenon is hard to explain. Perhaps the gene selected at the 26'th step of IRFS with tolerance is helpful to decrease the R^2W^2 criterion when combined with the genes selected at the following steps.

According to the results shows in Tab. 7.4 and 7.5, $\epsilon = 1\%$ is an appropriate value for all experiments. The top 25 genes selected by our proposed SVM-IRFS algorithms (with $\epsilon = 1\%$) on three microarray datasets are shown in Tab. 7.6~7.11, respectively. We also compared the test performance of SVM-IRFS ($\epsilon = 1\%$) with SVM-RFE. SVM-RFE is often considered as one of the best gene selection algorithms in the literature. The comparison results are shown in Fig. 7.6. W^2 -IRFS obtains similar performance to SVM-RFE. It is in line with our investigation in Section 7.6 that the criteria used in W^2 -IRFS and SVM-RFE are equivalent. R^2W^2 -IRFS outperforms the other two gene selection algorithms on the breast cancer and leukaemia datasets, but it is a bit inferior on the colon cancer dataset. In fact, the generalization performance of SVMs for separable cases depends on the ratio of radius/margin, and not simply on the large margin (that is, $\|\mathbf{w}\|^2$ measure) [28]. Similarly, for general cases, generalized radius/margin bound illustrates the generalization performance as it is derived as a bound for the leave-one-out error. Therefore as an evaluation criterion, generalized radius/margin bound should outperform generalized $\|\mathbf{w}\|^2$ measure. This is proved on the breast cancer and leukaemia datasets. Whereas on the colon cancer dataset, the generalized radius/margin bound does not achieve better performance than generalized $\|\mathbf{w}\|^2$ measure does.

7.8 Proof of Theorem 7.1

We first prove two lemmas.

Lemma 7.1. *If for 2-norm soft margin SVM Lagrange multipliers α^* satisfy the*

7.5.1: bootstrap sample set #58

7.5.2: #9

7.5.3: #20

7.5.4: #30

7.5.5: #45

7.5.6: #67

7.5.7: #73

7.5.8: #86

7.5.9: #89

7.5.10: #94

Figure 7.5: Comparison of the selection performances of R^2W^2 -IRFS algorithms for $\epsilon = 1\%$ and $\epsilon = 0$ on several particular bootstrap sample sets of leukaemia dataset.

7.6.1: Breast cancer dataset

7.6.2: Colon cancer dataset

7.6.3: Leukaemia dataset

Figure 7.6: The external B.632+ error shown as the number of selected genes, when SVM is employed as the classification algorithm to compare the performance of different gene selection algorithms, including W^2 -IRFS ($\epsilon = 1\%$), R^2W^2 -IRFS ($\epsilon = 1\%$) and SVM-RFE.

No.	Gene No.	Access No.	Gene Description
1*	5914	X58072	Trans-acting T-cell specific transcription factor
2*	6484	X87212	Cathepsin C
3*	1881	L17131	High mobility group protein
4*	1376	HG4716-HT5158	Guanosine 5' -Monophosphate Synthase
5*	6247	X75861	TEGT
6*	2159	L40401	Clone zap128
7*	4234	U30827	Splicing factor SRp40-3
8	2976	M62403	Insulin-like growth factor binding protein 4
9*	2565	M24485	Glutathione S-transferase pi
10	1512	J03827	Y box binding protein-1
11	5438	U96191	Trophoblast hypoxia-regulated factor-5
12	2912	M59815	Complement component C4A
13	5433	U96113	Nedd-4-like ubiquitin-protein ligase WWP1
14*	5591	X06700	Pro-alpha1(III) collagen
15	3377	M97252	Kallmann syndrome
16	918	D89667	c-myc binding protein
17	1790	L11329	Protein tyrosine phosphatase
18	15	AB000468	Zinc finger protein
19	6904	Z19002	Kruppel-like zinc finger protein
20	1982	L26336	Heat shock protein HSPA2
21*	3442	S62539	Insulin receptor substrate-1
22	1095	HG2715-HT2811	Tyrosine Kinase
23*	5884	X56667	Calretinin
24	5280	U84011	Glycogen debranching enzyme isoform 6
25	822	D86985	KIAA0232

Table 7.6: The top 25 genes selected by W^2 -IRFS algorithm ($\epsilon = 1\%$) on the breast cancer dataset. The genes marked with * are also selected by R^2W^2 -IRFS.

No.	Gene No.	Access No.	Gene Description
1	4414	U39840	Hepatocyte nuclear factor-3 alpha
2*	5914	X58072	Trans-acting T-cell specific transcription factor
3*	6247	X75861	TEGT
4*	1881	L17131	High mobility group protein
5*	6484	X87212	Cathepsin C
6*	4234	U30827	Splicing factor SRp40-3
7*	5591	X06700	Pro-alpha1(III) collagen
8	4220	U29656	DR-nm23
9*	2159	L40401	Clone zap128
10	7006	Z47553	Flavin-containing monooxygenase 5
11	6476	X86693	Hevin like protein
12	219	D12485	Nucleotide pyrophosphatase
13*	1376	HG4716-HT5158	Guanosine 5' -Monophosphate Synthase
14*	2565	M24485	Glutathione S-transferase pi
15	2487	M21005	Migration inhibitory factor-related protein 8
16	3624	U00115	Zinc-finger protein (bcl-6)
17*	3442	S62539	Insulin receptor substrate-1
18	3796	U09196	1.1 kb mRNA upregulated in retinoic acid treated HL-60 neutrophilic cells
19*	5884	X56667	Calretinin
20	842	D87258	Serin protease
21	6482	X87176	17-beta-hydroxysteroid dehydrogenase
22	7064	Z69043	Translocon-associated protein delta
23	2604	M26730	Mitochondrial ubiquinone-binding protein
24	2779	M34516	Omega light chain protein 14.1
25	4450	U41371	Spliceosome associated protein

Table 7.7: The top 25 genes selected by R^2W^2 -IRFS algorithm ($\epsilon = 1\%$) on the breast cancer dataset. The genes marked with * are also selected by W^2 -IRFS.

No.	Gene No.	Access No.	Gene Description
1*	249	M63391	Desmin gene
2*	1772	H08393	Collagen alpha 2(XI)
3*	377	Z50753	GCAP-II/uroguanylin
4	765	M76378	Cysteine-rich protein
5*	1870	H55916	Peptidyl-prolyl cis-trans isomerase
6	1582	X63629	p cadherin
7	493	R87126	Myosin heavy chain, nonmuscle [Gallus gallus]
8*	1346	T62947	60S ribosomal protein L24 [Arabidopsis thaliana]
9*	66	T71025	Metallothionein 1G
10	1644	R80427	C4-dicarboxylate transport sensor protein [Rhizobium leguminosarum]
11*	1423	J02854	Myosin regulatory light chain 2
12	1993	L11706	Hormone-sensitive lipase
13*	1024	R65697	ATP synthase A chain [Trypanosoma brucei brucei]
14	286	H64489	Leukocyte antigen CD37
15	1482	T64012	Acetylcholine receptor protein, delta chain
16	974	U00968	Sterol regulatory element binding protein 1
17	1641	K02268	Enkephalin B
18	1504	H78386	Interleukin receptor
19	802	X70326	MacMarcks
20*	245	M76378	Cysteine-rich protein
21*	1873	L07648	MXI1
22	1546	T51493	PP2A B56-gamma1
23	1976	K03474	Mullerian inhibiting substance (MIS, AMH)
24	617	H67764	Estrogen sulfotransferase [Bos taurus]
25	1241	T64885	General negative regulator of transcription subunit 1 [Saccharomyces cerevisiae]

Table 7.8: The top 25 genes selected by W^2 -IRFS algorithm ($\epsilon = 1\%$) on the colon cancer dataset. The genes marked with * are also selected by R^2W^2 -IRFS.

No.	Gene No.	Access No.	Gene Description
1*	249	M63391	Desmin gene
2	138	M26697	Nucleolar protein (B23)
3*	245	M76378	Cysteine-rich protein
4*	377	Z50753	GCAP-II/uroguanylin
5*	1870	H55916	Peptidyl-prolyl cis-trans isomerase
6	1668	M82919	Gamma amino butyric acid receptor beta-3
7	515	T56604	Tubulin beta chain
8*	66	T71025	Metallothionein 1G
9	1972	T57468	Fibrillarin
10*	1423	J02854	Myosin regulatory light chain 2
11	1058	M80815	a-L-fucosidase
12	1466	D00860	Ribose-phosphate pyrophosphokinase I
13	1335	X83301	SMA5
14	988	L08044	Intestinal trefoil factor
15	267	M76378	Cysteine-rich protein
16	799	J03210	Collagenase type IV
17	1248	D43950	KIAA0098
18	1671	M26383	Monocyte-derived neutrophil-activating protein
19*	1346	T62947	60S ribosomal protein L24 [Arabidopsis thaliana]
20	70	T61661	Profilin I
21*	1024	R65697	ATP synthase A chain [Trypanosoma brucei brucei]
22*	1873	L07648	MXI1
23	1954	M35531	GDP-L-fucose:beta-D-galactoside 2-alpha-l-fucosyltransferase
24*	1772	H08393	Collagen alpha 2(XI)
25	1608	L12723	Heat shock protein 70

Table 7.9: The top 25 genes selected by R^2W^2 -IRFS algorithm ($\epsilon = 1\%$) on the colon cancer dataset. The genes marked with * are also selected by W^2 -IRFS.

No.	Gene No.	Access No.	Gene Description
1*	4847	X95735	Zyxin
2	804	HG1612-HT1612	Macmarcks
3*	4951	Y07604	Nucleoside-diphosphate kinase
4	1081	J03589	Ubiquitin-like protein GDX
5	1882	M27891	Cystatin C
6	6539	X85116	Epb72
7	5107	Z29067	Nek3
8*	5348	M61853	CYP2C18 Cytochrome P450
9	1834	M23197	CD33 antigen
10	1975	M34344	ITGA2B Integrin
11*	1779	M19507	MPO Myeloperoxidase
12	3847	U82759	GB DEF = Homeodomain protein HoxA9
13	1674	M11147	FTL Ferritin, light polypeptide
14*	4196	X17042	PRG1 Proteoglycan 1
15	3391	U53468	NADH:ubiquinone oxidoreductase subunit B13
16	1796	M20902	Apolipoprotein CI
17	1829	M22960	Protective protein for beta-galactosidase
18	2288	M84526	Adipsin (D component of complement)
19	3320	U50136	Leukotriene C4 synthase
20	1928	M31303	Oncoprotein 18
21	6169	M13690	C1NH Complement component 1 inhibitor
22	5119	Z30426	CD69 antigen
23*	5002	Y10207	GB DEF = CD171 protein
24	3252	U46499	Glutathione S-transferase
25	4955	Y07755	S100A2

Table 7.10: The top 25 genes selected by W^2 -IRFS algorithm ($\epsilon = 1\%$) for the leukaemia cancer dataset. The genes marked with * are also selected by R^2W^2 -IRFS.

No.	Gene No.	Access No.	Gene Description
1*	4847	X95735	Zyxin
2*	4951	Y07604	Nucleoside-diphosphate kinase
3	4389	X63469	General transcription factor TFIIE beta
4	3714	U77604	Microsomal glutathione S-transferase
5	2354	M92287	Cyclin D3
6	1465	L31881	Nuclear factor I/X
7	1546	L38929	Protein tyrosine phosphatase, receptor type, delta
8	1962	M33680	Cell surface protein TAPA-1
9*	5348	M61853	CYP2C18 Cytochrome P450
10	1369	L19437	TALDO Transaldolase
11	2138	M63896	GB DEF = Transcriptional enhancer factor (TEF1)
12	6184	M26708	Prothymosin alpha
13	6225	M84371	CD19
14	3085	U34879	17-beta-hydroxysteroid dehydrogenase
15	5925	X01038	Fetal gene for apolipoprotein AI
16*	4196	X17042	PRG1 Proteoglycan 1
17	4609	X78669	ERC-55
18	5094	Z24727	Tropomyosin alpha chain
19	461	D49950	Interferon-gamma inducing factor
20	3017	U30828	Splicing factor SRp55-2
21	2295	M85169	Homologue of yeast sec7
22	3344	U51240	KIAA0085
23	2111	M62762	ATP6C Vacuolar H+ ATPase
24*	5002	Y10207	GB DEF = CD171 protein
25*	1779	M19507	MPO Myeloperoxidase

Table 7.11: The top 25 genes selected by R^2W^2 -IRFS algorithm ($\epsilon = 1\%$) on the leukaemia cancer dataset. The genes marked with * are also selected by W^2 -IRFS.

following optimization problem,

$$\begin{aligned} \text{maximize } \mathcal{W}(\boldsymbol{\alpha}) &= \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n y_i y_j \alpha_i \alpha_j (\mathbf{x}_i^T \mathbf{x}_j + \frac{1}{C} \delta_{ij}) \\ \text{subject to } \sum_{i=1}^n y_i \alpha_i &= 0, \\ \alpha_i &\geq 0, \quad i = 1, \dots, n. \end{aligned} \quad (7.26)$$

then,

$$\sum_{i=1}^n \alpha_i^* = 2\mathcal{W}(\boldsymbol{\alpha}^*). \quad (7.27)$$

Proof. According to the strong duality theorem and KKT conditions (refer to Appendix A),

$$\begin{aligned} \mathcal{W}(\boldsymbol{\alpha}^*) &= \mathcal{P}(\mathbf{w}^*, b^*, \boldsymbol{\xi}^*) \\ &= \frac{1}{2} \|\mathbf{w}^*\|^2 + \frac{C}{2} \sum_{i=1}^n \xi_i^{*2} \end{aligned} \quad (7.28)$$

Putting the solutions

$$\begin{aligned} \mathbf{w}^* &= \sum_{i=1}^n y_i \alpha_i^* \mathbf{x}_i, \\ \xi_i^* &= \frac{\alpha_i^*}{C} \end{aligned} \quad (7.29)$$

into Eq. (7.28), we obtain

$$\mathcal{W}(\boldsymbol{\alpha}^*) = \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n y_i y_j \alpha_i^* \alpha_j^* (\mathbf{x}_i^T \mathbf{x}_j + \frac{1}{C} \delta_{ij}). \quad (7.30)$$

Note that α^* is the solution for the optimization problem (7.26), so

$$\begin{aligned}\mathcal{W}(\alpha^*) &= \sum_{i=1}^n \alpha_i^* - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n y_i y_j \alpha_i^* \alpha_j^* (\mathbf{x}_i^T \mathbf{x}_j + \frac{1}{C} \delta_{ij}) \\ &= \sum_{i=1}^n \alpha_i^* - \mathcal{W}(\alpha^*)\end{aligned}$$

Therefore, $\sum_{i=1}^n \alpha_i^* = 2\mathcal{W}(\alpha^*)$. □

Lemma 7.2. *If in the leave-one-out procedure for 2-norm soft margin SVM $\mathbf{x}_l$ is recognized incorrectly, then the inequality*

$$\alpha_l^* \geq \frac{1}{D^2 + 2/C} \quad (7.31)$$

holds, where D is the diameter of the smallest sphere containing all training samples, $\mathbf{x}_1, \dots, \mathbf{x}_n$.

Proof. Assume that using all training samples, we have constructed the 2-norm soft margin hyperplane,

$$f^*(\mathbf{x}) = \mathbf{w}^{*T} \mathbf{x} + b^* = 0 \quad (7.32)$$

where

$$\mathbf{w}^* = \sum_{i=1}^n y_i \alpha_i^* \mathbf{x}_i, \quad (7.33)$$

and α^* is obtained from the following quadratic programming problem

$$\begin{aligned}\text{maximize} \quad & \mathcal{W}(\alpha) = \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n y_i y_j \alpha_i \alpha_j (\mathbf{x}_i^T \mathbf{x}_j + \frac{1}{C} \delta_{ij}) \\ \text{subject to} \quad & \sum_{i=1}^n y_i \alpha_i = 0, \\ & \alpha_i \geq 0, \quad i = 1, \dots, n.\end{aligned} \quad (7.34)$$

Assume that in the leave-one-out procedure sample $\mathbf{x}_l$ makes a leave-one-out error.

That means the inequality,

$$y_l f^l(\mathbf{x}_l) = y_l(\mathbf{w}^{lT} \mathbf{x}_l + b^l) < 0, \quad \mathbf{w}^l = \sum_{i=1}^n y_i \alpha_i^l \mathbf{x}_i, \quad (7.35)$$

holds, where $\boldsymbol{\alpha}^l = (\alpha_1^l, \dots, \alpha_n^l)^T$ is the solution to maximize the quadratic programming problem (7.34) with an additional constraint,

$$\alpha_l = 0. \quad (7.36)$$

Since $\boldsymbol{\alpha}^l$ maximizes the problem (7.34) under an additional constraint (7.36), the following inequality holds,

$$\mathcal{W}(\boldsymbol{\alpha}^l) \geq \mathcal{W}(\boldsymbol{\alpha}^* - \boldsymbol{\gamma}) \quad (7.37)$$

where the vector $\boldsymbol{\gamma} = (\gamma_1, \dots, \gamma_n)^T$ satisfies the following conditions:

$$\begin{aligned} \gamma_l &= \alpha_l^*, \\ \sum_{i=1}^n y_i \gamma_i &= 0, \\ \alpha_i^* - \gamma_i &\geq 0, \quad i = 1, \dots, n, \\ \text{if } \alpha_i^* &= 0, \text{ then } \gamma_i = 0. \end{aligned}$$

Similarly since $\boldsymbol{\alpha}^*$ maximizes the problem (7.34), the following inequality holds.

$$\mathcal{W}(\boldsymbol{\alpha}^*) \geq \mathcal{W}(\boldsymbol{\alpha}^l + \boldsymbol{\beta}) \quad (7.38)$$

where $\boldsymbol{\beta} = (\beta_1, \dots, \beta_n)^T$ is a vector satisfying the constraint

$$\begin{aligned} \sum_{i=1}^n y_i \beta_i &= 0, \\ \alpha_i^l + \beta_i &\geq 0, \quad i = 1, \dots, n, \\ \text{if } \alpha_i^l &= 0, \text{ then } \beta_i = 0, \quad i \neq l. \end{aligned}$$

We obtain

$$\mathcal{W}(\boldsymbol{\alpha}^l + \boldsymbol{\beta}) - \mathcal{W}(\boldsymbol{\alpha}^l) \leq \mathcal{W}(\boldsymbol{\alpha}^*) - \mathcal{W}(\boldsymbol{\alpha}^l) \leq \mathcal{W}(\boldsymbol{\alpha}^*) - \mathcal{W}(\boldsymbol{\alpha}^* - \boldsymbol{\gamma}) \quad (7.39)$$

We denote $I_1 = \mathcal{W}(\boldsymbol{\alpha}^l + \boldsymbol{\beta}) - \mathcal{W}(\boldsymbol{\alpha}^l)$ and $I_2 = \mathcal{W}(\boldsymbol{\alpha}^*) - \mathcal{W}(\boldsymbol{\alpha}^* - \boldsymbol{\gamma})$. The maximum of I_1 is equal to the minimum of I_2 , and both of them are $\mathbf{W}(\boldsymbol{\alpha}^*) - \mathbf{W}(\boldsymbol{\alpha}^l)$.

$$\begin{aligned} I_1 &= \mathcal{W}(\boldsymbol{\alpha}^l + \boldsymbol{\beta}) - \mathcal{W}(\boldsymbol{\alpha}^l) \\ &= \sum_i \beta_i - \sum_{i,j} y_i y_j \beta_i \alpha_j^l (\mathbf{x}_i^T \mathbf{x}_j + \frac{1}{C} \delta_{ij}) - \frac{1}{2} \sum_{i,j} y_i y_j \beta_i \beta_j (\mathbf{x}_i^T \mathbf{x}_j + \frac{1}{C} \delta_{ij}) \\ &= \sum_i \beta_i (1 - y_i \sum_j y_j \alpha_j^l \mathbf{x}_i^T \mathbf{x}_j - \frac{y_i}{C} \sum_j y_j \alpha_j^l \delta_{ij}) - \frac{1}{2} \sum_{i,j} y_i y_j \beta_i \beta_j (\mathbf{x}_i^T \mathbf{x}_j + \frac{1}{C} \delta_{ij}) \\ &= \sum_i \beta_i (1 - y_i \mathbf{w}^{lT} \mathbf{x}_i - \xi_i^l) - \frac{1}{2} \sum_{i,j} y_i y_j \beta_i \beta_j (\mathbf{x}_i^T \mathbf{x}_j + \frac{1}{C} \delta_{ij}) \\ &= \sum_{i \neq l} \beta_i [1 - y_i (\mathbf{w}^{lT} \mathbf{x}_i + b^l) - \xi_i^l] + \beta_l [1 - y_l (\mathbf{w}^{lT} \mathbf{x}_l + b^l)] \\ &\quad - \frac{1}{2} \sum_{i,j} y_i y_j \beta_i \beta_j (\mathbf{x}_i^T \mathbf{x}_j + \frac{1}{C} \delta_{ij}) \\ &= \beta_l [1 - y_l f^l(\mathbf{x}_l)] - \frac{1}{2} \sum_{i,j} y_i y_j \beta_i \beta_j (\mathbf{x}_i^T \mathbf{x}_j + \frac{1}{C} \delta_{ij}) \end{aligned}$$

We consider vector $\boldsymbol{\beta} = a\boldsymbol{\mu}$, where $a \geq 0$ is some constant, and $\boldsymbol{\mu}$ is defined as follows:

$$\begin{aligned} \mu_l &= 1, \\ \sum_{i=1}^n y_i \mu_i &= 0. \end{aligned}$$

For this vector $\boldsymbol{\beta}$, we obtain

$$I_1 = a[1 - y_l f^l(\mathbf{x}_l)] - \frac{a^2}{2} [(\sum_i y_i \mu_i \mathbf{x}_i)^T (\sum_i y_i \mu_i \mathbf{x}_i) + \frac{1}{C} \sum_i \mu_i^2]. \quad (7.40)$$

We choose value of a as follows to maximize the above expression,

$$a = \frac{1 - y_l f^l(\mathbf{x}_l)}{(\sum_i y_i \mu_i \mathbf{x}_i)^T (\sum_i y_i \mu_i \mathbf{x}_i) + \frac{1}{C} \sum_i \mu_i^2}.$$

Putting this expression back into (7.40), we obtain

$$\begin{aligned} I_1 &= \frac{[1 - y_l f^l(\mathbf{x}_l)]^2}{2[(\sum_i y_i \mu_i \mathbf{x}_i)^T (\sum_i y_i \mu_i \mathbf{x}_i) + \frac{1}{C} \sum_i \mu_i^2]} \\ &\geq \frac{1}{2[(\sum_i y_i \mu_i \mathbf{x}_i)^T (\sum_i y_i \mu_i \mathbf{x}_i) + \frac{1}{C} \sum_i \mu_i^2]}. \end{aligned} \quad (7.41)$$

Now we estimate the right hand side of inequality (7.39),

$$\begin{aligned} I_2 &= \mathcal{W}(\boldsymbol{\alpha}^*) - \mathcal{W}(\boldsymbol{\alpha}^* - \boldsymbol{\gamma}) \\ &= \sum_i \gamma_i - \sum_{i,j} y_i y_j \gamma_i \alpha_j^* (\mathbf{x}_i^T \mathbf{x}_j + \frac{1}{C} \delta_{ij}) + \frac{1}{2} \sum_{i,j} y_i y_j \gamma_i \gamma_j (\mathbf{x}_i^T \mathbf{x}_j + \frac{1}{C} \delta_{ij}) \\ &= \sum_i \gamma_i (1 - y_i \mathbf{x}_i^T \sum_j y_j \alpha_j^* \mathbf{x}_j - \frac{y_i}{C} \sum_j y_j \alpha_j^* \delta_{ij}) + \frac{1}{2} \sum_{i,j} y_i y_j \gamma_i \gamma_j (\mathbf{x}_i^T \mathbf{x}_j + \frac{1}{C} \delta_{ij}) \\ &= \sum_i \gamma_i [1 - y_i (\mathbf{w}^{*T} \mathbf{x}_i + b^*) - \xi_i] + \frac{1}{2} \sum_{i,j} y_i y_j \gamma_i \gamma_j (\mathbf{x}_i^T \mathbf{x}_j + \frac{1}{C} \delta_{ij}) \\ &= \frac{1}{2} \sum_{i,j} y_i y_j \gamma_i \gamma_j (\mathbf{x}_i^T \mathbf{x}_j + \frac{1}{C} \delta_{ij}) \\ &= \frac{1}{2} [(\sum_i y_i \gamma_i \mathbf{x}_i)^T (\sum_i y_i \gamma_i \mathbf{x}_i) + \frac{1}{C} \sum_i \gamma_i^2]. \end{aligned} \quad (7.42)$$

Note that if a sample $\mathbf{x}_l$ is committed as a leave-one-out error the corresponding multiplier $\alpha_l^* \neq 0$. We consider vector $\boldsymbol{\lambda}$ as,

$$\lambda_i = \gamma_i / \alpha_l^*. \quad (7.43)$$

Then we have

$$\begin{aligned}\lambda_l &= 1, \\ \sum_i y_i \lambda_i &= 0.\end{aligned}$$

Putting the Eq. (7.43) into Eq. (7.42),

$$I_2 = \frac{(\alpha_l^*)^2}{2} \left[\left(\sum_i y_i \lambda_i \mathbf{x}_i \right)^T \left(\sum_i y_i \lambda_i \mathbf{x}_i \right) + \frac{1}{C} \sum_i \lambda_i^2 \right] \quad (7.44)$$

According to inequalities (7.39), (7.41) and (7.44), we have

$$(\alpha_l^*)^2 \geq \frac{1}{\left[\left(\sum_i y_i \mu_i \mathbf{x}_i \right)^T \left(\sum_i y_i \mu_i \mathbf{x}_i \right) + \frac{1}{C} \sum_i \mu_i^2 \right] \left[\left(\sum_i y_i \lambda_i \mathbf{x}_i \right)^T \left(\sum_i y_i \lambda_i \mathbf{x}_i \right) + \frac{1}{C} \sum_i \lambda_i^2 \right]}. \quad (7.45)$$

We assign vector $\boldsymbol{\mu}$ as follows:

$$\begin{aligned}\mu_l &= 1, \\ \mu_i &= 0, \text{ such that } y_i = y_l, \\ \mu_i &\geq 0, \text{ such that } y_i \neq y_l.\end{aligned}$$

It is easy to verify that such $\boldsymbol{\mu}$ is valid. Therefore $\left(\sum_i y_i \mu_i \mathbf{x}_i \right)^T \left(\sum_i y_i \mu_i \mathbf{x}_i \right)$ describes the square of the distance between two vectors: one of which is $\mathbf{x}_l$ and the other belongs to the convex hull of support vectors of the opposite class. Since both of them are in the smallest sphere containing all samples, the distance does not exceed D , that is,

$$\left(\sum_i y_i \mu_i \mathbf{x}_i \right)^T \left(\sum_i y_i \mu_i \mathbf{x}_i \right) \leq D^2.$$

And we also have,

$$\begin{aligned}\sum_i \mu_i^2 &= \sum_{i \neq l} \mu_i^2 + \mu_l^2 \\ &\leq \left(\sum_{i \neq l} \mu_i \right)^2 + 1 = 2.\end{aligned}$$

Similarly,

$$\begin{aligned}\left(\sum_i y_i \lambda_i \mathbf{x}_i \right)^T \left(\sum_i y_i \lambda_i \mathbf{x}_i \right) &\leq D^2 \\ \sum_i \lambda_i^2 &\leq 2\end{aligned}\tag{7.46}$$

Combining all together, we obtain,

$$\alpha_i^* \geq \frac{1}{D^2 + 2/C}.\tag{7.47}$$

Lemma has been proved. $\square$

Theorem. For 2-norm soft margin SVMs, the leave-one-out error L meets the following inequality,

$$L \leq \frac{8}{n} \mathcal{W}(\boldsymbol{\alpha}^*) \left(R^2 + \frac{1}{2C} \right)$$

where n is the size of training samples, R is the radius of the smallest sphere that contains all the training samples, $\mathcal{W}(\cdot)$ is defined in the dual problem (7.5), and $\boldsymbol{\alpha}^*$ is the solution of optimization problem (7.5).

Proof. Consider a training sample $\mathbf{x}_l$ incorrectly classified in the leave-one-out procedure. Then Lemma 7.2 gives

$$\alpha_l^* \geq \frac{1}{D^2 + 2/C}.$$

Now we sum the left and right hand sides of this inequality over all training samples where the leave-one-out procedure makes an error

$$\frac{nL}{D^2 + 2/C} \leq \sum_{\#} \alpha_i^* \leq \sum_{i=1}^n \alpha_i^*$$

Here $\sum_{\#}$ indicates that the sum is taken only over samples which are incorrectly classified in the leave-one-out procedure. According to Lemma 7.1, we have

$$\begin{aligned} L &\leq \frac{1}{n} \left(\sum_{i=1}^n \alpha_i^* \right) \left(D^2 + \frac{2}{C} \right) \\ &= \frac{2}{n} \mathcal{W}(\boldsymbol{\alpha}^*) \left(4R^2 + \frac{2}{C} \right) \\ &= \frac{8}{n} \mathcal{W}(\boldsymbol{\alpha}^*) \left(R^2 + \frac{1}{2C} \right). \end{aligned}$$

□

Chapter 8

Conclusions and Recommendations

8.1 Conclusions

Current microarray experiments allow expression levels of thousands of genes to be measured at once to provide complete transcription information in cells. Microarray data is like a golden mine, from which a lot can be explored. One of the explorations is to identify gene markers that present the maximum discriminant power between different conditions of interest, such as tumor and normal tissues. Gene selection has emerged as a major issue of microarray data analysis.

The goodness of a selected feature subset is often evaluated in terms of its predictive capacity. Hence it is nature to consider the error rate as an evaluation criterion in the wrapper-based feature selection. We have investigated the performance of several error counting-based wrapper methods in gene selection, including the re-substitution error, leave-one-out error, cross-validation error and bootstrap error on three benchmark microarray datasets. Our experiments show that error counting-

based wrapper methods might select less significant genes when more than one gene subset result in a same estimated error. We believe that the main cause of the ties problem is the discrete nature of counting-based error estimators. We have tested and suggested two continuous criteria for gene selection. Experimental results show that well-designed continuous evaluation criteria, such as the generalized $\|\mathbf{w}\|^2$ measure for Support vector machines and the modified Relief's measure for k -Nearest Neighbors, outperform the counting-based error estimators in terms of stability and predictive accuracy for gene selection on high dimensional and small-sized microarray data.

Knowing that the ties problem arises from discrete evaluation criteria would deteriorate the gene selection performance, we focus on designing novel gene selection algorithms using continuous evaluation criteria. In this thesis, three new gene selection algorithms, including LS Bound-based gene selection (Chapter 5), RN-based gene selection (Chapter 6) and SVM-IRFS (Chapter 7), are proposed.

The LS Bound-based gene selection employs the LS Bound measure we developed as the evaluate criterion. This approach can be considered as a hybrid of filter and wrapper methods. On one hand, the LS Bound measure is derived from the leave-one-out procedure of LS-SVMs. As an upper bound for results of leave-one-out classification, the LS Bound measure has direct relation to the generalization performance of LS-SVMs. Consequently, the obtained gene subset can achieve more accurate classification results than the filter method. On the other hand, unlike the error rate-based wrapper method, the LS Bound measure does not demand repeated training for cross validation. The training procedure involved is implicitly expressed in Eq. (5.13), whose computational complexity is similar to that of the filter method.

The RN-based gene selection algorithm we proposed has the same strengths as the LS Bound-based gene selection algorithm, because its evaluation criterion is also derived from the leave-one-out procedure and the computation of the RN-based

criterion would not involve repeated training. The main difference between RN-based algorithm and LS Bound-based algorithm is that the former calculates the exact leave-one-out classification results instead of estimating their bound in the latter one, which leads to an increased classification accuracy. One similar example is that recently [104] employed an accurate leave-one-out estimation [26] instead of the LS Bound measure we proposed in Chapter 5 to improve the gene selection performance.

SVM-IRFS algorithms combine SVM-based evaluation criteria with a new search procedure, Iterative Reduced Forward Search (IRFS), to address the gene selection problem on high dimensional microarray data. Two SVM-based criteria, generalized $\|\mathbf{w}\|^2$ measure and generalized radius/margin bound, are employed in this work. These criteria are modified from the well-known separating margin and radius/margin bound with applications to both separable and non-separable cases. To reduce the computations of SVM-based criteria, we developed a new search procedure, IRFS, to avoid unnecessary computational cost. Our experimental studies show that IRFS algorithm achieves similar (or identical, with tolerance $\epsilon = 0$) performance as SFS (sequential forward selection) does, but the computational cost of our IRFS algorithm is less than 1% of SFS.

The effectiveness of our proposed gene selection algorithms has been tested on three benchmark microarray datasets: leukaemia dataset [45], breast cancer dataset [114] and colon cancer dataset [9]. Because the test procedure of SVM-IRFS algorithms carried out in Chapter 7 employs a different pre-processing technique. We repeat the evaluation of SVM-IRFS algorithms on the pre-processed datasets used in Chapter 4~6. The running times (in minutes) using these algorithms, respectively, to identify 50 most important genes (from 1000 genes) on three benchmark datasets are listed in Tab. 8.1. The programs were all executed in MATLAB 7.0 environment (2.5GHz P4 CPU with 512MB RAM). The test errors of our proposed algorithms are compared

in Fig. 8.1. The RN-based gene selection is the fastest. R^2W^2 -IRFS, one of the SVM-IRFS algorithms, achieves the most accurate prediction on the leukaemia and breast cancer datasets, and is a bit inferior to RN-based gene selection on the colon cancer dataset, but the computational cost is the highest among the algorithms we proposed.

Dataset	LS Bound-based gene selection	RN-based gene selection	SVM-IRFS	
			W^2 -IRFS	R^2W^2 -IRFS
Leukaemia	0.87	1.34	9.59	27.58
Breast cancer	0.45	0.72	1.51	4.71
Colon cancer	0.76	1.09	1.39	2.90

Table 8.1: The running times (in minutes) using our proposed algorithms, including LS Bound-based gene selection, RN-based gene selection and two SVM-IRFS algorithms (W^2 -IRFS and R^2W^2 -IRFS), respectively, to identify 50 most important genes (from 1000 genes) on three datasets

8.2 Recommendations for Further Research

Some possible aspects for future work are listed as follows:

- **Fast implementation of SVM-IRFS algorithms.** In Chapter 7, we proposed SVM-IRFS algorithms for gene selection. Although IRFS is efficient in saving unnecessary computations, the computational cost is still quite high due to the high computational complexity of SVM-based criterion, which involves one or two QP problems. Especially, when more samples are in the dataset, the running time to compute SVM-based criteria would be exponentially increasing. Therefore, fast implementation of SVM-IRFS algorithms would be of importance for real applications. In present study, LOQO algorithm [110] is employed to solve the QP problems arising from SVM training and minimum enclosing ball problem. For SVM training, other improved algorithm, such as Sequential Minimal Optimization (SMO) [86], SVMTorch [30], LIBSVM [27],

8.1.1: Leukaemia dataset

8.1.2: Breast cancer dataset

8.1.3: Colon cancer dataset

Figure 8.1: The external B.632+ error shown as the number of selected genes, when SVM is employed as the classification algorithm to compare the performance of four gene selection algorithms we proposed, including LS Bound-based gene selection, RN-based gene selection, W^2 -IRFS and R^2W^2 -IRFS.

or SVM^{light} [56] can be used to reduce the computations, while minimum enclosing ball problem can be solved by $(1 + \epsilon)$ -approximation algorithm [24].

- **Tighter error bound for SVMs.** We have proposed an error bound, generalized radius/margin bound, to address the gene selection problem, and it has achieved quite satisfactory performance. However, as an error upper bound, the generalized radius/margin bound is not very tight, which may not exactly reflect the generalization performance. A tighter error bound could help to select genes with better predictive accuracy.
- **Extension of SVM-IRFS to multi-class SVMs.** In the literature of SVMs, the approach to solving multi-class problem is often implemented by reducing a single multi-class problem into a collection of multiple independent binary problems. There are two typical methods in this way: one-versus-all [112] and one-versus-one [65]. While Weston *et al.* [115], Crammer *et al.* [31] and Lee *et al.* [67] proposed three different frameworks, respectively, to solve the multi-class problem by considering all the classes simultaneously. The generalization performance for the frameworks of one-versus-all and one-versus-one can be assessed by combination of binary problems. We need further investigations on the generalization performance for frameworks of Weston, Crammer and Lee, respectively. The criteria related to generalization can be combined with our proposed IRFS algorithm to perform gene selection on multi-class microarray data.
- **Unsupervised gene clustering.** In this thesis, we only consider the gene (feature) selection in a supervised manner. However, many microarray problems are grouped as unsupervised when only expression data are available. In these problems the aim is to select distinct gene subsets with similar expressions suggesting that the genes in a subset may be biological related to the problem under investigation. The unsupervised gene clustering is of biological

significance in practice, and is an interesting topic for further research.

- **Comparison between feature selection and feature extraction.** Feature selection and feature extraction are two major categories of dimensionality reduction methods. Features generated by the feature extraction approach may not have clear meanings in the real world, while those provided by feature selection approaches retain their original meanings. That is the reason why we prefer feature selection rather than feature extraction. However, if only considering the classification performance, we do not have a clear understanding about which one is better. It is worthy of further investigation.

Author's Publications

1. Zhou, X. and Mao, K. Z. (2005) LS Bound based gene selection for DNA microarray data. *Bioinformatics*, 21(8), 1559-1564.
2. Zhou, X. and Mao, K. Z. (2005) Gene selection of DNA microarray data based on regularization network. Gallapher, M., Hogan, J. and Maire, F. (Eds.): *The Sixth International Conference on Intelligent Data Engineering and Automated Learning (IDEAL'05)*, *Lecture Notes in Computer Science*, 3578, 414-421.
3. Zhou, X. and Mao, K. Z. (2006) Regularization network-based gene selection for DNA microarray data. Special Issue on Adaptive Neural Methods for Intelligent Data Analysis, *International Journal of Neural Systems*, 16(5), 341-352.
4. Zhou, X. and Mao, K. Z. (2006) The ties problem resulting from counting-based error estimators and its impact on gene selection algorithms. *Bioinformatics*, 22(20), 2507-2515.
5. Zhou, X., Wu, X. Y. , Tuck, D. P. and Mao, K. Z. (2007) Fast gene selection for microarray data using SVM-based criteria. Submitted to *Bioinformatics*.

Bibliography

- [1] "The chipping forecast," *Special Supplement to Nature Genetics*, vol. 21, Jan 1999.
- [2] "The chipping forecast ii," *Special supplement to Nature Genetics*, vol. 32, 2002.
- [3] Affymetrix, *Affymetrix microarray suite users guide*, 4th ed. Santa Clara, CA: Affymetrix, 1999.
- [4] —, *Affymetrix microarray suite users guide*, 5th ed. Santa Clara: Affymetrix, 2001.
- [5] —, "Statistical algorithms reference guide," Affymetrix, Santa Clara, CA, Tech. Rep., 2001.
- [6] —, "Statistical algorithms description document, 4th ed." Affymetrix, Santa Clara, CA, Tech. Rep., 2002.
- [7] A. A. Alizadeh, M. B. Eisen, R. E. Davis, C. Ma, I. S. Lossos, A. Rosenwald, J. C. Boldrick, H. Sabet, T. Tran, X. Yu, J. I. Powell, L. Yang, G. E. Marti, T. Moore, J. James Hudson, L. Lu, D. B. Lewis, R. Tibshirani, G. Sherlock, W. C. Chan, T. C. Greiner, D. D. Weisenburger, J. O. Armitage, R. Warnke, R. Levy, W. Wilson, M. R. Grever, J. C. Byrd, D. Botstein, P. O. Brown, and L. M. Staudt, "Distinct types of diffuse large B-cell lymphoma identified by gene expression profiling," *Nature*, vol. 403, pp. 503–511, 2000.
- [8] S. V. Allander, N. N. Nupponen, M. Ringner, G. Hostetter, G. W. Maher, N. Goldberger, Y. Chen, J. Carpten, A. G. Elkahloun, and P. S. Meltzer, "Gastrointestinal stromal tumors with KIT mutations exhibit a remarkably homogeneous gene expression profile," *Cancer Res*, vol. 61, no. 24, pp. 8624–8628, 2001.

- [9] U. Alon, N. Barkai, D. Notterman, K. Gish, S. Ybarra, D. Mack, and A. Levine, "Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays," *Proc. Natl. Acad. Sci. USA*, vol. 96, no. 12, pp. 6745–6750, 1999.
- [10] O. Alter, P. O. Brown, and D. Botstein, "Singular value decomposition for genome-wide expression data processing and modeling," *PNAS*, vol. 97, no. 18, pp. 10 101–10 106, 2000.
- [11] C. Ambrose and G. J. McLachlan, "Selection bias in gene extraction on the basis of microarray gene-expression data," *Proc. Natl. Acad. Sci.*, vol. 99, no. 10, pp. 6562–6566, 2002.
- [12] S. A. Armstrong, J. E. Staunton, L. B. Silverman, R. Pieters, M. L. den Boer, M. D. Minden, S. E. Sallan, E. S. Lander, T. R. Golub, and S. J. Korsmeyer, "MLL translocations specify a distinct gene expression profile that distinguishes a unique leukemia," *Nat Genet*, vol. 30, no. 1, pp. 41–47, 2002.
- [13] P. Baldi and G. W. Hatfield, *DNA microarrays and gene expression: From experiments to data analysis and modeling*. Cambridge, New York: Cambridge University Press, 2002.
- [14] J. Barrett and E. Kawasaki, "Microarrays: the use of oligonucleotides and cDNA for the analysis of gene expression," *Drug Discov. Today*, vol. 8, pp. 134–141, 2003.
- [15] R. E. Bellman, *Adaptive Control Processes*. Princeton, NJ: Princeton University Press, 1961.
- [16] A. Ben-Dor, L. Bruhn, N. Friedman, I. Nachman, M. Schummer, and Z. Yakhini, "Tissue classification with gene expression profiles," in *RECOMB '00: Proceedings of the fourth annual international conference on Computational molecular biology*. New York, NY, USA: ACM Press, 2000, pp. 54–64.
- [17] A. L. Blum and P. Langley, "Selection of relevant features and examples in machine learning," *Artificial Intelligence*, vol. 97, no. 1-2, pp. 245–271, 1997.
- [18] T. Bø and I. Jonassen, "New feature subset selection procedures for classification of expression profiles," *Genome Biology*, vol. 3, no. 4, pp. research0017.1–research0017.11, 2002.

- [19] G. E. P. Box and G. M. Jenkins, *Time series analysis: forecasting and control*. Holden-Day, London, 1970.
- [20] U. Braga-Neto and E. R. Dougherty, "Bolstered error estimation," *Pattern Recognition*, vol. 37, no. 6, pp. 1267–1281, 2004.
- [21] —, "Is cross-validation valid for small-sample microarray classification?" *Bioinformatics*, vol. 20, no. 3, pp. 374–380, 2004.
- [22] A. Brazma and J. Vilo, "Gene expression data analysis," *Federation of European Biochemical Societies Letters*, vol. 480, pp. 17–24, 2000.
- [23] M. P. S. Brown, W. N. Grundy, D. Lin, N. Cristianini, C. W. Sugnet, T. S. Furey, J. Ares, Manuel, and D. Haussler, "Knowledge-based analysis of microarray gene expression data by using support vector machines," *PNAS*, vol. 97, no. 1, pp. 262–267, 2000.
- [24] M. Bădoiu and K. L. Clarkson, "Optimal core-sets for balls," in *DIMACS Workshop on Computational Geometry*, 2002.
- [25] C. J. C. Burges, "A tutorial on support vector machines for pattern recognition," *Data Min. Knowl. Discov.*, vol. 2, no. 2, pp. 121–167, 1998.
- [26] G. C. Cawley and N. L. Talbot, "Fast exact leave-one-out cross-validation of sparse least-squares support vector machines," *Neural Networks*, vol. 17, no. 10, pp. 1467–1475, 2004.
- [27] C.-C. Chang and C.-J. Lin, *LIBSVM: a library for support vector machines*, 2001, software available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm>.
- [28] O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee, "Choosing multiple parameters for support vector machines," *Machine Learning*, vol. 46, no. 1-3, pp. 131–159, 2002.
- [29] J.-H. Cho, D. Lee, J. H. Park, and I.-B. Lee, "Gene selection and classification from microarray data using kernel machine," *FEBS Letters*, vol. 571, pp. 93–98, 2004.
- [30] R. Collobert and S. Bengio, "SVM-Torch: Support vector machines for large-scale regression problems," *Journal of Machine Learning Research*, vol. 1, pp. 143–160, 2001.

- [31] K. Crammer and Y. Singer, "On the algorithmic implementation of multiclass kernel-based vector machines," *Journal of Machine Learning Research*, vol. 2, pp. 265–292, 2001.
- [32] N. Cristianini and J. Shawe-Taylor, *An introduction to support Vector Machines: and other kernel-based learning methods*. New York, NY, USA: Cambridge University Press, 2000.
- [33] M. Dash and H. Liu, "Feature selection for classification," *Intelligent Data Analysis*, vol. 1, pp. 131–156, 1997.
- [34] P. Devijver and J. Kittler, *Pattern Recognition: a Statistical Approach*. London: Prentice Hall, 1982.
- [35] C. H. Ding, "Unsupervised feature selection via two-way ordering in gene expression analysis," *Bioinformatics*, vol. 19, no. 10, pp. 1259–1266, 2003.
- [36] S. Dudoit, J. Fridlyand, and T. P. Speed, "Comparison of discrimination methods for the classification of tumors using gene expression data," *Journal of the American Statistical Association*, vol. 97, no. 457, pp. 77–87, 2002.
- [37] B. Efron and R. Tibshirani, "Improvements on cross-validation: The .632+ bootstrap method," *Journal of the American Statistical Association*, vol. 92, no. 438, pp. 548–560, 1997.
- [38] M. B. Eisen, P. T. Spellman, P. O. Brown, and D. Botstein, "Cluster analysis and display of genome-wide expression patterns," *PNAS*, vol. 95, no. 25, pp. 14863–14868, 1998.
- [39] T. Evgeniou, M. Pontil, and T. Poggio, "Regularization networks and support vector machines," *Advanced In Computational Mathematics*, vol. 13, no. 1, pp. 1–50, 2000.
- [40] G. Fort and S. Lambert-Lacroix, "Classification using partial least squares with penalized logistic regression," *Bioinformatics*, vol. 21, no. 7, pp. 1104–1111, 2005.
- [41] T. Furey, N. Cristianini, N. Duffy, D. Bednarski, M. Schummer, and D. Hausler, "Support vector machine classification and validation of cancer tissue samples using microarray expression data," *Bioinformatics*, vol. 16, no. 10, pp. 906–914, 2000.

- [42] M. E. Garber, O. G. Troyanskaya, K. Schluens, S. Petersen, Z. Thaesler, M. Pacyna-Gengelbach, M. van de Rijn, G. D. Rosen, C. M. Perou, R. I. Whyte, R. B. Altman, P. O. Brown, D. Botstein, and I. Petersen, "Diversity of gene expression in adenocarcinoma of the lung," *PNAS*, vol. 98, no. 24, pp. 13 784–13 789, 2001.
- [43] R. Gentleman, V. J. Carey, W. Huber, R. A. Irizarry, and S. Dudoit, *Bioinformatics and Computational Biology Solutions Using R and Bioconductor*, R. Gentleman, V. J. Carey, W. Huber, R. A. Irizarry, and S. Dudoit, Eds. Springer, 2005.
- [44] G. H. Golub and C. F. Van Loan, *Matrix computations (3rd ed.)*. Johns Hopkins University Press, 1996.
- [45] T. Golub, D. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. Mesirov, H. Coller, M. Loh, J. Downing, M. Caligiuri, C. Bloomfield, and E. Lander, "Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring," *Science*, vol. 286, pp. 531–537, 1999.
- [46] S. Gruvberger, M. Ringner, Y. Chen, S. Panavally, L. H. Saal, A. Borg, M. Ferno, C. Peterson, and P. S. Meltzer, "Estrogen receptor status in breast cancer is associated with remarkably distinct gene expression patterns," *Cancer Res*, vol. 61, no. 16, pp. 5979–5984, 2001.
- [47] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik., "Gene selection for cancer classification using support vector machines," *Machine Learning*, vol. 46, pp. 389–422, 2002.
- [48] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *Journal of Machine learning Research*, vol. 3, pp. 1157–1182, 2003.
- [49] D. A. Harville, *Matrix Algebra From a Statistician's Perspective*. Springer, September 1997.
- [50] T. Hastie, R. Tibshirani, and J. H. Friedman, *The Elements of Statistical Learning*. Springer, August 2001.
- [51] I. Hedenfalk, D. Duggan, Y. Chen, M. Radmacher, M. Bittner, R. Simon, P. Meltzer, B. Gusterson, M. Esteller, M. Raffeld, Z. Yakhini, A. Ben-Dor, E. Dougherty, J. Kononen, L. Bubendorf, W. Fehrle, S. Pittaluga, S. Gruvberger, N. Loman, O. Johannsson, H. Olsson, B. Wilfond, G. Sauter, O.-P.

- Kallioniemi, A. Borg, and J. Trent, "Gene-expression profiles in hereditary breast cancer," *N Engl J Med*, vol. 344, no. 8, pp. 539–548, 2001.
- [52] R. A. Heller, M. Schena, A. Chai, D. Shalon, T. Bedilion, J. Gilmore, D. E. Woolley, and R. W. Davis, "Discovery and analysis of inflammatory disease-related genes using cDNA microarrays," *PNAS*, vol. 94, no. 6, pp. 2150–2155, 1997.
- [53] R. Herwig, A. J. Poustka, C. Muller, C. Bull, H. Lehrach, and J. O'Brien, "Large-scale clustering of cDNA-fingerprinting data," *Genome Res.*, vol. 9, no. 11, pp. 1093–1105, 1999.
- [54] I. Inza, B. Sierra, R. Blanco, and P. Larranaga, "Gene selection by sequential search wrapper approaches in microarray cancer class prediction," *Journal of Intelligent and Fuzzy System*, vol. 12, no. 1, pp. 25–33, 2002.
- [55] R. A. Irizarry, B. M. Bolstad, F. Collin, L. M. Cope, B. Hobbs, and T. P. Speed, "Summaries of affymetrix genechip probe level data," *Nucl. Acids Res.*, vol. 31, no. 4, p. e15, 2003.
- [56] T. Joachims, "Making large-scale support vector machine learning practical," in *Advances in Kernel Methods: Support Vector Machines*, B. Schölkopf, C. Burges, and A. Smola, Eds. MIT Press, Cambridge, MA, 1999, pp. 169–184.
- [57] G. H. John, R. Kohavi, and K. Pfleger, "Irrelevant features and the subset selection problem," in *International Conference on Machine Learning*. AAAI Press, 1994, pp. 121–129.
- [58] M. K. Kerr, M. Martin, and G. A. Churchill, "Analysis of variance for gene expression microarray data," *Journal of Computational Biology*, vol. 7, no. 6, pp. 819–837, 2000.
- [59] J. Khan, R. Simon, M. Bittner, Y. Chen, S. Leighton, T. Pohida, P. Smith, Y. Jiang, G. Gooden, J. Trent, and P. Meltzer, "Gene expression profiling of alveolar rhabdomyosarcoma with cDNA microarrays," *Cancer Res*, vol. 58, no. 22, pp. 5009–5013, 1998.
- [60] J. Khan, J. S. Wei, M. Ringner, L. H. Saal, M. Ladanyi, F. Westermann, F. Berthold, M. Schwab, C. R. Antonescu, C. Peterson, and P. S. Meltzer,

- "Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks," *Nat Med*, vol. 7, no. 6, pp. 673–679, 2001.
- [61] C. Kihara, T. Tsunoda, T. Tanaka, H. Yamana, Y. Furukawa, K. Ono, O. Kitahara, H. Zembutsu, R. Yanagawa, K. Hirata, T. Takagi, and Y. Nakamura, "Prediction of sensitivity of esophageal tumors to adjuvant chemotherapy by cDNA microarray analysis of gene-expression profiles," *Cancer Res*, vol. 61, no. 17, pp. 6474–6479, 2001.
- [62] R. Kohavi and G. H. John, "Wrappers for feature subset selection," *Artificial Intelligence*, vol. 97, no. 1-2, pp. 273–324, 1997.
- [63] D. Koller and M. Sahami, "Toward optimal feature selection," in *International Conference on Machine Learning*, 1996, pp. 284–292.
- [64] I. Kononenko, "Estimating attributes: Analysis and extensions of RELIEF." in *ECML*, ser. Lecture Notes in Computer Science, F. Bergadano and L. D. Raedt, Eds., vol. 784. Springer, 1994, pp. 171–182.
- [65] U. H.-G. Kreißel, "Pairwise classification and support vector machines," pp. 255–268, 1999.
- [66] Y. LeCun, J. Denker, S. Solla, R. E. Howard, and L. D. Jackel, "Optimal brain damage," in *Advances in Neural Information Processing Systems II*, D. S. Touretzky, Ed. San Mateo, CA: Morgan Kauffman, 1990.
- [67] Y. Lee and C.-K. Lee, "Classification of multiple cancer types by multicategory support vector machines using gene expression data," *Bioinformatics*, vol. 19, no. 9, pp. 1132–1139, 2003.
- [68] B. Lewin, *Gene VII*. New York: Oxford University Press, 2000.
- [69] L. Li, C. R. Weinberg, T. A. Darden, and L. G. Pedersen, "Gene selection for sample classification based on gene expression data: study of sensitivity to choice of parameters of the GA/KNN method," *Bioinformatics*, vol. 17, no. 12, pp. 1131–1142, 2001.
- [70] W. Li and Y. Yang, "How many genes are needed for a discriminant microarray data analysis?" in *Methods of Microarray Data Analysis: Papers from CAMDA '00*, S. Lin and K. Johnson, Eds. Kluwer Academic, 2002, pp. 137–150.

- [71] Y. Lin, "Support vector machines and the bayes rule in classification," *Data Mining and Knowledge Discovery*, vol. 6, pp. 259–275, 2002.
- [72] D. Lockhart and E. Winzeler, "Genomics, gene expression and dna arrays," *Nature Insight*, vol. 405, pp. 827–836, 2000.
- [73] A. Luntz and V. Brailovsky, "On estimation of characters obtained in statistical procedure of recognition (in russian)," *Technicheskaya Kibernetika*, vol. 3, 1969.
- [74] J. Luo, D. J. Duggan, Y. Chen, J. Sauvageot, C. M. Ewing, M. L. Bittner, J. M. Trent, and W. B. Isaacs, "Human prostate cancer and benign prostatic hyperplasia: Molecular dissection by gene expression profiling," *Cancer Res*, vol. 61, no. 12, pp. 4683–4688, 2001.
- [75] L. Ma, J. Li, L. Qu, J. Hager, Z. Chen, H. Zhao, and X. W. Deng, "Light control of arabidopsis development entails coordinated regulation of genome expression and cellular pathways," *Plant Cell*, vol. 13, no. 12, pp. 2589–2607, 2001.
- [76] T. J. MacDonald, K. M. Brown, B. LaFleur, K. Peterson, C. Lawlor, Y. Chen, R. J. Packer, P. Cogen, and D. A. Stephan, "Expression profiling of medulloblastoma: PDGFRA and the RAS/MAPK pathway as therapeutic targets for metastatic disease," *Nat Genet*, vol. 29, no. 2, pp. 143–152, 2001.
- [77] K. Mao, "Feature subset selection for support vector machines through discriminative function pruning analysis," *IEEE Transactions on systems man and cybernetics Part B-Cybernetics*, vol. 34, no. 1, pp. 60–67, 2004.
- [78] E. Marshall, "Do-it-yourself gene watching," *Science*, vol. 286, no. 5439, pp. 444–447, 1999.
- [79] S. Mukherjee, P. Tamayo, D. Slonim, A. Verri, T. Golub, J. Mesirov, and T. Poggio, "Support vector machine classification of microarray data," *A.I. Memo 1677, MIT Artificial Intelligence Laboratory*, 1998.
- [80] D. Newman, S. Hettich, C. Blake, and C. Merz, "UCI repository of machine learning databases," 1998. [Online]. Available: <http://www.ics.uci.edu/~mlearn/MLRepository.html>
- [81] D. V. Nguyen and D. M. Rocke, "Multi-class cancer classification via partial least squares with gene expression profiles," *Bioinformatics*, vol. 18, no. 9, pp. 1216–1226, 2002.

- [82] —, “Tumor classification by partial least squares using microarray gene expression data,” *Bioinformatics*, vol. 18, no. 1, pp. 39–50, 2002.
- [83] D. A. Notterman, U. Alon, A. J. Sierk, and A. J. Levine, “Transcriptional gene expression profiles of colorectal adenoma, adenocarcinoma, and normal tissue examined by oligonucleotide arrays,” *Cancer Res*, vol. 61, no. 7, pp. 3124–3130, 2001.
- [84] C. H. Ooi and P. Tan, “Genetic algorithms applied to multi-class prediction for the analysis of gene expression data,” *Bioinformatics*, vol. 19, no. 1, pp. 37–44, 2003.
- [85] C. M. Perou, T. Sorlie, M. B. Eisen, M. van de Rijn, S. S. Jeffrey, C. A. Rees, J. R. Pollack, D. T. Ross, H. Johnsen, L. A. Akslen, O. Fluge, A. Pergamenschikov, C. Williams, S. X. Zhu, P. E. Lonning, A.-L. Borresen-Dale, P. O. Brown, and D. Botstein, “Molecular portraits of human breast tumours,” *Nature*, vol. 406, no. 6797, pp. 747–752, 2000.
- [86] J. C. Platt, “Fast training of support vector machines using sequential minimal optimization,” in *Advances in Kernel Methods: Support Vector Machines*, B. Schölkopf, C. Burges, and A. Smola, Eds. MIT Press, Cambridge, MA, 1999, pp. 185–208.
- [87] S. L. Pomeroy, P. Tamayo, M. Gaasenbeek, L. M. Sturla, M. Angelo, M. E. McLaughlin, J. Y. H. Kim, L. C. Goumnerova, P. M. Black, C. Lau, J. C. Allen, D. Zagzag, J. M. Olson, T. Curran, C. Wetmore, J. A. Biegel, T. Poggio, S. Mukherjee, R. Rifkin, A. Califano, G. Stolovitzky, D. N. Louis, J. P. Mesirov, E. S. Lander, and T. R. Golub, “Prediction of central nervous system embryonal tumour outcome based on gene expression,” *Nature*, vol. 415, no. 6870, pp. 436–442, 2002.
- [88] P. Pudil, J. Novovicova, and J. Kittler, “Floating search methods in feature selection,” *Pattern Recogn. Lett.*, vol. 15, no. 11, pp. 1119–1125, 1994.
- [89] J. Quackenbush, “Computational analysis of microarray data,” *Nat Rev Genet*, vol. 2, no. 6, pp. 418–427, 2001.
- [90] A. Rakotomamonjy, “Variable selection using svm based criteria,” *J. Mach. Learn. Res.*, vol. 3, pp. 1357–1370, 2003.
- [91] S. Ramaswamy, P. Tamayo, R. Rifkin, S. Mukherjee, C.-H. Yeang, M. Angelo, C. Ladd, M. Reich, E. Latulippe, J. P. Mesirov, T. Poggio, W. Gerald,

- M. Loda, E. S. Lander, and T. R. Golub, "Multiclass cancer diagnosis using tumor gene expression signatures," *PNAS*, vol. 98, no. 26, pp. 15 149–15 154, 2001.
- [92] M. Ringnér, C. Peterson, and J. Khan, "Analyzing array data using supervised methods," *Pharmacogenomics*, vol. 3, no. 3, pp. 403–415, May 2002.
- [93] D. T. Ross, U. Scherf, M. B. Eisen, C. M. Perou, C. Rees, P. Spellman, V. Iyer, S. S. Jeffrey, M. V. de Rijn, M. Waltham, A. Pergamenschikov, J. C. Lee, D. Lashkari, D. Shalon, T. G. Myers, J. N. Weinstein, D. Botstein, and P. O. Brown, "Systematic variation in gene expression patterns in human cancer cell," *Nat Genet*, vol. 24, no. 3, pp. 227–235, 2000.
- [94] M. Schena, D. Shalon, R. W. Davis, and P. O. Brown, "Quantitative monitoring of gene expression patterns with a complementary DNA microarray," *Science*, vol. 270, no. 5235, pp. 467–470, October 1995.
- [95] B. Scholkopf, R. Herbrich, and A. J. Smola, "A generalized representer theorem," in *Proceedings of the 14th Annual Conference on Computational Learning Theory and and 5th European Conference on Computational Learning Theory*. Springer-Verlag, 2001, pp. 416–426.
- [96] M. A. Shipp, K. N. Ross, P. Tamayo, A. P. Weng, J. L. Kutok, R. C. Aguiar, M. Gaasenbeek, M. Angelo, M. Reich, G. S. Pinkus, T. S. Ray, M. A. Koval, K. W. Last, A. Norton, T. A. Lister, J. Mesirov, D. S. Neuberg, E. S. Lander, J. C. Aster, and T. R. Golub, "Diffuse large B-cell lymphoma outcome prediction by gene-expression profiling and supervised machine learning," *Nat Med*, vol. 8, no. 1, pp. 68–74, 2002.
- [97] C. Sima, S. Attoor, U. Braga-Neto, J. Lowey, E. Suh, and E. R. Dougherty, "Impact of error estimation on feature selection," *Pattern Recognition*, vol. 38, no. 12, pp. 2472–2482, 2005.
- [98] C. Sima, U. Braga-Neto, and E. R. Dougherty, "Superior feature-set ranking for small samples using bolstered error estimation," *Bioinformatics*, vol. 21, no. 7, pp. 1046–1054, 2005.
- [99] D. K. Slonim, P. Tamayo, J. P. Mesirov, T. R. Golub, and E. S. Lander, "Class prediction and discovery using gene expression data," in *RECOMB '00: Proceedings of the fourth annual international conference on Computational molecular biology*. New York, NY, USA: ACM Press, 2000, pp. 263–272.

- [100] A. I. Su, J. B. Welsh, L. M. Sapinoso, S. G. Kern, P. Dimitrov, H. Lapp, P. G. Schultz, S. M. Powell, C. A. Moskaluk, J. Frierson, Henry F., and G. M. Hampton, "Molecular classification of human carcinomas by use of gene expression signatures," *Cancer Res*, vol. 61, no. 20, pp. 7388–7393, 2001.
- [101] J. A. K. Suykens and J. Vandewalle, "Least squares support vector machine classifiers," *Neural Processing Letters*, vol. 9, no. 3, pp. 293–300, 1999.
- [102] P. Tamayo, D. Slonim, J. Mesirov, Q. Zhu, S. Kitareewan, E. Dmitrovsky, E. S. Lander, and T. R. Golub, "Interpreting patterns of gene expression with self-organizing maps: Methods and application to hematopoietic differentiation," *PNAS*, vol. 96, no. 6, pp. 2907–2912, 1999.
- [103] Y. Tan, L. Shi, W. Tong, and C. Wang, "Multi-class cancer classification by total principal component regression (TPCR) using microarray gene expression data," *Nucl. Acids Res.*, vol. 33, no. 1, pp. 56–65, 2005.
- [104] E. K. Tang, P. Suganthan, and X. Yao, "Gene selection algorithms for microarray data based on least squares support vector machine," *BMC Bioinformatics*, vol. 7, no. 1, p. 95, 2006.
- [105] S. Tavazoie, J. D. Hughes, M. J. Campbell, R. J. Cho, and G. M. Church, "Systematic determination of genetic network architecture," *Nat Genet*, vol. 22, no. 3, pp. 281–285, 1999.
- [106] S. Theodoridis and K. Koutroumbas, *Pattern Recognition (2nd edition)*. San Diego, CA: Academic Press, 2003.
- [107] A. N. Tikhonov and V. Y. Arsenin, *Solutions of Ill-posed problems*. Washington D.C.: W. H. Winston, 1977.
- [108] G. J. Tortora, B. R. Funke, and C. L. Case, *Microbiology: An Introduction*. Benjamin Cummings, May 2003.
- [109] L. J. van 't Veer, H. Dai, M. J. van de Vijver, Y. D. He, A. A. M. Hart, M. Mao, H. L. Peterse, K. van der Kooy, M. J. Marton, A. T. Witteveen, G. J. Schreiber, R. M. Kerkhoven, C. Roberts, P. S. Linsley, R. Bernards, and S. H. Friend, "Gene expression profiling predicts clinical outcome of breast cancer," *Nature*, vol. 415, no. 6871, pp. 530–536, 2002.
- [110] R. J. Vanderbei, "LOQO: An interior point code for quadratic programming," *Optimization Methods and Software*, vol. 11, pp. 451–484, 1999.

- [111] V. Vapnik and O. Chapelle, "Bounds on error expectation for support vector machines," *Neural Computation*, vol. 12, no. 9, pp. 2013–2036, 2000.
- [112] V. N. Vapnik, *Statistical Learning Theory*. New York: John Wiley and Sons, 1998.
- [113] A. R. Webb, *Statistical Pattern Recognition, 2nd Edition*. John Wiley & Sons, October 2002.
- [114] M. West, C. Blanchette, H. Dressman, E. Huang, S. Ishida, R. Spang, H. Zuzan, J. Olson, John A., J. R. Marks, and J. R. Nevins, "Predicting the clinical status of human breast cancer by using gene expression profiles," *Proc. Natl. Acad. Sci. USA*, vol. 98, no. 20, pp. 11 462–11 467, 2001.
- [115] J. Weston and C. Watkins, "Support vector machines for multiclass pattern recognition," in *Proceedings of the Seventh European Symposium On Artificial Neural Networks*, 4 1999.
- [116] J. Weston, S. Mukherjee, O. Chapelle, M. Pontil, T. Poggio, and V. Vapnik, "Feature selection for svms," in *Advances in Neural Information Processing System*, 2001, pp. 668–674.
- [117] M. Xiong, W. Li, J. Zhao, L. Jin, and E. Boerwinkle, "Feature (gene) selection in gene expression-based tumor classification," *Molecular Genetics and Metabolism*, vol. 73, pp. 239–247, 2001.
- [118] M. Xiong, X. Fang, and J. Zhao, "Biomarker identification by feature wrappers," *Genome Research*, vol. 11, no. 11, pp. 1878–1887, 2001.
- [119] H. Zembutsu, Y. Ohnishi, T. Tsunoda, Y. Furukawa, T. Katagiri, Y. Ueyama, N. Tamaoki, T. Nomura, O. Kitahara, R. Yanagawa, K. Hirata, and Y. Nakamura, "Genome-wide cDNA microarray screening to correlate gene expression profiles with sensitivity of 85 human cancer xenografts to anticancer drugs," *Cancer Res*, vol. 62, no. 2, pp. 518–527, 2002.
- [120] T. Zhang, "Leave-one-out bounds for kernel methods," *Neural Computation*, vol. 15, no. 6, pp. 1397–1437, 2003.
- [121] X. Zhang and W. Wong, "Recursive sample classification and gene selection based on svm: method and software description," Department of Biostatistics, Harvard School of Public Health, Tech. Rep., 2001.

Appendix A.

Lagrangian Optimization Theory

The optimization problem that arises from Support Vector Machines (SVMs) is closely related to the Lagrangian Optimization Theory. The introduction in this Appendix, which is excerpted from Chapter 5 in [32], provides a deeper understanding of the mechanism of SVMs.

The purpose of Lagrangian theory is to characterize the solution of an optimization problem with equality and/or inequality constraints. The general optimization problem can be stated as follows:

Definition A..1 (Primal optimization problem). *Given functions $f, g_i, i = 1, \dots, k$, and $h_i, i = 1, \dots, m$, defined on a domain $\Omega \subseteq \mathbb{R}^n$,*

$$\begin{aligned} & \text{minimize} && f(\mathbf{w}), && \mathbf{w} \in \Omega, \\ & \text{subject to} && g_i(\mathbf{w}) \leq 0, && i = 1, \dots, k, \\ & && h_i(\mathbf{w}) = 0, && i = 1, \dots, m, \end{aligned}$$

where $f(\mathbf{w})$ is called the objective function, and the remaining relations are called, respectively, the inequality and equality constraints. The optimal value of the objective function is called the value of the optimization problem.

To simplify the notation, we denote $\mathbf{g}(\mathbf{w}) \leq \mathbf{0}$ to indicate $g_i(\mathbf{w}) \leq 0, i = 1, \dots, k$. $\mathbf{h}(\mathbf{w}) = \mathbf{0}$ has a similar meaning for the equality constraints.

The region of the domain, in which the objective function is defined and all the constraints are satisfied, is called the feasible region, and it is defined as,

$$R = \{\mathbf{w} \in \Omega | \mathbf{g}(\mathbf{w}) \leq \mathbf{0}, \mathbf{h}(\mathbf{w}) = \mathbf{0}\}$$

A solution of the optimization problem is a point $\mathbf{w}^* \in R$ such that there does not exist other point $\mathbf{w} \in R$ for which $f(\mathbf{w}) < f(\mathbf{w}^*)$.

Other concepts with respect to the training of SVMs are the convex function and convex set.

Definition A..2 (Convex function). *A real-valued function $f(\mathbf{w})$ is called convex for $\mathbf{w} \in \mathbb{R}^n$ if, $\forall \mathbf{w}, \mathbf{u} \in \mathbb{R}^n$, and for any $\theta \in (0, 1)$,*

$$f(\theta \mathbf{w} + (1 - \theta) \mathbf{u}) \leq \theta f(\mathbf{w}) + (1 - \theta) f(\mathbf{u}).$$

Definition A..3 (Convex set). *A set $\Omega \subseteq \mathbb{R}^n$ is called convex if, $\forall \mathbf{w}, \mathbf{u} \in \Omega$, and for any $\theta \in (0, 1)$, the point $[\theta \mathbf{w} + (1 - \theta) \mathbf{u}] \in \Omega$.*

Because the optimization problem introduced from the training of SVMs is convex quadratic, that is, the objective function is convex and quadratic while the constraints are all linear, the Lagrangian theory presented at the following is restricted to the case of convex quadratic optimization problem.

In constrained problems, a function, known as the Lagrangian, is also defined as the objective function plus a linear combination of the constraints to detect solutions.

Definition A..4 (Lagrangian function). *Given an optimization problem of Defini-*

tion A..1, the Lagrangian function is defined as

$$\begin{aligned}\mathcal{L}(\mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta}) &= f(\mathbf{w}) + \sum_{i=1}^k \alpha_i g_i(\mathbf{w}) + \sum_{i=1}^m \beta_i h_i(\mathbf{w}) \\ &= f(\mathbf{w}) + \boldsymbol{\alpha}^T \mathbf{g}(\mathbf{w}) + \boldsymbol{\beta}^T \mathbf{h}(\mathbf{w}).\end{aligned}$$

where the coefficients α_i, β_i are called the Lagrange multipliers.

We can now define the Lagrangian dual problem.

Definition A..5 (Lagrangian dual problem). *The Lagrangian dual problem of the primal problem of Definition A..1 is the following problem:*

$$\begin{aligned}\text{maximize } \mathcal{L}_D(\boldsymbol{\alpha}, \boldsymbol{\beta}) &= \inf_{\mathbf{w} \in \Omega} \mathcal{L}(\mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta}), \\ \text{subject to } \boldsymbol{\alpha} &\geq 0,\end{aligned}$$

where $\inf(\mathcal{S})$ is denoted as infimum (greatest lower bound) of a set $\mathcal{S}$ of real numbers. The value of the objective function at the optimal solution is called the value of the problem.

The relationship between the primal and dual problems is exhibited from the following weak duality theorem.

Theorem A..1 (Weak duality theorem). *Let $\mathbf{w} \in \Omega$ be a feasible solution of the primal problem of Definition A..1 and $(\boldsymbol{\alpha}, \boldsymbol{\beta})$ be a feasible solution of the dual problem of Definition A..5. Then $f(\mathbf{w}) \leq \mathcal{L}_D(\boldsymbol{\alpha}, \boldsymbol{\beta})$*

The value of the dual problem is upper bounded by the value of the primal problem. The difference between the values of the two problems is known as the duality gap. If the duality gap is zero, the primal problem can be converted to solve the dual problem. However the absence of the duality gap is not in general guaranteed. But

in some cases, the dual and primal problems have the same value for the optimization problems.

Theorem A..2 (Strong duality theorem). *Given an optimization problem with convex domain $\Omega \subseteq \mathbb{R}^n$,*

$$\begin{aligned} & \text{minimize} && f(\mathbf{w}), && \mathbf{w} \in \Omega, \\ & \text{subject to} && g_i(\mathbf{w}) \leq 0, && i = 1, \dots, k, \\ & && h_i(\mathbf{w}) = 0, && i = 1, \dots, m, \end{aligned}$$

where the g_i and h_i are affine functions, the duality gap is zero.

An affine function is defined as the form, $\mathbf{f}(\mathbf{x}) = \mathbf{A}\mathbf{x} + \mathbf{b}$, for some matrix $\mathbf{A}$ and vector $\mathbf{b}$. Note that affine functions are convex.

The Kuhn-Tucker theorem is introduced to give conditions for an optimum solution to a general optimization problem.

Theorem A..3 (Kuhn-Tucker theorem). *Given an optimization problem with convex domain $\Omega \subseteq \mathbb{R}^n$,*

$$\begin{aligned} & \text{minimize} && f(\mathbf{w}), && \mathbf{w} \in \Omega, \\ & \text{subject to} && g_i(\mathbf{w}) \leq 0, && i = 1, \dots, k, \\ & && h_i(\mathbf{w}) = 0, && i = 1, \dots, m, \end{aligned}$$

with $f \in C^1$ convex and g_i, h_i affine, necessary and sufficient conditions for a normal

point $\mathbf{w}^*$ to be an optimum are the existence of α^*, β^* such that

$$\frac{\partial \mathcal{L}(\mathbf{w}^*, \alpha^*, \beta^*)}{\partial \mathbf{w}} = 0, \quad (\text{A.-14})$$

$$\frac{\partial \mathcal{L}(\mathbf{w}^*, \alpha^*, \beta^*)}{\partial \beta} = 0, \quad (\text{A.-13})$$

$$\alpha_i^* g_i(\mathbf{w}^*) = 0, \quad i = 1, \dots, k, \quad (\text{A.-12})$$

$$g_i(\mathbf{w}^*) \leq 0, \quad i = 1, \dots, k, \quad (\text{A.-11})$$

$$\alpha_i^* \geq 0, \quad i = 1, \dots, k. \quad (\text{A.-10})$$

The third relation, Eq. (A.-12), is known as Karush-Kuhn-Tucker complementarity condition. It implies that for the constraints with $g_i(\mathbf{w}^*) < 0$ the corresponding $\alpha_i^* = 0$, whereas for constraints with $g_i(\mathbf{w}^*) = 0$ the $\alpha_i^* \geq 0$.