EFFECTIVE USE OF DATA MINING TECHNOLOGIES ON BIOLOGICAL AND CLINICAL DATA

LIU HUIQING

National University of Singapore

2004

EFFECTIVE USE OF DATA MINING TECHNOLOGIES ON BIOLOGICAL AND CLINICAL DATA

LIU HUIQING

(M.Science, National University of Singapore, Singapore)

(M.Engineering, Xidian University, PRC)

(B.Economics, Huazhong University of Science and Technology, PRC)

A THESIS SUBMITTED

FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

INSTITUTE FOR INFOCOMM RESEARCH

NATIONAL UNIVERSITY OF SINGAPORE

2004

In memory of my mother, and
to my father

Acknowledgements

First and foremost I would like to acknowledge my supervisor, Professor Wong Limsoon, for his patient, tireless and prompt help. Limsoon always provides me complete freedom to explore and work on the research topics that I have interests in. Although it was difficult for me to make quick progress at the beginning, I must appreciate the wisdom of his supervision when I started to think for myself and become relatively independent. On the other hand, he never delays to answer my questions. It is my luck to study and work under his guidance. I also thank my Ph.D. advisory committee members, Dr Li Jinyan and Dr Wynne Hsu, for many valuable discussions.

During the past three years, my colleagues in the Knowledge Discovery Department of the Institute for Infocomm Research (I^2R) have provided me much appreciated help in my daily work. I would like to thank all of them for their generous assistance, valuable suggestions and friendship. Special acknowledgements go to Mr Han Hao for his collaborations on dealing with problems from sequence data, and my department head, Dr Brusic Vladimir, for his encouragements on my study.

I thank staff in the Graduate Office, School of Computing, National University of Singapore, and graduate studies support staff in the Human Resource Department of I^2R. They always gave me very quick responses when I encountered problems during my past four years of study.

I can not finish my thesis work without the strong support from my family. In the middle of my study, I lost my dearest mother, who once was my closest person in this world and died from lung cancer in year 2002. In order to let me concentrate on my study, she tried her best to take care of the whole family though she was very weak herself. Even during her last days in this world, she still cared about my research progress. I owed her too much. Besides my mother, I have a great father as well. He has provided and is still providing his unconditional support and encouragement on my research work. Without his love and help, I might have given up the study when my mother passed away. Special thanks must go to my two lovely daughters — Yugege and Yungege — who are my angels and the source of my happiness. Together with them is my hubby, Hongming, who is always there to provide me his support through both the highs and lows of my time.

Contents

List of Tables

List of Figures

Summary

With more and more biological information generated, the most pressing task of bioinformatics has become to analyse and interpret various types of data, including nucleotide and amino acid sequences, protein structures, gene expression profilings and so on. In this thesis, we apply the data mining techniques of feature generation, feature selection, and feature integration with learning algorithms to tackle the problems of disease phenotype classification and patient survival prediction from gene expression profiles, and the problems of functional site prediction from DNA sequences.

When dealing with problems arising from gene expression profiles, we propose a new feature selection process for identifying genes associated with disease phenotype classification or patient survival prediction. This method, *ERCOF* (Entropy-based Rank sum test and COrrelation Filtering), aims to select a set of sharply discriminating genes with little redundancy by combining entropy measure, Wilcoxon rank sum test and Pearson correlation coefficient test. As for classification algorithms, we focus on methods built on the idea of ensemble of decision trees, including widely used bagging, boosting and random forests, as well as newly published CS4. To compare the decision tree methods with other state-of-the-art classifiers, support vector machines (SVM) and k-nearest neighbour are also used. Various comparisons among different feature selection methods and different classification algorithms are addressed based on more than one thousand tests conducted on six gene expression profiles and one proteomic data.

In the study of patient survival prediction, we present a new idea of selecting informative training samples by defining long-term and short-term survivors. ERCOF is then applied to identify genes from these samples. A regression function built on the selected samples and genes by a linear kernel SVM is worked out to assign a risk score to each patient. Kaplan-Meier plots for different risk groups formed on the risk scores are then drawn to show the effectiveness of the model. Two case studies, one on survival prediction for patients after chemotherapy for diffuse large-B-cell lymphoma and one on lung adenocarcinomas, are conducted.

In order to apply data mining methodology to identify functional sites in biological sequences, we first generate candidate features using k-gram nucleotide acid or amino acid patterns and then transform original sequences respect to the new constructed feature space. Feature selection is then conducted to find signal patterns that can distinguish true functional sites from those false ones. These selected features are further integrated with learning algorithms to build classification and prediction models. Our idea is used to recognize translation initiation sites and polyadenylation signals in DNA and mRNA sequences. For each application, experimental results across different data sets (including both public ones and our own extracted ones) are collected to demonstrate the effectiveness and robustness of our method.

Chapter 1

Introduction

The past few decades witness an explosive growth in biological information generated by the scientific community. This is caused by major advances in the field of molecular biology, coupled with advances in genomic technologies. In turn, the huge amount of genomic data generated not only leads to a demand on the computer science community to help store, organize and index the data, but also leads to a demand for specialized tools to view and analyze the data.

"Biology in the 21st century is being transformed from a purely lab-based science to an information science as well" [3].

As a result of this transformation, a new field of science was born, in which biology, computer science, and information technology merge to form a single discipline [3]. This is *bioinformatics*.

"The ultimate goal of bioinformatics is to enable the discovery of new biological insights as well as to create a global perspective from which unifying principles in biology can be discerned" [3].

1.1 Motivation

At the beginning, the main role of bioinformatics was to create and maintain databases to store biological information, such as nucleotide and amino acid sequences. With more and more data generated, nowadays, the most pressing task of bioinformatics has moved to analyse and interpret various types of data, including nucleotide and amino acid sequences, protein domains, protein structures and so on. To meet the new requirements arising from the new tasks, researchers in the field of bioinformatics are working on the development of new algorithms (mathematical formulas, statistical methods and etc) and software tools which are designed for assessing relationships among large data sets stored, such as methods to locate a gene within a sequence, predict protein structure and/or function, understand diseases at gene expression level and etc.

Motivated by the fast development of bioinformatics, this thesis is designed to apply data mining technologies to some biological data so that the relevant biological problems can be solved by computer programs. The aim of data mining is to automatically or semi-automatically discover hidden knowledge, unexpected patterns and new rules from data. There are a variety of technologies involved in the process of data mining, such as statistical analysis, modeling techniques and database technology. During the last ten years, data mining is undergoing very fast development both on techniques and applications. Its typical applications include market segmentation, customer profiling, fraud detection, (electricity) loading forecasting, credit risk analysis and so on. In the current post-genome age, understanding floods of data in molecular biology brings great opportunities and big challenges to data mining researchers. Successful stories from this new application will greatly benefit both computer science and biology communities. We would like to call this *discovering biological knowledge "in silico" by data mining*.

1.2 Work and Contribution

To make use of original biological and clinical data in the data mining process, we follow the regular process flow in data mining but with emphasis on three steps of feature manipulation, viz. feature space generation, feature selection and feature integration with learning algorithms. These steps are important in dealing with biological and clinical data.

(1) Some biological data, such as DNA sequences, have no explicit features that can be easily used by learning algorithms. Thus, constructing a feature space to describe original data becomes necessary.

(2) Quite a number of biological and clinical data sets possess many features. Selecting signal features and removing noisy ones will not only largely reduce the processing time and greatly improve the learning performance in the later stage, but also help locate good pat-

terns that are related to the essence of the study. For example, in gene expression data analysis, feature selection methods have been widely used to find genes that are most associated with a disease or a subtype of certain cancer.

(3) Many issues arising from biological and clinical data, in the final analysis, can be treated as or converted into classification problems and then can be solved by data mining algorithms.

In this thesis, we will mainly tackle gene expression profiles and DNA sequence data.

For gene expression profiles, we apply our method to solve two kinds of problems: phenotype classification and patient survival prediction. In these two problems, genes serve as features. Since profile data often contains thousands of genes, we put forward a new feature selection method ERCOF to identify genes most related to the problem. ERCOF conducts three-phase of gene filtering. First, it selects genes using an entropy-based discretization algorithm, which generally keeps only 10% of discriminating genes. Secondly, these remaining genes are further filtered by Wilcoxon rank sum test, a non-parametric statistic alternative to the t-test. Genes passing this round of filtering are automatically divided into two groups: one group consists of genes that are highly expressed in one type of samples (such as *cancer*) while another group consists of genes that are highly expressed in another type of samples (such as *non-cancer*). In the third phase, correlated genes in each group are determined by Pearson correlation coefficient test and only some representatives of them are kept to form the final set of selected genes.

When applying learning algorithms to classify phenotypes, we focus on classifiers built on the idea of an ensemble of decision trees, including the newly published CS4 [63, 62], as well as state-of-the-art Bagging [19], Boosting [38], and Random forests [20]. More than one thousand tests are conducted on six published gene expression profiling data sets and one proteomic data set. To compare the performance of these ensembles of decision tree methods with those widely used learning algorithms in gene expression studies, experimental results on support vector machines (SVM) and k-nearest neighbour (k-NN) are also collected. SVM is chosen because it is a representative of kernel function. k-NN is chosen because it is the most typical instance-based classifier. To demonstrate the main advantage of the decision tree methods, we present some of decision trees induced from data sets. These trees are simple, explicit and easy to understand. For each classifier, besides ERCOF, we also try features selected by several other entropy-based filtering methods. Therefore, various comparisons of learning algorithms and feature selection methods can be addressed.

In the study of using gene expression profiles to predict patient survival status, we present a new idea of selecting informative training samples by defining "long-term" and "short-term" survivors. After identifying genes associated with survival via ERCOF, a scoring model built on SVM is worked out to assign risk score to each patient. Kaplan-Meier plots for different risk groups formed on the risk scores are then drawn to show the effectiveness of the model.

Another biological domain to which the proposed 3-step feature manipulation method is applied is the recognition of functional sites in DNA sequences, such as translation initiation sites (TIS) and polyadenylation (poly(A)) signal. In this study, we put our emphasis on feature generation — k-gram nucleotide acid or amino acid patterns are used to construct the feature space and the frequency of each pattern appearing in the sequence is used as value. Under the description of the new features, original sequence data are then transformed to frequency vector data to which feature selection and classification can be applied. In TIS recognition, we test our methods on three independent data sets. Besides the cross validation within each dat set, we also conduct the tests across different data sets. In the identification of poly(A) signal, we make use of both public and our own collected data and build different models for DNA and mRNA sequences. In both studies, we achieve comparable or better prediction accuracy than those reported in the literature on the same data sets. In addition, we also verify some known motifs and find some new patterns related to the identification of relevant functional sites.

The main contributions of this thesis are

(1) articulating a 3-step feature manipulation method to solve some biological problems;

(2) putting forward a new feature selection strategy to identify good genes from a large amount of candidates in gene expression data analysis;

(3) presenting a new method for the study on patient survival prediction, including selecting informative training samples, choosing related genes and building an SVM-based scoring model;

(4) applying the proposed techniques to published gene expression profiles and proteomic data, and addressing various comparisons on classification and feature selection methods from a large amount of experimental results;

(5) pointing out significant genes from each analysed data set, comparing them with literature and relating some of them to the relevant diseases;

(6) recognizing two types of functional sites in DNA sequence data by using k-gram amino acid or nucleotide acid patterns to construct feature space and validating learning models across different independent data sets.

1.3 Structure

Chapter 2 first defines terms and introduces some concepts of supervised machine learning. Then it reviews some learning algorithms and techniques, including support vector machines (SVM), k-nearest neighbour (k-NN) and decision tree induction. Presenting methods of ensemble decision trees is the emphasis of this chapter and state-of-the-art algorithms, such as Bagging, Boosting, Random forests, are described in detail. Newly implemented and published CS4 (cascading-and-sharing for decision trees) is illustrated at the end, which makes use of different top-ranked features as the root node of a decision tree in an ensemble.

Chapter 3 surveys feature selection techniques for data mining. It begins with introducing two broad categories of selection algorithms — filter and wrapper, and indicating that filter is more suitable to solve biological problems. Then it presents a variety of common filter methods, such as t-statistic measure, Wilcoxon rank sum test, entropy-based measures, principal components analysis and so on. Following these methods, there comes ERCOF, our proposed 3-phase feature filtering strategy for gene expression data analysis. The chapter ends with a discussion on applying feature selection to bioinformatics.

Chapter 4 is a literature review of microarray gene expression data studies. The idea of microarray experiments and the problems arising from gene expression data are introduced before the extensive survey on various technologies that have been involved in this research area. These technologies are described in terms of data preprocessing, gene selection, supervised learning, clustering, and patient survival analysis.

Chapter 5 describes in detail my experimental work on phenotype classification from gene expression data. The chapter starts with illustrating the proposed feature selection and supervised learning scenarios, experimental design and evaluation methods. Then, it presents more than 1,000 experimental results obtained from six gene expression profiles and one proteomic data. For each data set, not only the classification and prediction accuracy is given, but also the selected discriminatory genes are reported and related to the literature and the disease. Some comparisons among feature selection methods and learning algorithms are also made based on the large amount of experimental results. ERCOF and CS4 are shown to be the best feature selection method and ensemble tree algorithm, respectively.

Chapter 6 presents my work on patient survival prediction using gene expression data. A new method is illustrated in detail according to the order of selecting informative training samples, identifying related genes and building an SVM-based scoring model. Case studies, on survival prediction for patients after chemotherapy for diffuse large-B-cell lymphoma and Stage I and III lung adenocarcinomas, are presented following the description of the method.

Chapter 7 is my work on applying data mining technologies to recognize functional sites in DNA sequences. The chapter begins with describing our method of feature manipulation for dealing with sequence data, with the stress on feature generation using k-gram nucleotide acid or amino acid patterns. Then the method is applied to identify translation initiation site (TIS) and polyadenylation (poly(A)) signal. The presentation order for each application is: background knowledge, data sets description, experimental results, and discussion. For both TIS and poly(A) signal recognitions, results achieved by our method are comparable or superior to previously reported ones, and several independent data sets are used to test the effectiveness and robustness of our prediction models.

Chapter 8 makes conclusions and suggests future work.

Figure 1.1 shows the structure of this thesis in a graph.

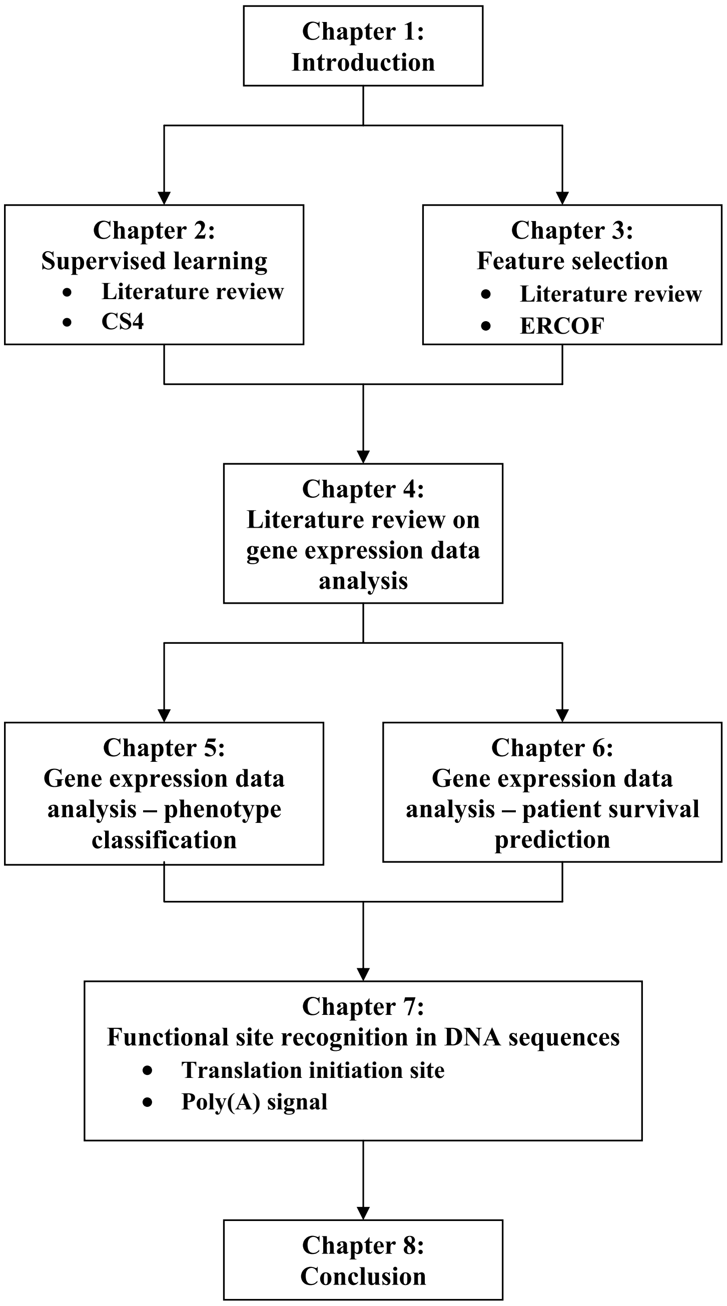

Chapter 1:
Introduction
Chapter 2:
Supervised learning
• Literature review
• CS4
Chapter 3:
Feature selection
• Literature review
• ERCOF
Chapter 4:
Literature review on gene expression data analysis
Chapter 5:
Gene expression data analysis – phenotype classification
Chapter 6:
Gene expression data analysis – patient survival prediction
Chapter 7:
Functional site recognition in DNA sequences
• Translation initiation site
• Poly(A) signal
Chapter 8:
Conclusion

Figure 1.1: Thesis structure.

Chapter 2

Classification — Supervised Learning

Data mining is to extract implicit, previously unknown and potentially useful information from data [134]. It is a learning process, achieved by building computer programs to seek regularities or patterns from data automatically. Machine learning provides the technical basis of data mining. One major type of learning we will address in this thesis is called classification learning, which is a generalization of concept learning [122]. The task of concept learning is to acquire the definition of a general category given a set of positive class and negative class training instances of the category [78]. Thus, it infers a boolean-valued function from training instances. As a more general format of concept learning, classification learning can deal with more than two class instances. In practice, the learning process of classification is to find models that can separate instances in the different classes using the information provided by training instances. Thus, the models found can be applied to classify a new unknown instance to one of those classes. Putting it more prosaically, given some instances of the positive class and some instances of the negative class, can we use them as a basis to decide if a new unknown instance is positive or negative [78]. This kind of learning is a process from general to specific and is supervised because the class membership of training instances are clearly known.

In contrast to supervised learning is unsupervised learning, where there is no pre-defined classes for training instances. The main goal of unsupervised learning is to decide which instances should be grouped together, in other words, to form the classes. Sometimes, these two kinds of learnings are used sequentially — supervised learning making use of class information derived from unsupervised learning. This two-step strategy has achieved some success in gene

Table 2.1: An example of gene expression data. There are two samples, each of which is described by 5 genes. The class label in the last column indicates the phenotype of the sample.

Gene1	Gene2	Gene3	Gene4	Gene5	Class
298	654	1284	800	163	ALL
2947	1811	198	679	225	AML

expression data analysis field [41, 6], where unsupervised clustering methods were first used to discover classes (for example, subtypes of leukemia) so that supervised learning algorithms could be employed to establish classification models and assign a phenotype to a newly coming instance.

2.1 Data Representation

In a typical classification task, data is represented as a table of *samples* (also known as *instances*). Each sample is described by a fixed number of *features* (also known as *attributes*) and a label that indicated its *class* [44]. For example, in studies of phenotype classification, gene expression data on m genes for n mRNA samples is often summarized by an $n \times (m+1)$ table $(X, Y) = (x_{ij}, y_i)$, where x_{ij} denotes the expression level of gene j in mRNA sample i, and y_i is the class (e.g. acute lymphoblastic leukemia) to which sample i belongs ($i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, m$). Table 2.1 shows two samples from a leukemia data set.

2.2 Results Evaluation

Evaluation is the key to making real progress in data mining [134]. To evaluate performance of classification algorithms, one way is to split samples into two sets, training samples and test samples. Training samples are used to build a learning model while test samples are used to evaluate the accuracy of the model. During validation, test samples are supplied to the model, having their class labels “hidden”, and then their predicted class labels assigned by the model are compared with their corresponding original class labels to calculate prediction accuracy. If two labels (actual and predicated) of a test sample are same, then the prediction to this sample is counted as a *success*; otherwise, it is an *error* [134]. An often used performance evaluation term is *error rate*, which is defined as the proportion of errors made over a whole set of test samples. In

		predicted class	
		A	***B***
actual class	***A***	**true positive**	**false negative**
	B	**false positive**	**true negative**

Figure 2.1: Confusion matrix for two-class classification problem.

some cases, we just simply use number of errors to indicate the performance. Note that, although the error rate on test samples is often more meaningful to evaluate a model, the error rate on the training samples is nevertheless useful to know as well since the model is derived from them.

Let's see the confusion matrix illustrated in Figure 2.1 of a two-class problem. The *true positive* (*TP*) and *true negative* (*TN*) are correct classifications in samples of each class, respectively. A *false positive* (*FP*) is when a class $\mathcal{B}$ sample is incorrectly predicted as a class $\mathcal{A}$ sample; a *false negative* (*FN*) is when a class $\mathcal{A}$ sample is predicted as a class $\mathcal{B}$ sample. Then each element of a confusion matrix shows the number of test samples for which the actual class is the row and the predicted class is the column. Thus, the error rate is just the number of false positives and false negatives divided by the total number of test samples (i.e. error rate = $(FP + FN)/(TP + TN + FP + FN)$).

Error rate is a measurement of overall performance of a classification algorithm (also known as a classifier); however, a lower error rate does not necessarily imply better performance on a target task. For example, there are 10 samples in class $\mathcal{A}$ and 90 samples in class $\mathcal{B}$. If $TP = 5$ and $TN = 85$, then $FP = 5$, $FN = 5$ and error rate is only 10%. However, in class $\mathcal{A}$, there are only 50% samples are correctly classified. To more impartially evaluate the classification results, some other evaluation metrics are constructed:

1. True positive rate (TP rate) = $TP/(TP + FN)$, also known as *sensitivity* or *recall*, which measures the proportion of samples in class $\mathcal{A}$ that are correctly classified as class $\mathcal{A}$.

2. True negative rate (TN rate) = $TN/(FP+TN)$, also known as *specificity*, which measures the proportion of samples in class $\mathcal{B}$ that are correctly classified as class $\mathcal{B}$.

3. False positive rate (FP rate) $= FP/(FP + TN)$ = $1 - specificity$.

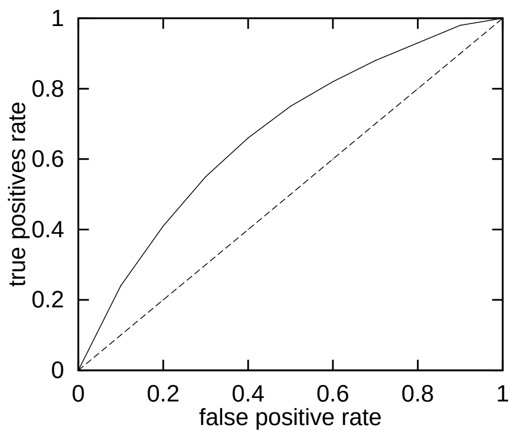


Figure 2.2: A sample ROC curve. The dotted line on the 45 degree diagonal is the expected curve for a classifier making random predictions.

4. False negative rate (FN rate) $= FN/(TP + FN)$ = $1 - sensitivity$.

5. Positive predictive value (PPV) $= TP/(TP + FP)$, also known as *precision*, which measures the proportion of the claimed class $\mathcal{A}$ samples are indeed class $\mathcal{A}$ samples.

In classification, it is a normal situation that along with a higher TP rate, there comes a higher FP rate, and same to the TN rate and FN rate. Thus, the receiver operating characteristic (ROC) curve was invented to characterize the tradeoff between TP rate and FP rate. The ROC curve plots TP rate on the vertical axis against FP rate on the horizontal axis. With an ROC curve of a classifier, the evaluation metric will be the area under the ROC curve. The larger the area under the curve (the more closely the curve follows the left-hand border and the top border of the ROC space), the more accurate the test. Thus, the ROC curve for a perfect classifier has an area of 1. The expected curve for a classifier making random predictions will be a line on the 45 degree diagonal and its expected area is 0.5. Please refer to Figure 2.2 for a sample ROC curve. ROC curve is widely used in bioinformatics domain, for example, it has been adopted to implement the evaluation scoring system of KDD Cup 2001 (`http://www.cs.wisc.edu/˜dpage/kddcup2001/`) and KDD Cup 2002 (`http://www.biostat.wisc.edu/˜craven/kddcup/`), both of them were about classifying biological data.

If the number of samples for training and testing is limited, a standard way of predicting the error rate of a learning technique is to use stratified k-fold cross validation. In k-fold cross validation, first, a full data set is divided randomly into k disjoint subsets of approximately equal size, in each of which the class is represented in approximately the sample proportions as in the full data set [134]. Then the above process of training and testing will be repeated k times on the k data subsets. In each iteration, (1) one of the subsets is held out in turn, (2) the classifier is trained on the remaining $k-1$ subsets to build classification model, (3) the classification error of this iteration is calculated by testing the classification model on the holdout set. Finally, the k number of errors are added up to yield an overall error estimate. Obviously, at the end of cross validation, every sample has been used exactly once for testing.

A widely used selection for k is 10. Why 10? "Extensive tests on numerous different data sets, with different learning techniques, have shown that ten is about the right number of folds to get the best estimate of error, and there is also some theoretical evidence that backs this up" [134]. Although 10-fold cross validation has become the standard method in practical terms, a single 10-fold cross validation might not be enough to get reliable error estimate [134]. The reason is that, if the seed of the random function that is used to divide data into subsets is changed, the cross validation with the sample classifier and data set will often produce different results. Thus, for a more accurate error estimate, it is suggested to repeat the 10-fold cross validation process ten times and average the error rates. This is called ten 10-fold cross validation and naturally, it is a computation-intensive undertaking.

Instead of running cross validation ten times, another approach for a reliable results is called *leave-one-out* cross validation (LOOCV). LOOCV is simply n-fold cross validation, where n is the number of samples in the full data set. In LOOCV, each sample in turn is left out and the classifier is trained on all the remaining $n-1$ samples. Classification error for each iteration is judged on the class prediction for the holdout sample, success or failure. Different from k-fold ($k < n$) cross validation, LOOCV makes use of the greatest possible amount of samples for training in each iteration and involves no random shuffling of samples.

2.3 Algorithms

There are various ways to find models that separate two or more data classes, i.e. do classification. Models derived from the same sample data can be very different from one classification algorithm to another. As a result, different models represent the knowledge learned in different formats as well. For example, decision trees represent the knowledge in a tree structure; instance-based algorithms, such as nearest neighbour, use the instances themselves to represent what is learned; naive Bayes method represents knowledge in the form of probabilistic summaries. In this section, we will describe a number of classification algorithms that have been used in the biomedical domain, including k-nearest neighbour, support vector machines and decision tree induction methods.

2.3.1 K-nearest neighbour

K-nearest neighbour (k-NN) is a typical instance-based classification and prediction algorithm. Learning in this kind of methods consists of simply storing the training data [78]. When a new instance comes, a set of similar related instances is retrieved from memory and used to classify the new instance. By k-NN, the class label of a new testing sample is decided by the majority class of its k closest training samples. The distance between two samples is measured by a certain metric. Generally, the standard Euclidean distance is used. If there are m features to describe a sample x and $f_i(x)$ denotes the value of ith feature ($i = 1, 2, \cdots, m$), then the Euclidean distance between two samples x_1 and x_2 is defined to be $d(x_1, x_2)$, where

$$d(x_1, x_2) \equiv \sqrt{\sum_{i=1}^{m} (f_i(x_1) - f_i(x_2))^2} \tag{2.1}$$

Note that using above distance metric assumes that the features are numeric, normalized and are of equal importance. If different features are measured on different scales and Euclidean distance is still used directly, the effect of some features might be completely dwarfed by others that have larger scales of measurement. Therefore, in such case, normalization must be conducted in advance. For nominal features whose values are symbolic rather than numeric, the distance between two values is often taken to be 1 if the values are not same, to be 0 if the values are same. No scaling is necessary in this case since only the values 0 and 1 are used. As for the selection of k, it can be done by running cross validation on training samples. The k for which the cross validation error rate is smallest is retained for use on further testing and prediction. In practice, 1, 3 and 5 are the generally adopted values for k.

Although the class prediction for a new sample relies on its k closest neighbours, the contribution of these k neighbours could not be treated equally since some of them might be a bit far from the target sample while some are closer to it. Thus, one refinement to k-NN algorithm is to weight the contribution of each of the k nearest neighbours according to their distance to the testing sample, assigning bigger weight to closer neighbours. For example, use $1/distance$ as the weight.

The nearest neighbour idea originated many decades ago, and k-NN started to be analyzed by statisticians in early 1950s [134]. Fix and Hodges published their pioneering analysis of the nearest neighbour in 1951 [37], and Johns first reported its usage in classification problem in 1961 [52]. Recently, k-NN has been widely used in classifying biomedical data — for example, gene expression data [135, 67, 140, 35, 10], and translation initiation site prediction in DNA sequences [142, 72]. However, there are some disadvantages of instance-based approaches.

(1) Generally, the cost of classifying new instances can be high. This is due to the fact that almost all computation happens at the classification time rather than when the training samples are loaded.

(2) Since there is no separate learning phase, all training samples have to be stored in the memory when class prediction for a new sample is done. This may consume a long-term unrealistic amounts of storage.

(3) Typically, instance-based algorithms, especially k-NN, consider all features when finding similar training samples from memory. This makes them very sensitive to feature selection.

(4) Most of the algorithms do not output explicit knowledge that is learned. When dealing with biomedical data, this drawback is conspicuous since comprehensible knowledge is expected by biologists and medical doctors.

2.3.2 Support vector machines

Support vector machines (SVM) is a kind of a blend of linear modeling and instance-based learning [134], which uses linear models to implement nonlinear class boundaries. It originates from research in statistical learning theory [130]. An SVM selects a small number of critical boundary samples from each class of training data and builds a linear discriminant function (also called *maximum margin hyperplane*) that separates them as widely as possible. The selected samples that are closest to the maximum margin hyperplane are called *support vectors*. Then the discriminant function $f(T)$ for a test sample T is a linear combination of the support vectors and

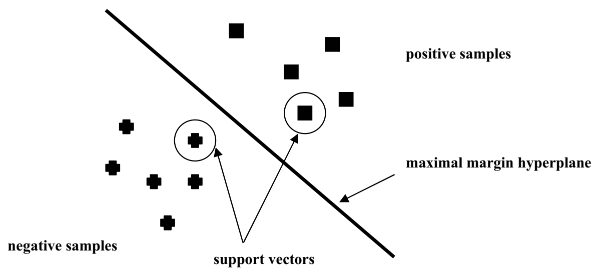


Figure 2.3: A linear support vector machine.

its constructed as:

$$f(T) = \sum_i \alpha_i y_i (X_i \cdot T) + b \tag{2.2}$$

where the vectors X_i are the support vectors, y_i are the class labels (which are assumed to have been mapped to 1 or -1) of X_i, vector T represents a test sample. $(X_i \cdot T)$ is the *dot product* of the test sample T with one of the support vectors X_i. α_i and b are numeric parameters (like weights) to be determined by the learning algorithm. Please see Figure 2.3 for representation of a linear SVM.

In the case that no linear separation is possible, the training data will be mapped into a higher-dimensional space $\mathcal{H}$ and an optimal hyperplane will be constructed there. The mapping is performed by a kernel function $K(\cdot,\cdot)$ which defines an inner product in $\mathcal{H}$. Different mappings construct different SVMs. When there is a mapping, the discriminant function is given like:

$$f(T) = \sum_i \alpha_i y_i K(X_i, T) + b \tag{2.3}$$

An SVM is largely characterized by the choice of its kernel function. There are two types of widely used kernel functions [24]: *polynomial* kernel and Gaussian *radial basis function* kernel (RBF).

- A polynomial kernel is $K(X_1, X_2) = (X_1 \cdot X_2 + 1)^d$, the value of power d is called degree and generally is set as 1, 2 or 3. Particularly, the kernel becomes a linear function if $d = 1$. It is suggested to choose the value of degree starting with 1 and increment it until the estimated error ceases to improve. However, it has been observed that the degree of a polynomial kernel plays a minor role in the final results [106] and sometime, linear function performs better than quadratic and cubic kernels due to overfitting of the latter kernels.

- An RBF kernel has the form $K(X_1, X_2) = exp(-\frac{||X_1 - X_2||^2}{2\sigma^2})$, where σ is the width of the Gaussian. The selection of parameter σ can be conducted via cross validation or some other manners. In [23], when using SVM with RBF kernel on gene expression data analysis, Brown *et al* set σ equal to the median of the Euclidean distances from each positive sample (sample with class label as 1) to the nearest negative sample (sample with class label as -1).

Besides polynomial kernel and Gaussian RBF kernel, other kernel functions include sigmoid kernel [108], B_n-spline kernel [108], locality-improved kernel [145], and so on. A tutorial of SVM can be found in [24].

In order to determine parameters α and b in (2.3), the construction of the discriminant function finally turns out to be a constrained quadratic problem on maximizing the Lagrangian dual objective function [131]:

$$\max_{\alpha} W(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j K(X_i, X_j) \tag{2.4}$$

under constraints

$$\sum_{i=1}^{n} \alpha_i y_i = 0, \alpha_i \geq 0, (i = 1, 2, \cdots, n) \tag{2.5}$$

where n is the number of samples in training data. However, the quadratic programming (QP) problem in equation (2.4) can not be solved easily via standard techniques since it involves a matrix that has a number of elements equals to the square of the number of training samples. To efficiently find the solution of the above QP program, Platt developed the sequential minimal optimization (SMO) algorithm [93] — one of the fastest SVM training methods. Like other SVM training algorithms, SMO breaks the large QP problem into a series of smaller possible QP problems. Unlike other algorithms, SMO tackles these small QP problems analytically, which avoids using a time-consuming numerical QP optimization as an inner loop. The amount of memory required by SMO is linear with number of training samples [93]. SMO has been implemented into *Weka*, a data mining software package [134].

SVMs have been shown to perform well in multiple areas of biological analysis, such as detecting remote protein homologies, recognizing translation initiation sites [145, 142, 72], and prediction of molecular bioactivity in drug design [132]. Recently, more and more bioinformaticians employ SVMs in their research on evaluating and analyzing microarray expression data [23, 39, 140]. SVMs have many mathematical features that make them attractive for gene expression analysis, including their flexibility in choosing a similarity function, sparseness of solution when dealing with large data sets, the ability to handle large feature spaces, and the ability to identify outliers [23]. Among many published works in this area, Brown *et al* [23] studied an expression data set from 2467 genes from the budding yeast Saccharomyces cerevisiae measured in 79 different DNA microarray hybridization experiments. Their results show that SVMs outperformed Parzen window, Fisher's linear discriminant and two decision tree classifiers (C4.5 and MOC1). Furey *et al* [39] analysed three data sets: ovarian cancer [109], colon cancer [84] and subtype leukaemia [41]. They reported low test errors on these data sets despite the small number of tissue samples available for investigation.

On the other hand, in [76], Meyer *et al* did a bench mark study on comparison of SVMs with 16 classification methods based on their performance on 21 data sets from widely used UCI machine learning database [15]. These classifiers include k-NN, classification trees (bagging, random forests and multiple additive regression trees), linear/quadratic discriminant analysis, neural networks and so on. For SVMs, they used the C++ library LIBSVM at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`. They evaluated the performance of an algorithm by classification error and mean squared error. They drew their conclusions that: "support vector machines yielded good performance, but were not top ranked on all data sets. Simple statistical procedures and ensemble methods proved very competitive, mostly producing good results 'out of the box' without the inconvenience of delicate and computationally expensive hyperparameter tuning. ...... In short, our results confirm the potential of SVMs to yield good results, but their overall superiority can not be attested".

In many practical data mining applications, success is measured more subjectively in terms of how acceptable the learned description — rules, decision trees, or whatever — are to a human user [134]. This measurement is especially important to biomedical applications such as cancer studies where comprehensive and correct rules are crucial to help biologists and doctors understand the disease.

2.3.3 Decision trees

Decision tree induction is among the most popular classification methods. As mentioned above, decision tree has an important advantage over other machine learning algorithms such as k-NN and SVM, in a qualitative dimension: rules produced by decision tree induction are easy to interpret and understand, and hence, can help greatly in appreciating the underlying mechanisms that separate samples in different classes.

In general, decision trees try to find an optimal partitioning of the space of possible observations, mainly by the means of subsequent recursive splits. Most of the algorithms implement this induction process in a *top-down* manner: (1) determining the root feature that most discriminatory with regard to the entire training data; (2) using the root feature to split the data into non-overlapping subsets; (3) selecting a significant feature of each of these subsets to recursively partition them until reaching one of stopping criteria. This idea was first developed by Ross Quinlan and his classic paper was published in 1986 [96]. Figure 2.4 is a decision tree example from a study of gene expression in two subtypes of acute leukemias, acute lymphoblastic leukemia (ALL) and acute myeloid leukemia (AML). To classify a new sample, a decision tree sorts the sample down the tree from the root to some leaf node, which provides the classification of the sample. Established decision trees can also be re-presented as sets of *if-then* rules to improve human readability. For example, from the left-most branch of the decision tree illustrated in Figure 2.4, a decision rule can be derived as "if *Attribute2233*$\leq$*80.34* and *Attribute4847*$\leq$*506.77*, then *the sample is an ALL sample*".

Among many decision tree based classifiers, C4.5 [97] is a well-established and widely used algorithm. C4.5 uses the information *gain ratio* criterion to determine the most discriminatory feature at each step of its decision tree induction process. In each round of selection, the gain ratio criterion chooses, from those features with an average-or-better information gain, the feature that maximizes the ratio of its gain divided by its entropy. C4.5 stops recursively building sub-trees when (1) an obtained data subset contains samples of only one class (then the leaf node is labeled by this class); or (2) there is no available feature (then the leaf node is labeled by the majority class); or (3) when the number of samples in the obtained subset is less than a specified threshold

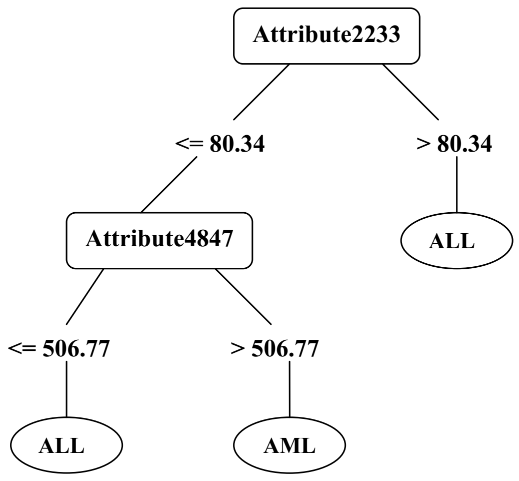


Figure 2.4: A decision tree for two types (ALL v.s. AML) acute leukemias classification. Branches correspond to the values of attributes (genes); leaves indicate classifications.

(then leaf node is labeled by the majority class). The precise definition and calculation formulae of information gain and gain ratio are given in Section 3.22 of Chapter 3. After obtaining a learned decision tree, C4.5 conducts tree *post-pruning* to make a decision tree simple and reduce the probability of over-fitting the training data.

This pruning is known as *reduced error pruning*. For each of the nodes in the tree, the traditional process of this pruning consists of removing the subtree rooted at a node, making it a leaf node and assigning it the most common class of the training samples affiliated with that node. A node is removed only if the resulting pruned tree performs no worse than the original over the cross validation set [78]. Since the performance is measured on validation set, this pruning strategy suffers from the disadvantage that the actual tree is based on less data. However, in practice, C4.5 makes some estimate of error based on training data itself — using the upper bound of a confidence interval (by default is 25%) on the resubstitution error. The estimated error of the leaf is within one standard deviation of the estimated error of the node. Besides reduced error pruning, C4.5 also provides another pruning option known as *subtree raising*. In subtree raising, an internal node might be replaced by one of nodes below and samples will be redistributed. For a detailed illustration on how C4.5 conducts its post-pruning, please refer to [97, 134].

Other algorithms for decision tree induction include ID3 (predecessor of C4.5) [96], C5.0

(successor of C4.5), CART (classification and regression trees) [22] (`http://www.salford-systems.com/`), LMDT (Linear Machine Decision Trees) [128], OC1 (oblique classifier 1) [81] and so on. This group of algorithms are most successful for analysis of clinical data and diagnosis from clinical data. Some examples include locating protein coding regions in Human DNA [104], prediction of post-traumatic acute lung injury [99], identification of acute cardiac ischemia [110], prediction of neurobehavioral outcome in head-injury survivors [120]. More recently, they have been used to learn from gene expression data to reconstruct molecular networks [117] or classify tumors [35].

2.3.4 Ensemble of decision trees

Ensemble methods are learning algorithms that construct a set of classifiers and then classify new samples by taking a vote of their predictions [33]. Generally speaking, an ensemble method can increase predictive performance over a single classifier. In [33], Dietterich gave three fundamental reasons for why ensemble methods are able to outperform any single classifier within the ensemble — in terms of statistical, computational and representational issues. Besides, plenty of experimental comparisons have been performed to show significant effectiveness of ensemble methods in improving the accuracy of single base classifiers [98, 13, 34, 20, 107].

The original ensemble method is Bayesian averaging [33], but bagging (bootstrap aggregation) [19] and boosting [38] are two of most popular techniques for constructing ensembles. Next, we will introduce how these two ideas and some other ensemble methods are implemented to generate decision tree committees.

Bagging of decision trees

The technique of bagging was coined by Breiman [19], who investigated the properties of bagging theoretically and empirically for both classification and numeric prediction. Bagging of trees combines several tree predictors trained on bootstrap samples of the training data and gives prediction by taking majority vote. In bagging, given a training set S with n samples, a new training set S' is obtained by drawing n samples uniformly with replacement from S. When there is a limited amount of training samples, bagging attempts to neutralize the instability of single decision tree classifier by randomly deleting some samples and replicating others. The instability

```
Generation of trees:
Let $n$ be the number of samples in the training data $S$.
For each of $k$ iterations:
  Obtain a new training set $S'$ by drawing $n$ samples with replacement from $S$.
  Apply the decision tree algorithm to $S'$.
  Store the resulting tree.

Classification:
Given a new sample.
For each of the $k$ trees:
  Predict class of sample according to the tree.
Return class that has been predicted most often.
```

Figure 2.5: Algorithm for bagging.

inherent in learning algorithms means that small changes to the training set cause large changes in the learned classifier. Figure 2.5 is the algorithm for bagging.

Boosting of decision trees

Unlike bagging where individual trees are built independently, each new tree generated in boosting is influenced by the performance of those built previously. Boosting encourages new trees to become "experts" for samples handled incorrectly by earlier ones [134]. When making classification, boosting weights a tree's contribution by its performance, rather than giving equal weight to all trees which is adopted by bagging.

There are many variants on the idea of boosting. The version introduced below is called *AdaBoostM1* which was developed by Freund and Schapire [38] and designed specifically for classification. The AdaBoostM1 algorithm maintains a set of weights over the training data set S and adjusts these weights after each iteration learning of the base classifier. The adjustments increase the weight of samples that are misclassified and decrease the weight of samples that are properly classified. By weighting samples, the decision trees are forced to concentrate on those samples with high weight. There are two ways that AdaBoostM1 manipulates these weights to construct a new training set S' to feed to the decision tree classifier [134]. One way is called *boosting by sampling*, in which samples are drawn with replacement from S with probability proportional to their weights. Another way is *boosting by weighting*, in which the presence of sample weights changes the error calculation of tree classifier — using the sum of the weights

```
Generation of trees:
Let $n$ be the number of samples in the training data $S$.
Assign equal weight $1/n$ to each sample in $S$.
For each of $k$ iterations:
  Apply decision tree algorithm to weighted samples.
  Compute error $e$ of the obtained tree on weighted samples.
  If $e$ is equal to zero:
    Store the obtained tree.
    Terminate generation of trees.
  If $e$ is greater or equal to 0.5:
    If the obtained tree is the first tree generated:
      Store the obtained tree.
    Terminate generation of trees.
  For each of samples in $S$:
    If sample is classified correctly by the obtained tree:
      Multiply weight of the sample by $e/(1-e)$.
  Normalize weight of all samples.

Classification
Given a new sample.
Assign weight of zero to all classes.
For each of the tree stored:
  Add $-log(e/(1-e))$ to the weight of the class predicted by the tree.
Return class with highest weight.
```

Figure 2.6: Algorithm for AdaBoostM1.

of the misclassified samples divided by the total weight of all samples, instead of the fraction of samples that are misclassified. Please refer to Figure 2.6 for a detailed algorithm of AdaBoostM1 using boosting by weighting.

Please note that the approach of boosting by weighting can be used only when the learning algorithm can cope with weighted samples. If this is not the case, an unweighted data set is generated from the weighted data by resampling. Fortunately, C4.5 decision tree induction algorithm has been implemented to deal with weighted samples. For more details about this, please refer to [98].

Besides bagging and boosting, Dietterich put forward an alternative but very simple idea, randomization trees, to build ensemble trees. With this idea, the split at each *internal* node is selected at random from the k (20 by default) best splits. In case of continuous attributes, each possible threshold is considered to be a distinct split, so the k best splits may all involve splitting on the same attribute. Experimentally, Dietterich [34] also compared randomization with bagging and boosting of constructing ensembles of C4.5 decision trees using 33 data sets. His experimental results showed that (1) when there is little or no classification noise, randomization is competitive with (sometime is slightly superior to) bagging but not as accurate as boosting; (2) where there is substantial classification noise, bagging is much better than boosting, and sometimes better than randomization.

Random forests

Random forests is based on bagged trees, but in addition uses random feature selection at each node for the set of splitting variables [20].

A more precise definition of random forests given in [20] is: "a random forest is a classifier consisting of a collection of tree-structured classifiers $h(X, V_k)$ ($k = 1, \cdots$), where the V_k are independent identically distributed random vectors and each tree casts a unit vote for the most popular class at input X". Using random forests, in the kth iteration, a random vector V_k is generated, independent of the past random vectors but with the same distribution. For instance, V_k is generated by drawing samples with replacement from original training data. Based on the bootstrapped data, in [20], the forests using randomly selected attribute or combinations of attributes at each node were studied. In the former case, at each node, m_{try} number of candidate features are selected from all m features and the best split on these m_{try} is used to split the node. m_{try} is defined by the user, and has the same value for each tree grown in the ensemble. It can take any value in the range of 1 to m. In [20], two values of m_{try} were tried — 1 and $int(log_2 m + 1)$. The experimental results illustrated that the algorithm is not very sensitive to the value of m_{try}. In the latter case, more features are defined by taking random linear combinations of a number of the original input attributes. This approach is used when there are only a few attributes available so that higher correlations between individual classifiers are expected. After a splitting feature is determined, random forests grow the tree using CART [22] methodology to maximum size and do not prune. Different from C4.5, CART selects splitting feature using GINI impurity criterion. Please refer to Figure 2.7 for the general algorithm of random forests.

In [21], Breiman claimed that "in random forests, there is no need for cross-validation or a separate test set to get an unbiased estimate of the test set error." The reason was as follows. In each of k iterations, about one-third of the samples are left out of the new bootstrap training set

```
Generation of trees:
Let $n$ be the number of samples in the training data $S$,
$k$ be the number of trees to grow,
$m_{try}$ be an integer and $m_{try} << m$, ($m$ is the number of features).
For each of $k$ iterations:
    Obtain a new training set $S'$ by drawing $n$ samples with replacement from $S$.
    Grow a tree, where at each node, the best split is chosen among $m_{try}$ randomly selected features.

Classification:
Given a new sample.
For each of the $k$ trees:
    Predict class of sample according to the tree.
Return class that has been predicted most often.
```

Figure 2.7: Algorithm for random forests.

and not used in the construction of the tree [20]. These samples are called "out-of-bag" (OOB) samples to which the tree built in this iteration will be applied to get classification. In this way, a test set classification is obtained for each sample in about one-third of the constructed trees. The final classification for a sample is the class having the most votes from the trees in the forest. Then the final classifications are compared with the real class labels of the OOB samples to achieve an OOB error estimation.

Although in random forests, the feature selection at each node is random, an upper bound for its generalization error still can be derived in terms of *strength* of the individual decision tree classifiers and their *correlations* [20]. This not only measures how accurate the individual classifiers are and the dependence between them, but also gives insight into the ability of the random forest to predict. The estimation for strength and correlation is conducted by the above out-of-bag idea. Please see Appendix II of [20] for more information about this issue. Random forests was claimed to achieve comparable or better accuracy than AdaboostM1 did. Besides, it is [20] (a) relatively robust to outliers and noise, (b) faster than bagging and boosting, (c) simple and easily parallelized. In addition, (d) useful internal estimates of error, strength, correlation and variable importance are possible to obtain.

CS4 — a new method of ensemble of decision trees

CS4 stands for *cascading-and-sharing for* decision trees. It is a newly developed classification algorithm based on an ensemble of decision trees. The main idea of this method is to use different top-ranked features as the root node of a decision tree in an ensemble (also named as a committee) [62, 63]. Different from bagging or boosting which uses bootstrapped data, CS4 always builds decision trees using exactly the same set of training samples. The difference between this algorithm and Dietterich's randomization trees is also very clear — the root node features of CS4 induced trees are different from each other while every member of a committee of randomized trees always shares the same root node feature (the random selection of the splitting feature is only applied to internal nodes). On the other hand, compared with the random forests method which selects splitting features randomly, CS4 picks up root node features according to their rank order of certain measurement (such as entropy, gain ratio). Thus, CS4 is claimed as a novel ensemble tree method.

In detail, to construct k number of decision trees ($k \leq m$, m is the number of features describe the data), we have following steps:

(1) Ranking all the m features according to a certain criterion, with the best feature at the first position.

(2) $i = 1$.

(3) Using the ith feature as root node to construct ith decision tree using base classifier.

(4) If $i < k$, increasing i by 1 and goto (3); otherwise, stop.

In this thesis, we use C4.5 as the base classifier of CS4 and information *gain ratio* (Section 3.22 of Chapter 3) as the measure to rank features.

In the classification phase, CS4 defines the *coverage* of a rule in a tree as the percentage of the samples in its class satisfying the rule. Suppose we have discovered k decision trees from our training set containing class $\mathcal{A}$ and class $\mathcal{B}$ samples. Then, all the rules derived from the k trees can be categorized into two groups: one group only containing rules for $\mathcal{A}$ samples, another containing rules for $\mathcal{B}$ samples. In each group, we rank the rules in descending order according to their coverage, such as

$$rule_1^{\mathcal{A}}, rule_2^{\mathcal{A}}, \cdots$$

and

$$rule_1^{\mathcal{B}}, rule_2^{\mathcal{B}}, \cdots$$

Given a test sample T, each of the k trees will have a rule to fit this sample and therefore, give a prediction for this sample. Suppose that T satisfies the following k_1 rules of class $\mathcal{A}$ samples and k_2 of class $\mathcal{B}$ samples:

$$rule(T)_1^{\mathcal{A}}, rule(T)_2^{\mathcal{A}}, \cdots, rule(T)_{k_1}^{\mathcal{A}},$$

and

$$rule(T)_1^{\mathcal{B}}, rule(T)_2^{\mathcal{B}}, \cdots, rule(T)_{k_2}^{\mathcal{B}}.$$

Where $0 \leq k_1, k_2 \leq k$ and $k_1 + k_2 = k$. The order of these rules is also based on their coverage. When we make a prediction for T, two scores will be calculated as follows:

$$Score(T)^{\mathcal{A}} = \sum_{i=1}^{k_1} \frac{coverage(rule(T)_i^{\mathcal{A}})}{coverage(rule_i^{\mathcal{A}})} / k \qquad (2.6)$$

$$Score(T)^{\mathcal{B}} = \sum_{i=1}^{k_2} \frac{coverage(rule(T)_i^{\mathcal{B}})}{coverage(rule_i^{\mathcal{B}})} / k \qquad (2.7)$$

If $Score(T)^{\mathcal{A}} \geq Score(T)^{\mathcal{B}}$, then T will be predicted as a class $\mathcal{A}$ sample; Otherwise, T predicted as a class $\mathcal{B}$ sample. In practice, the tie-score case occurs rarely [62].

The algorithm of CS4 can be easily applied to solve multi-class problems. If the given data set contains p classes samples ($p \geq 2$), similarly, we can sort p groups of top k rules according to their coverage. When classifying a sample T, those rules in the k trees which are satisfied by T are found and sorted. Then the classification score for a specific class $\mathcal{C}$ is calculated by

$$Score(T)^{\mathcal{C}} = \sum_{i=1}^{k_{\mathcal{C}}} \frac{coverage(rule(T)_i^{\mathcal{C}})}{coverage(rule_i^{\mathcal{C}})} / k \qquad (2.8)$$

The effectiveness of CS4 has been tested on some UCI data sets [15] as well as some public gene expression profiles that are described by more than 10,000 features [62]. One of the main works of this thesis is to do further comparison of CS4 with bagging, boosting, random forests as well as SVM and k-NN using a huge number of experimental results obtained from various biological data sets.

2.4 Chapter Summary

In this chapter, we introduced the concept of classification in data mining as well as the ways to evaluate the classification performance. We selected to present in detail some of classification algorithms — putting the emphasis on several methods using ensemble of decision trees, including bagging, boosting, randomization tree, random forests and the newly invented CS4. Besides, two widely used classifiers, SVM and k-NN were also described so that comparisons among decision tree methods, kernel function approaches and instance-based techniques can be addressed in the later chapters using experimental results.

Chapter 3

Feature Selection for Data Mining

A known problem in classification (in general machine learning) is to find ways to reduce the dimensionality of the feature space to overcome the risk of over-fitting. Data over-fitting happens when the number of features is large ("curse of dimensionality") and the number of training samples is comparatively small ("curse of data set sparsity"). In such a situation, a decision function can perform very well on classifying training data, but does poorly on test samples. Feature selection is concerned with the issue of distinguishing signal from noise in data analysis.

3.1 Categorization of Feature Selection Techniques

Feature selection techniques can be categorized according to a number of criteria [46]. One popular categorization is based on whether the target classification algorithm will be used during the process of feature evaluation. A feature selection method, that makes an independent assessment only based on general characteristics of the data, is named "filter" [134]; while, on the other hand, if a method evaluates features based on accuracy estimates provided by certain learning algorithm which will ultimately be employed for classification, it will be named as "wrapper" [55, 134]. With wrapper methods, the performance of a feature subset is measured in terms of the learning algorithm's classification performance using just those features. The classification performance is estimated using the normal procedure of cross validation, or the bootstrap estimator [134]. Thus, the entire feature selection process is rather computation-intensive. For example, if each evaluation involves a 10-fold cross validation, the classification procedure will be executed 10 times. For this reason, wrappers do not scale well to data sets containing many features [45]. Besides, wrappers have to be re-run when switching from one classification algorithm to another. In contrast to wrapper methods, filters operate independently of any learning algorithm and the features selected can be applied to any learning algorithm at the classification stage. Filters have been proven to be much faster than wrappers and hence, can be applied to data sets with many features [45]. Since the biological data sets discussed in the later chapters of this thesis often contain a huge number of features (e.g. gene expression profiles), we concentrate on filter methods.

Another taxonomy of feature selection techniques is to separate algorithms evaluating the worth or merit of a subset features from those of individual features. Most of the feature selection methods introduced in this chapter evaluate how well an individual feature contributes to the separation of samples in different classes and produce a simple feature ranking. However, there is also one method in this chapter, correlation-based feature selection, that assesses and selects a subset of features. We will also present a new feature selection algorithm, ERCOF, which first evaluates features individually and then forms the final representative feature set by considering the correlations between the features.

There are some other dimensions to categorize feature selection methods. For example, some algorithms can handle regression problem, that is, the class label is numeric rather than a discrete valued variable; and some algorithms evaluate and rank features independently from class, i.e. unsupervised feature selection. We will restrict our study to the data sets with discrete class label since this is the case of the biological problems analysed in later chapters of this thesis, though some algorithms presented can be applied to numeric class label as well.

3.2 Feature Selection Algorithms

There are various ways to conduct feature selection. Let us start with introducing some often used methods conducted by analysing the statistical properties of the data.

3.2.1 T-test, signal-to-noise and Fisher criterion statistical measures

Highly consistent with the well-known ANOVA principle, a basic concept for identifying a relevant feature from an irrelevant one is the following: if the values of a feature in samples of class $\mathcal{A}$ are significantly different from the values of the same feature in samples of class $\mathcal{B}$, then the feature is likely to be more relevant than a feature that has similar values in $\mathcal{A}$ and $\mathcal{B}$. More specifically, in order for a feature f to be relevant, its mean value $\mu_f^{\mathcal{A}}$ in $\mathcal{A}$ should be significantly different from its mean value $\mu_f^{\mathcal{B}}$ in $\mathcal{B}$. However, if the values of a feature f varies greatly within the same class of samples, even if $\mu_f^{\mathcal{A}}$ differs greatly from $\mu_f^{\mathcal{B}}$, the feature f is not a reliable one. This situation leads us to a second basic concept: the standard deviation $\sigma_f^{\mathcal{A}}$ and variance $(\sigma_f^{\mathcal{A}})^2$ of f in $\mathcal{A}$ and the standard deviation $\sigma_f^{\mathcal{B}}$ and variance $(\sigma_f^{\mathcal{B}})^2$ of f in $\mathcal{B}$ should be small.

The classical t-statistic is constructed to test the difference between means of two groups of independent samples. So if samples in different classes are independent, the t-statistic can be used to find features that has big difference in mean level between the two classes. These features can be then considered to have ability to separate samples between different classes.

Given a data set X consisting of n sample vectors:

$$X_i = (x_{i1}, \cdots, x_{im}, y_i) \tag{3.1}$$

where $1 \leq i \leq n$, m is the number of features and y_i is the class label of X_i. Each sample belongs to one of two classes $\mathcal{A}$ (i.e. $y_i = \mathcal{A}$) and $\mathcal{B}$ (i.e. $y_i = \mathcal{B}$) (such as *tumor* v.s. *normal*). Similarly, a feature in the data set can be denoted as f_j and x_{ij} stands for its value in sample i ($1 \leq j \leq m$). In addition, $n^{\mathcal{A}}$ (resp. $n^{\mathcal{B}}$) is the number of samples in class $\mathcal{A}$ (resp. $\mathcal{B}$). For each feature f_j, the mean $\mu_j^{\mathcal{A}}$ (resp. $\mu_j^{\mathcal{B}}$) and the standard deviation $\delta_j^{\mathcal{A}}$ (resp. $\delta_j^{\mathcal{B}}$) using only the samples labeled $\mathcal{A}$ (resp. $\mathcal{B}$) are calculated by

$$\mu_j^{\mathcal{A}} = \frac{\sum\limits_{y_k=\mathcal{A}} x_{kj}}{n^{\mathcal{A}}} \tag{3.2}$$

$$\delta_j^{\mathcal{A}} = \sqrt{\frac{\sum\limits_{y_k=\mathcal{A}} (x_{kj} - \mu_j^{\mathcal{A}})^2}{n^{\mathcal{A}}}} \tag{3.3}$$

A t score $t(f_j)$ for feature f_j then can be obtained by

$$t(f_j) = \frac{|\mu_j^{\mathcal{A}} - \mu_j^{\mathcal{B}}|}{\sqrt{\frac{(\delta_j^{\mathcal{A}})^2}{n^{\mathcal{A}}} + \frac{(\delta_j^{\mathcal{B}})^2}{n^{\mathcal{B}}}}} \tag{3.4}$$

The t-test statistical measure is known [105] to follow a Student distribution with

$$\frac{\left(\frac{(\sigma_j^{\mathcal{A}})^2}{n^{\mathcal{A}}} + \frac{(\sigma_j^{\mathcal{B}})^2}{n^{\mathcal{B}}}\right)^2}{\frac{\left(\frac{(\sigma_j^{\mathcal{A}})^2}{n^{\mathcal{A}}}\right)^2}{n^{\mathcal{A}}-1} + \frac{\left(\frac{(\sigma_j^{\mathcal{B}})^2}{n^{\mathcal{B}}}\right)^2}{n^{\mathcal{B}}-1}} \tag{3.5}$$

degrees of freedom. A feature f_j can be considered better than a feature f_l $(l \neq j)$ if $t(f_j) > t(f_l)$. Thus, when making feature selection, we can simply sort candidate features by their t scores and pick those with largest scores. In [82], t score is used to select important genes for classification after applying the algorithm of partial least squares to the original high dimension gene expression data.

In [41, 116, 39], a slightly different statistical measure from t-test was proposed to find discriminatory genes that can distinguish tumor cells from normal ones using gene expression profilings. This test s is named *signal-to-noise* statistical measure and is constructed as

$$s(f_j) = \frac{|\mu_j^{\mathcal{A}} - \mu_j^{\mathcal{B}}|}{\delta_j^{\mathcal{A}} + \delta_j^{\mathcal{B}}} \tag{3.6}$$

As with t-test, when using signal-to-noise statistical measure, a feature f_j can be considered better than a feature $f_l(l \neq j)$ if $s(f_j) > s(f_l)$, so we always pick those features with largest scores. Compared with t-test, the statistical property of signal-to-noise is not fully understood.

Another statistical measure that is closely related to the t-test is the *Fisher criterion score*, defined as

$$fisher(f_j) = \frac{(\mu_j^{\mathcal{A}} - \mu_j^{\mathcal{B}})^2}{(\delta_j^{\mathcal{A}})^2 + (\delta_j^{\mathcal{B}})^2} \tag{3.7}$$

A feature f_j can be considered better than a feature f_l $(l \neq j)$ if $fisher(f_j) > fisher(f_l)$. In [68], Fisher criterion score is used to select genes to distinguish two subtypes of leukemia from expression profilings.

T-test, signal-to-noise and Fisher criterion statistical measures are easy to compute and thus straightforward to use. However, there are three considerations that may make them ineffective for feature selection [72]. The first consideration is that use of these tests are justified only if it can be assumed that the data have a normal distribution, and this is almost not the case of biological data. The second consideration is that the sample sizes $n^{\mathcal{A}}$ and $n^{\mathcal{B}}$ should be sufficiently large; otherwise, underestimates of the standard deviations and variances will occur. The third consideration is more subtle and we illustrate it using an example.

Let f_1 and f_2 be two features. Suppose f_1 has values ranging from 0 to 99 in class $\mathcal{A}$ with $\mu_1^{\mathcal{A}} = 75$ and has values ranging from 100 to 199 in class $\mathcal{B}$ with $\mu_1^{\mathcal{B}} = 125$. Suppose f_2 has values ranging from 25 to 125 in class $\mathcal{A}$ with $\mu_2^{\mathcal{A}} = 50$ and has values ranging from 100 to 175 in class $\mathcal{B}$ with $\mu_2^{\mathcal{B}} = 150$. We see that $\mu_2^{\mathcal{B}} - \mu_2^{\mathcal{A}} = 100 > 50 = \mu_1^{\mathcal{B}} - \mu_1^{\mathcal{A}}$. Suppose the variances of f_1 and f_2 in $\mathcal{A}$ and $\mathcal{B}$ are comparable. Then according to the t, s and $fisher$ measures, f_2 is better than f_1. However, we note that the values of f_1 are distributed so that all those in $\mathcal{A}$ are below 100 and all those in $\mathcal{B}$ are at least 100. In contrast, the values of f_2 in $\mathcal{A}$ and $\mathcal{B}$ overlap in the range 100 to 125. Then clearly f_1 should be preferred. The effect is caused by the fact that t, s and $fisher$ are sensitive to all changes in the values of f, including those changes that may not be important. When dealing with gene expression data, one of the pre-processing works is to transform the data into the space of log-ratios by taking the logarithm of each gene (i.e. feature) divided by the median of that gene across a set of experiments [85]. It has been shown that the rankings of same set of candidate features, that based on t, s or $fisher$ statistical measures, might be different before and after this logarithm transformation.

3.2.2 Wilcoxon rank sum test

In order to avoid the assumption that feature values have to follow normal distribution, one can use non-parametric tests. One of the best known non-parametric tests is *Wilcoxon rank sum test*, or the equivalent Mann-Whitney test. Wilcoxon rank sum test [133] is an alternative to t-test for testing the quality of two populations' mean or medians. It is a kind of non-parametric test since it is based on rank of samples rather than distribution parameters such as mean and standard deviation. It does not require the two populations to conform to a normal distribution, but to the same shape [105]. However, it may not be as powerful as t-test, signal-to-noise or Fisher criterion statistical measures, if the normality assumption is correct [105].

The Wilcoxon rank sum test statistical measure of a feature f_j, $w(f_j)$, can be obtained using following procedure:

(1) Sort the values $x_{1j}, x_{2j}, \cdots, x_{nj}$ of f_j across all the n samples in ascending order.

(2) Assign rank (from 1) $r(x_{ij})$ to each value x_{ij} above and use average of the ranks for ties. Then, $1 \leq r(x_{ij}) \leq n$.

(3) Use the sum of the ranks for the class, which has smaller number of samples, as test statistic, $w(f_j)$. For example, class $\mathcal{A}$ has fewer samples than class $\mathcal{B}$, then

$$w(f_j) = \sum_{y_i = \mathcal{A}} r(x_{ij}) \tag{3.8}$$

where y_i is the class label of sample X_i. If the number of samples is same in each class, the choice of which class to use for the test statistic is arbitrary.

To use the Wilcoxon rank sum test to decide if a feature f is relevant, we set up the null hypothesis that: the values of f have the same continuous distribution in $\mathcal{A}$ and $\mathcal{B}$. Then $w(f)$ is used to accept or reject the hypothesis. To decide whether to accept or reject the null hypothesis, we compare $w(f)$ with the upper and lower critical values derived from a significant level α. For small numbers of samples in class $\mathcal{A}$ and $\mathcal{B}$, e.g. < 10, the critical values have been tabulated and can be found in most of textbooks of statistics, such as [105]. If either $n^{\mathcal{A}}$ or $n^{\mathcal{B}}$ is larger than what is supplied in the table, the following normal approximation can be used [105]. The expected value of w is (assuming class $\mathcal{A}$ has fewer samples than class $\mathcal{B}$ does):

$$\mu = \frac{n^{\mathcal{A}}(n^{\mathcal{A}} + n^{\mathcal{B}} + 1)}{2} \tag{3.9}$$

The standard deviation of w is:

$$\delta = \sqrt{\frac{n^{\mathcal{A}} n^{\mathcal{B}}(n^{\mathcal{A}} + n^{\mathcal{B}} + 1)}{12}} \tag{3.10}$$

The formula for calculating the upper and lower critical values is:

$$\mu_{\pm} z_{\alpha} \delta \tag{3.11}$$

where z_α is the z score for significant level α. If a feature f's test $w(f)$ falls in the range given by the upper and lower critical values, then we accept the null hypothesis; otherwise, we reject the hypothesis, and this indicates that the values of feature f have different distribution between samples in class $\mathcal{A}$ and $\mathcal{B}$. Thus, those features whose Wilcoxon rank sum test statistic rejects the hypothesis will be considered as signals.

The non-parametric Wilcoxon rank sum test has several advantages over t-test, signal-to-noise and Fisher criterion statistical measures [87]. The first one is its robustness. Because it uses ranks rather than actual values of a feature, it is more robust to outliers. This feature is important to biological data, which may need many steps of experiments in the laboratory and may have many potential sources of error. The second advantage is related to data transformation, such as normalization and logarithm transformations that are often used in preprocessing of microarray gene expression data. The rank sum test is not affected by any of these transformations since the ordering of the expression levels remains unchanged.

3.2.3 $\mathcal{X}^2$ statistical measure

$\mathcal{X}^2$ measure evaluates features individually by measuring the $\mathcal{X}^2$-statistic with respect to the class. Different from the preceding methods, $\mathcal{X}^2$ measure can only handle features with discrete values. $\mathcal{X}^2$ measure of a feature f with w discrete values is defined as

$$\mathcal{X}^2(f) = \sum_{i=1}^{w}\sum_{j=1}^{k} \frac{(A_{ij} - E_{ij})^2}{E_{ij}} \tag{3.12}$$

where k is the number of classes, A_{ij} is the number of samples with ith value of f in jth class, E_{ij} is the expected frequency of A_{ij} and

$$E_{ij} = R_i * C_j / n \tag{3.13}$$

R_i is the number of samples having ith value of f, C_j is the number of samples in the jth class, and n is the total number of samples.

We consider a feature f_j to be more relevant than a feature f_l $(l \neq j)$ if $\mathcal{X}^2(f_j) > \mathcal{X}^2(f_l)$. Obviously, the worst $\mathcal{X}^2$ value is 0 if the feature has only one value. The degree of freedom of the $\mathcal{X}^2$-statistic measure is $(w-1)*(k-1)$ [71]. With the degree of freedom known, the critical value for certain significant level can be found from most statistics books, such as [105]. However, note that, the value w might be varied from feature to feature.

To apply $\mathcal{X}^2$ measure to numeric features, a discretization preprocessing has to be taken. The most popular technique in this area is the state-of-art supervised discretization algorithm developed by Fayyad and Irani [36] based on the idea of entropy. At same time, feature selection can be also conducted as a by-product of discretization.

3.2.4 Entropy based feature selection algorithms

Entropy is a measure commonly used in information theory, which characterizes the (im)purity of a collection of samples [112, 78]. Given a collection S, containing samples in k classes, the entropy of S relative to this k classes classification is defined as

$$Ent(S) \equiv \sum_{i=1}^{k} -p_i * log_2 p_i \tag{3.14}$$

where p_i is the proportion of S belonging to class i. There are several points worth noting.

1. The logarithm is base 2 because entropy is a measure of the expected encoding length measured in bits [112].

2. In all calculations involving entropy, we define $0 * log_2 0 = 0$

3. $Ent(S)$ reaches its minimum value 0, if all the samples of S belong to the same class. For example, all samples are in class $\mathcal{A}$, then

$$p_i = \begin{cases} 1 & (i = \mathcal{A}); \\ 0 & (i \neq \mathcal{A}). \end{cases} \tag{3.15}$$

   Thus, $Ent(S) = -1 * log_2 1 = 0$.

4. $Ent(S)$ reaches its maximum value $log_2 k$, if S contains equal number of samples in each class. In this case, $p_i = 1/k$, for any $i \in [1, k]$. Thus,

$$Ent(S) = -k * (\frac{1}{k} log_2 \frac{1}{k}) = log_2 k \tag{3.16}$$

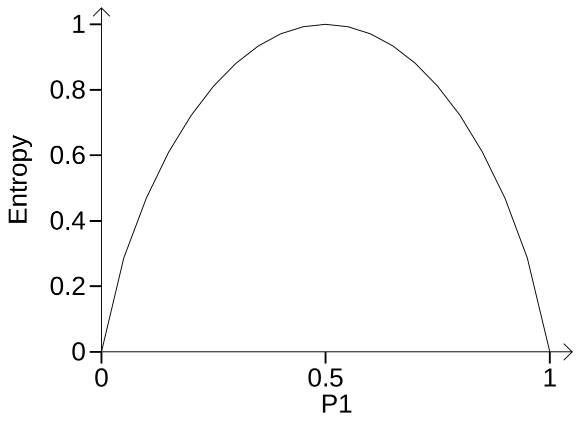


Figure 3.1: Entropy function of a two-class classification, p_1 is the proportion of samples in one class, with range [0,1].

Figure 3.1 shows the form of the entropy function when $k = 2$ (i.e. two classes), as p_1 varies between 0 and 1.

Fayyad's discretization algorithm

The essential idea of this discretization algorithm is to find some cut point(s) for a numeric feature's value range to make the resulting value intervals as pure as possible. Formally, let cut point T of feature f partition the sample set S into subsets S_1 and S_2. Then, the *class information entropy* of the partition, denoted $Ent(f, T, S)$, is given by [36]:

$$Ent(f, T, S) = \frac{|S_1|}{|S|} Ent(S_1) + \frac{|S_2|}{|S|} Ent(S_2) \tag{3.17}$$

where $Ent(S_j), (j = 1, 2)$ is the *class entropy* of a subset S. Assuming there are k classes $C_1, \cdots, C_k$, let $P(C_i, S_j)$ be the proportion of samples in S_j that have class C_i. According to the definition in (3.14),

$$Ent(S_j) = -\sum_{i=1}^{k} P(C_i, S_j) * log_2 P(C_i, S_j) \tag{3.18}$$

A binary discretization for f is determined by selecting the cut point T_f for which $Ent(f, T_f, S)$ is minimal amongst all the candidate cut points [36]. The selection of T_f can be achieved by recursively partitioning the ranges S_1 and S_2 until some stopping criteria is reached. A stopping criteria is needed because otherwise, we can always achieve perfect entropy by partitioning the range into many small intervals, each containing exactly one sample. A commonly used stopping criteria is the so-called *minimal description length* (MDL) principle described in [101, 36]. According to this principle, recursive partitioning within a range S stops iff S is partitioned into ranges S_1 and S_2 such that:

$$Gain(f,T,S) < \frac{log_2(n-1)}{n} + \frac{\delta(f,T,S)}{n} \tag{3.19}$$

where n is the number of samples in the set S, and,

$$Gain(f,T,S) = Ent(S) - Ent(f,T,S) \tag{3.20}$$

and

$$\delta(f,T,S) = log_2(3^k - 2) - [k * Ent(S) - k_1 * Ent(S_1) - k_2 * Ent(S_2)] \tag{3.21}$$

where k_i is the number of class labels represented in the range S_i. In the right side of (3.19), the first component is the amount of information needed to specify the partitioning point; the second one is a correction due to the need to transmit which classes correspond to upper and lower subintervals [36, 134]. With MDL principle, a feature f can not be discretized, if there is no such kind of cut point T whose $Gain(f,T,S)$ (defined in (3.20)) is greater than or equal to the right side of (3.19).

In [71], Setiono and Liu noted that discretization has the potential to perform feature selection among numeric features. If the distribution of a numeric feature's value is relatively random, then the feature would be treated as irrelevant to the classes and can be safely removed from the data set. In this case, there is no suitable cut point to split feature's value range, or, in other words, the feature can be only discretized to a single value. On the other hand, if a resulting value interval induced by the cut points of a feature contains only the same class of samples, then this partitioning of this feature has an entropy value of 0. This is an ideal case since the feature can clearly distinguish samples in the different classes. Please refer to Figure 3.2 for an illustration on entropy measure, cut point and intervals. Generally, under the entropy measure, feature f_j is more useful than feature f_l $(l \neq j)$ if $Ent(f_j, T_{f_j}, S) < Ent(f_l, T_{f_l}, S)$. Thus, when using entropy measure to select features, we sort the class entropy in an ascending order and consider

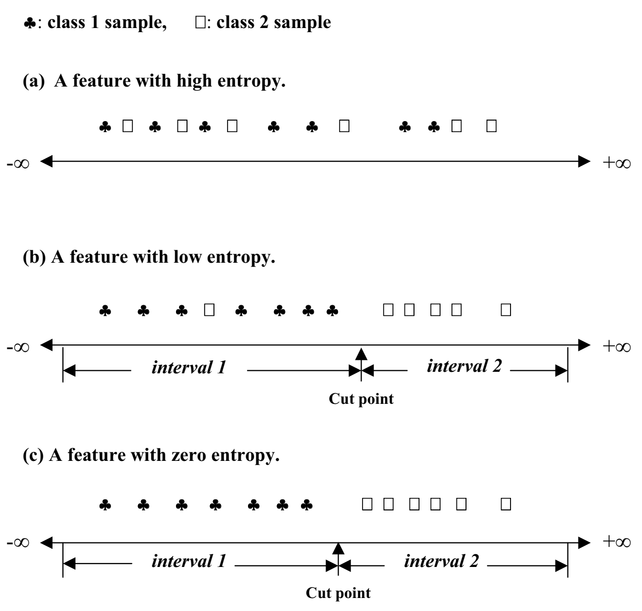

♣: class 1 sample, □: class 2 sample
(a) A feature with high entropy.
(b) A feature with low entropy.
(c) A feature with zero entropy.
-∞
+∞
interval 1
interval 2
Cut point

Figure 3.2: We place the values of a feature on the horizontal axis. There are 13 samples in two classes, class 1 and class 2. (a) shows a feature that is a poor signal and there is no cut point can be found to distinguish samples in the different classes; (b) shows a feature that is a potentially good signal and indicates a possible cut point. (c) shows a feature that is a strongest signal and indicates a cut point — different resulting intervals contains samples of different class.

those features with lowest values. In most of the cases, we are just interested in features having cut point(s) found for their value range.

For discrete features, we still can use entropy measure to select features since the "cut points" for each feature have been given naturally. Thus the class entropy of a feature f with w different values, can be simply derived by

$$Ent(f, S) = \sum_{i=1}^{w} \frac{|S_i|}{|S|} Ent(S_i) \tag{3.22}$$

where S_1 through S_w are the w subsets of samples resulting from partitioning of S by f and $Ent(S_i)$ can be calculated from (3.18).

Actually, $\mathcal{X}^2$ measure is one of the refinements of entropy measure. Other than the class entropy value of a feature, it uses the $\mathcal{X}^2$-statistic of the partitions S_1 and S_2 of the feature induced by the class entropy. Some other refinements include information gain measure and information gain ratio measure that are used respectively in ID3 [96] and C4.5 [97] to induce the splitting node of a decision tree.

Information gain and information gain ratio

Information gain is simply the expected reduction in entropy by partitioning the samples according to this feature, that is the amount of information gained by looking at the value of this feature. More precisely, the information gain $Gain(f, S)$ of a feature f, relatively to a set of samples S, is defined as

$$Gain(f, S) \equiv Ent(S) - Ent(f, T_f, S) \tag{3.23}$$

where $Ent(S)$ can be calculated from equation (3.14) and $Ent(f, T_f, S)$ is the class entropy of the feature (for a numeric feature f, T_f is the best partition to f's value range under certain criteria, such as MDL principle). Since $Ent(S)$ is a constant once S is given, the information gain and entropy measures are equivalent when evaluating the relevance of a feature. In contrast to the rule "the smaller the class entropy value, the more important the feature is" that is used in entropy measure, we consider a feature f_j to be more relevant than a feature f_l $(l \neq j)$ if $Gain(f_j, S) > Gain(f_l, S)$. In fact, the ID3 [96] decision tree induction algorithm uses information gain as the measure to pick discriminatory features for tree nodes. Besides, information gain is also involved in some recent studies of feature selection on biological data. For examples, Xing *et al* [136] used it as one filter to select genes from gene expression data and the winner of KDD Cup 2001 [25] also employed it as a measurement to reduce the dimensionality of a feature space containing 139,351 binary features in a thrombin data set provided by Dupont Pharmaceuticals Research Laboratories.

However, there is a natural bias in the information gain measure — it favors features with many values over those with few values. An extreme example is a feature having different values in different samples. Although the feature perfectly separates the current samples, it is a poor predictor on subsequent samples. One refinement measure that has been used successfully is called *information gain ratio*. The gain ratio measure penalizes features that with many values by incorporating amount of *split information*, which is sensitive to how broadly and uniformly the feature splits the data [78]:

$$SplitInformation(f, S) \equiv -\sum_{i=1}^{w} \frac{|S_i|}{|S|} * log_2 \frac{|S_i|}{|S|} \tag{3.24}$$

where S_1 through S_w are the w subsets of samples resulting from partitioning of S by w-valued discrete or w-value-intervaled numeric feature f. Then, the *gain ratio* measure is defined in terms of the earlier information gain measure and this split information, as follows:

$$GainRatio(f, S) \equiv \frac{Gain(f, S)}{SplitInformation(f, S)} \tag{3.25}$$

Note that split information is actually the entropy of S with respect to the values of feature f and it discourages the selection of features with many values [78]. For example, if there are total number of n samples in S, the split information of a feature f_1, which has different values in different samples, is $log_2 n$. In contrast, a boolean feature f_2 that splits the same n samples exactly in half will have split information of 1. If these two features produce the equivalent information gain, then clearly feature f_2 will have a higher gain ratio measure. Generally, a feature f_j is considered to be more significant than a feature f_l $(l \neq j)$ if $GainRatio(f_i, S) > GainRatio(f_l, S)$. When using gain ratio measure (or information gain measure) to select features, we sort the values of gain ratio (information gain) in an descending order and consider those features with highest values.

3.2.5 Principal components analysis

Principal components analysis (PCA) [53] is widely used in signal processing, statistics and neural computing. It selects features by transforming a number of original (high-dimensional) features into a smaller number of uncorrelated features called principal components. The first principal component accounts for as much of the variability in the data as possible, and each succeeding component accounts for as much of the remaining variability as possible. The mathematical technique used in PCA is called eigen analysis [2]. The eigenvector associated with the largest eigenvalue has the same direction as the first principal component; the eigenvector associated with the second largest eigenvalue determines the direction of the second principal component, and so on.

Feature selection through PCA can be performed in following several steps.

(1) Calculating the covariance matrix C of a data collection X defined in Equation (3.1), where X is a matrix with n rows (i.e. samples) and m columns (i.e. features). Each column data of X may have to be normalized. Each element c_{ij} $(i, j = 1, 2, \cdots, m)$ of matrix C is the linear correlation coefficient between the elements of columns (i.e. features) i and j of X and is calculated as:

$$c_{ij} = \frac{1}{n}\sum_{k=1}^{n}(\frac{x_{ki}-\mu_i}{\sigma_i})(\frac{x_{kj}-\mu_j}{\sigma_j}) \tag{3.26}$$

where x_{ki} (x_{kj}) is the element k in column i (j) of X, and μ_i (μ_j) and σ_i (σ_j) are the mean and standard derivation of column i (j) of X, respectively. It is easy to prove that the covariance matrix C is real and symmetric.

(2) Extracting eigenvalues λ_i $(i = 1, 2, \cdots, m)$ by equation,

$$|C - \lambda_i I| = 0 \tag{3.27}$$

where I is an identity matrix.

(3) Computing eigenvectors e_i $(i = 1, 2, \cdots, m)$, which are the so-called “principal components”, from

$$(C - \lambda_i I)e_i = 0 \tag{3.28}$$

(4) Ranking eigenvectors according to the amount of variation in the original data that they account for, which is given by

$$Variance_i = \frac{\lambda_i}{\sum\limits_{k=1}^{m}\lambda_k} \tag{3.29}$$

(5) Selecting features that account for most of the variation in the data. In this step, eigenvectors (i.e. principal components) that account for some percentage (for example: 95%) of the variance in the original data will be chosen while the rest features will be discarded.

Indeed, it can be proven that the representation given by PCA is an optimal linear dimension reduction technique in the mean-square sense [53]. It is worth noting that, different from other methods introduced in this chapter, PCA is an *unsupervised* (in contrast to *supervised*) feature selection method since it makes no use of the class attribute.

3.2.6 Correlation-based feature selection

All of the preceding measures evaluate features in terms of their individual relevance to separating samples in different classes. However, rather than ranking individual features, we can also scores the worth of subsets of features. *Correlation-based feature selection* (CFS) [44] is such a method which is built on the belief that "good feature subsets contain features highly correlated with the class, yet uncorrelated with each other". At the heart of the CFS algorithm is a subset evaluation heuristic that takes into account not only the usefulness of individual features for predicting the class, but also the level of inter-correlation among them [46].

CFS first calculates a matrix of feature-class and feature-feature correlations. Then a score of a subset of features is assigned using the following heuristic:

$$Merit_S = \frac{k\overline{r_{cf}}}{\sqrt{k + k(k-1)\overline{r_{ff}}}} \tag{3.30}$$

where $Merit_S$ is the heuristic merit of a feature subset S containing k features, $\overline{r_{cf}}$ is the average feature-class correlation, and $\overline{r_{ff}}$ is the average feature-feature inter-correlation. The numerator can be thought of as giving an indication of how predictive the subset of features are while the denominator indicates how much redundancy there is among them [46].

In order to apply Equation (3.30), it is necessary to calculate the correlation between features. In this step, CFS uses *symmetrical uncertainties* to estimate the degree of association between discrete features or between features and classes [44]. The formula (3.31) below measures the inter-correlation between two features or the correlation between a feature and a class which is in the range $[0, 1]$ (f_1 and f_2 are both presented features or one is feature, one is class).

$$r_{f_1 f_2} = 2.0 \times \left[\frac{H(f_1) + H(f_2) - H(f_1, f_2)}{H(f_1) + H(f_2)} \right] \tag{3.31}$$

where the numerator is the information gain between features and classes, $H(f)$ is the entropy of the feature f defined in (3.14). CFS starts from the empty set of features and uses the best-first-search heuristic with a stopping criterion of 5 consecutive fully expanded non-improving subsets.

The subset with the highest merit found during the search will be selected.

3.2.7 Feature type transformation

At the end of introduction on feature selection methods, there are several points that need to be addressed:

- Converting feature type from discrete to numeric. This kind of conversion will be useful for those algorithms that can only handle numeric features, such as t-test, signal-to-noise, PCA and so on. When dealing with a k-valued discrete feature, one can convert it to k binary features. Each of these new features has a "1" for every occurrence of the corresponding kth value of the original discrete feature, and a "0" for all other values [46]. Then the new binary features are treated as numeric features.

- Converting feature type from numeric to discrete. Some feature selection methods, such as $\mathcal{X}^2$-statistic measure, need numeric features to be discretized. Fayyad's algorithm described in Section 3.2.4 or other discretization methods have to be applied.

- Dealing with multiple classes problem. If a data set contains more than two class samples, a pairwised feature selection has to be conducted.

3.3 ERCOF: Entropy-based Rank sum test and COrrelation Filtering

In this section, we will put forward a new strategy to conduct feature selection, mainly aiming to find significant genes in supervised learning from gene expression data. In our strategy, we combine the above presented methods of entropy measure and Wilcoxon rank sum test, as well as Pearson correlation coefficient test together to form a three-phase feature selection process. We name this combined feature selection process as *ERCOF* — stands for Entropy-based Rank sum test and COrrelation Filtering.

In phase I, we apply Fayyad's entropy-based discretization algorithm described in Section 3.2.4 to all the numeric features. We will discard a feature, if the algorithm can not find a suitable cut point to split the feature's value range. One point needs to be emphasized here is that we will use numeric features all the way, though a discretization algorithm is involved to filter out some features in this phase.

In phase II, we conduct Wilcoxon rank sum test only on features output from phase I. For a feature f, the test statistical measure $w(f)$ can be calculated by the way described in (3.2.2). If $w(f)$ falls outside the interval $[c_{lower}, c_{upper}]$, where c_{lower} and c_{upper} are the lower and upper critical test values that given in Formula (3.11), we will reject the null hypothesis and this indicates that the values of feature f are significantly different between samples in different classes. In the calculation of the two critical values c_{lower} and c_{upper}, the standard 5% or 1% significant level is generally used. Therefore, by this phase, we are left with two groups of features: one group contains features f_1 such that $w(f_1) < c_{lower}$, the other group contains features f_2 such that $w(f_2) > c_{upper}$. Features in same group are supposed to have similar behavior — having relatively larger values in one class of samples and relatively smaller values in another class of samples. In a gene expression data analysis, it is of a great interest to find which genes are highly expressed in a special type of samples (such as tumor samples, or patients with certain disease).

In phase III, for each group of features, we examine correlations of features within the group. For those features that are in the same group and are highly correlated, we select only some representatives of them to form the final feature set. In gene expression study, high correlation between two genes can be a hint that the two genes belong to the same pathway, are co-expressed or are coming from the same chromosome. “In general, we expect high correlation to have a meaningful biological explanation. If, e.g. genes A and B are in the same pathway, it could be that they have similar regulation and therefore similar expression profiles” [51]. We propose to use more uncorrelated genes for classification since if we have lots of genes from one pathway, the classification result might be skewed.

Since with entropy measure, one is more likely to select all the genes in a primary pathway and neglect those of secondary pathways, we have to try to sort out the genes that passed Phase I and Phase II filterings into pathways. Currently, we adopt the commonly used Pearson correlation coefficient to measure the correlation between features. It has been applied to analyse gene expression data by some researchers [16, 40]. Pearson correlation coefficient (also known as the centred Pearson correlation coefficient) is a linear correlation metric. In general, the Pearson correlation coefficient between any two features $f_i = (x_{1i}, x_{2i}, \cdots, x_{ni})$ and $f_j = (x_{1j}, x_{2j}, \cdots, x_{nj})$ (defined in (3.1)), $r(f_i, f_j)$, is given as:

$$r(f_i, f_j) = \frac{1}{n} \sum_{k=1}^{n} (\frac{x_{ki} - \mu_i}{\sigma_i})(\frac{x_{kj} - \mu_j}{\sigma_j}) \tag{3.32}$$

where μ_i (μ_j) and σ_i (σ_j) are the mean and standard derivation of f_i (f_j), respectively. The value of r is between -1 and 1. In our case, we just consider two features to be correlated if their correlation coefficient is 1 or near 1 and ignore negative correlations since the features in same group are expected to have similar behavior. A threshold r_c of r is set in advance, so that if $r(f_i, f_j) > r_c$, then feature f_i and f_j are considered correlated.

Given a group of features, we subgroup features in this group based on correlation coefficient. First, we sort the features according to their class entropy measure in an ascending order (i.e., with best feature at first position). Then we pick up the best feature f_1, and calculate its Pearson correlation coefficient with all other features. Then we form a subgroup consisting of f_1 and all features that are correlated to f_1. The features that have been assigned to this subgroup are not considered again in the later rounds of correlation test. In the second round of subgrouping, we pick up the best one from remaining features, and form another subgroup of features. This correlation test proceeds until all the features in the group have been assigned to a subgroup. Note that it is possible for a subgroup to have only one feature. So, the groups of features are sub-grouped; in each subgroup, features are all correlated to a best feature such as f_1. Figure 3.3 gives the pseudo codes of this method.

Next, we select representative features from each subgroup to form the final feature set. In each subgroup, since the features are sorted by their class entropy measure, we calculate the average of the entropy values of all these features (named *mean entropy value* of this subgroup) and choose those top ones whose entropy measure is smaller than this mean entropy value. In case of only one feature in a subgroup, this feature is automatically selected. These representative features from all the subgroups are our final set of features. See Figure 3.4 for a whole picture of feature identification and selection by ERCOF.

Using ERCOF in gene expression data analysis where there is often more than thousands of features, we expect to identify of a subset of *sharply discriminating* features with *little redundancy*. The entropy measure is effective for identifying discriminating features. After narrowing

```
1. $k = 1$.
2. Rank all features in group $F$ on class entropy in an ascending order, $f_1, f_2, \cdots, f_l$.
3. Let $S_k = \{f_1\}$ and remove $f_1$ from $F$.
4. For each $f_i (i > 1)$
     calculate Pearson correlation coefficient $r(f_1, f_i)$;
     if $r(f_1, f_i) > r_c$
        add $f_i$ into $S_k$ and remove it from $F$;
5. $k = k + 1$ and goto step 2 until $F = \emptyset$.
```

Figure 3.3: Feature subgrouping by correlation testing. r_c is the Pearson correlation coefficient threshold, which should be near 1.0.

down by the Wilcoxon rank some test, the remaining features become sharply discriminating. Then, with the correlation examination, some highly correlated features are removed to reduce redundancy. We do not use CFS introduced in Section 3.2.6 in Phase III of ERCOF, because CFS sometimes returns too few features to comprehensively understand the data set. For example, CFS selects only one feature if the class entropy of this feature is zero. However, Pearson correlation coefficient also has a shortcoming — the calculation of correlation is dependent on the real values of features — it is sensitive to some data transformation operations. Therefore, other algorithms are being implemented to group correlated features.

3.4 Use of Feature Selection in Bioinformatics

The feature selection techniques reviewed in the preceding sections have been used as a key step in the handling of high-dimensional biomedical data. For example, their use is prevalent in the analysis of microarray gene expression data (an extensive review on this can be found in Chapter 4). Besides, they have been also used in the prediction of molecular bioactivity in drug design [132], and more recently, in the analysis of the context of recognition of functional site in DNA sequences [142, 72, 69].

One issue should be addressed here is the so-called "multiple comparisons problem" [85] which happens when we select features by choosing a statistical confidence level (like standard 5% or 1%) for t-test, $\mathcal{X}^2$-test, and other statistical measures. The description of the problem is: when performing m multiple independent significance tests, each at the α level, the probability of making at least one Type I error (rejecting the null hypothesis inappropriately) is $1-(1-\alpha)^m$.

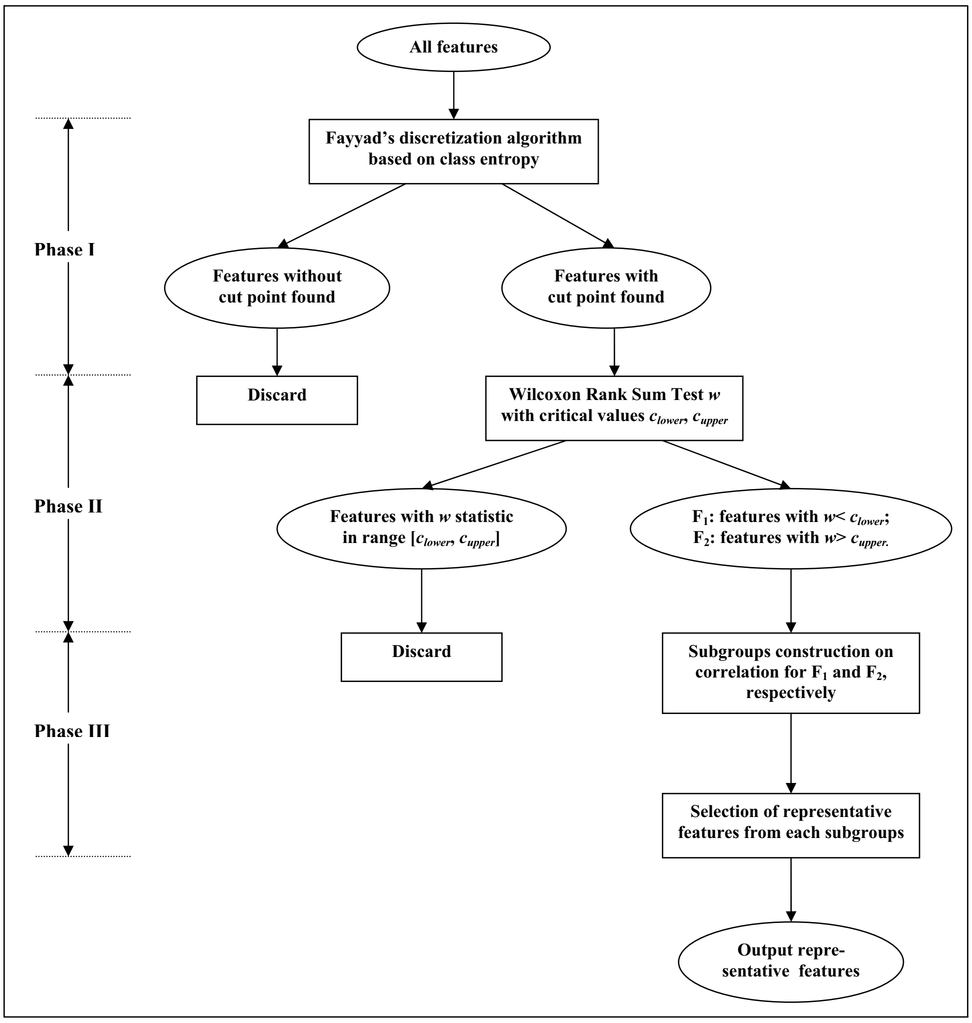

All features
Fayyad's discretization algorithm based on class entropy
Phase I
Features without cut point found
Features with cut point found
Discard
Wilcoxon Rank Sum Test w with critical values c_{lower}, c_{upper}
Phase II
Features with w statistic in range [c_{lower}, c_{upper}]
F_1: features with $w < c_{lower}$; F_2: features with $w > c_{upper}$.
Discard
Subgroups construction on correlation for F_1 and F_2, respectively
Phase III
Selection of representative features from each subgroups
Output representative features

Figure 3.4: A diagram of ERCOF: Entropy-based Rank sum test and COrrelation Filtering, a three-phase feature selection process combining concepts of entropy, Wilcoxon rank sum test and Pearson correlation coefficient.

For example, suppose we consider $m = 200$ features and perform independent statistic tests to each of them at the standard $\alpha = 5\%$ level, then the probability of getting at least one significant result is $1 - 0.95^{200} = 0.99996$ [85]. So, when we get a significant feature among the tests, how can we believe that it is "indeed" significant. In fact, under this setting, we would still expect to observe approximately 10 ($= 200 * 0.05$) "significant" features, even when there were actually no features that can distinguish the two classes. Obviously, the problem becomes serious when the total number of considered features is large, which is the case in some biological data such as gene expression profilings.

A standard conservative solution to this problem is the Bonferroni correction [100], which divides the test significant level by the number of tests, i.e. α/m. In the above example, it will be $0.05/200 = 0.00025$. Thus, for 200 features, the cutoff for significance would be 0.00025 instead of previous 0.05! In spite of its simplicity, the Bonferroni method has some shortcomings [91]. The biggest problem is that it is too conservative: each individual test is held to an unreasonably high standard and this will increase the probability of a Type II error where legitimate signal features will fail to be discovered. On the other hand, the method is applicable only to tests with known statistical distributions. For measures with unknown statistical distribution, permutation-based approaches are practically used .

In a permutation-based method, the adjusted significant level (also known as p-values) based on the number of tests undertaken is also computed, but in a way less conservative than the Bonferroni method. When conducting permutation, we assume that there is no relationship between features and classes so that new samples can be drawn by reassigning permuted class labels to original data. The p-value then can be calculated based on the feature statistics on many these kind of pseudo data sets. However, the conclusion that we really want to draw from the permutation test might be: *if we have selected w features using a particular statistic, what proportion of these features are likely to be false positive*? To make such a conclusion, one can follow the steps illustrated in Figure 3.5 for a testing at α level. Alternatively, in stead of single cutoff value, we can set up a series of thresholds and compute the p-value for each of them based on the permutation test, so that a table T of threshold versus p-value can be created. If we want no more than $q\%$ of the features selected in the original (non-permuted) experiment to be false positive, then we should look up table T using q for the threshold α_q and use α_q as the statistic threshold to pick up features from original experiment.

Although the permutation is designed to take the place of the Bonferroni correction, it is often found that the critical values determined in this method are nearly as conservative as those based on the Bonferroni adjustment [85]. However, it has no assumption on the distribution of the selected test statistic. As indicated earlier, another critical consideration of the permutation test is that the procedure does not address whether features are correlated. In the case of a large number of strongly correlated features versus a relative small number of samples, the test statistic on each permutation will not significantly change. Then the permutation becomes meaningless.

```
1. Select a statistic which will be used to measure differences between classes.
2. Determine the threshold of the statistic according to significant level $\alpha$.
3. Calculate the test statistic for each of total $m$ features
4. Get the number of features selected by the threshold, record as $w$.
5. For $i$th permutation test iteration ($i = 1, 2, \cdots, t$):
     generate a pseudo data set by randomly permuting the class labels of all the samples,
     calculate the same test statistic for every feature,
     record how many features are selected by the threshold, denote it as $k_i$.
6. Compute the percentage of features selected during the permutation test, $p = \frac{\sum_{i=1}^{t} k_i}{t \times m}$
calculate $p \times w$ to be the expected number of false positive.
```

Figure 3.5: A diagram of a permutation-based method for feature selection. In practice, the significant level α is often set as 5% or 1%, the permutation times t should be very large, say 10,000 times, or for all possible permutations of the class labels.

Unfortunately, in many biological domain, features have strong correlations from sample to sample.

3.5 Chapter Summary

In this chapter, we reviewed feature selection techniques for data mining. There are two broad categories of selection algorithms, filter and wrapper, and we indicated that filter approaches are more suitable to be applied to solve biological problems. We presented a variety of filter methods, such as t-statistic measure, Wilcoxon rank sum test, entropy-based measures, principal components analysis and so on. We also put forward a new feature selection strategy, ERCOF, which is a 3-phase feature filtering process aiming to identify a subset of sharply discriminating features with little redundancy from gene expression profiles. The chapter was ended with a discussion on using feature selection in bioinformatics.

Chapter 4

Literature Review on Microarray Gene Expression Data Analysis

One of the important recent breakthroughs in experimental molecular biology is microarray technology. This novel technology allows the monitoring of expression levels in cells for thousands of genes simultaneously and has been increasingly used in cancer research [7, 41, 6] to understand more of the molecular variations among tumors so that a more reliable classification becomes possible.

There are two main types of microarray systems [35]: the cDNA microarrays developed in the Brown and Botstein Laboratory at Stanford [32] and the high-density oligonucleotide chips from the Affymetrix company [73]. The cDNA microarrays are also known as spotted arrays [77], where the probes are mechanically deposited onto modified glass microscope slides using a robotic arrayer. Oligonucleotide chips are synthesized in silico (e.g., via photolithographic synthesis as in Affymetrix GeneChip arrays). For a more detailed introduction and comparison of the biology and technology of the two systems, please refer to [47].

Gene expression data from DNA microarrays are characterized by many measured variables (genes) on only a few observations (experiments), although both the number of experiments and genes per experiment are growing rapidly [82]. The number of genes on a single array is usually in the thousands while the number of experiments is only a few tens or hundreds. There are two different ways to view data: (1) data points as genes, and (2) data points as samples (e.g. patients). In the way (1), the data is presented by expression levels across different samples, thus there will be a large number of features and a small number of samples. In the way (2), the data is represented by expression levels of different genes, thus the case will be a large number of samples with a few attributes. In this thesis, all the discussions and studies on gene expression profiles are based on the first manner of data presentation.

Microarray experiments raise many statistical questions in many diversified research fields, such as image analysis, experimental design, cluster and discriminant analysis, and multiple hypothesis testing [35]. The main objectives of most microarray studies can be broadly classified into one of the following categories: class comparison, class discovery, or class prediction [77].

- *Class comparison* is to establish whether expression profiles differ between classes. If they do, what genes are differentially expressed between the classes, i.e. *gene identification*. For example, which genes are useful to distinguish tumor samples from non-tumor ones.
- *Class discovery* is to establish subclusters or structure among specimens or among genes, for example, to define previously unrecognized tumor subtypes [41, 140].
- *Class prediction* is to predict a phenotype using information from a gene expression profile [77]. This includes assignment of malignancies into known classes (tumor or non-tumor) or tumor samples into already discovered subtypes, prediction of patients outcome such as which patients are likely to experience severe drug toxicity versus who will have none, or which breast cancer patients will relapse within five years of treatment versus who will remain disease free. Figure 4.1 shows a work flow of class prediction.

In this thesis, we will focus on the class comparison and class prediction. For these two tasks, supervised analysis methods that use known class information are most effective [77]. In practice, feature selection techniques are used to identify discriminatory genes while classification algorithms are employed to build models on training samples and predict the phenotype of blind test cases.

4.1 Preprocessing of Expression Data

As with most of the data fed to machine learning algorithms, gene expression data also need necessary preprocessing before being further analysed. Based on the characteristics of the exper-

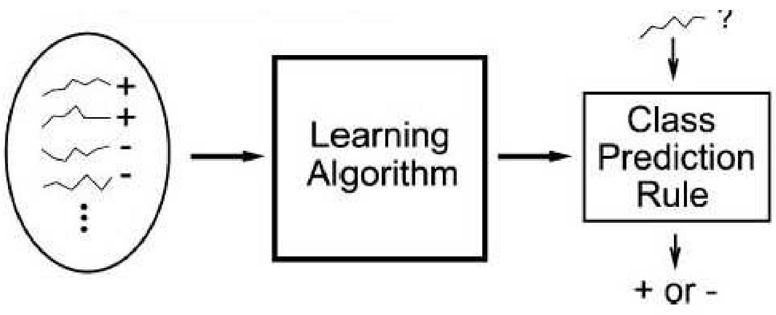


Figure 4.1: A work flow of class prediction from gene expression data. A collection of expression profiles with known class label (+ or -) is the input of a supervised learning algorithm. After being trained on these profiles, the prediction model built by the learning algorithm will be able to predict the class label of a new case of expression profile. The picture is captured from [77].

imental data, the normal preprocessing steps include scale transformation, data normalization, missing value management, replicate handling and so on [49].

4.1.1 Scale transformation and normalization

In cDNA microarray experiments utilizing “spotted arrays”, the two mRNA samples, known as targets, are reverse transcribed into cDNA (labeled using two different fluorophores — usually a red fluorescent dye cyanine 5 and a green fluorescent dye cyanine 3), and mixed in equal proportions and hybridized simultaneously to the glass slide [35]. Intensity values generated from hybridization to individual DNA spots are indicative of gene expression levels. Then the ratio of the red and green fluorescence for each spot is used to measure the change between samples. In order to accurately and precisely measure gene expression changes, it is important to understand sources of variance in expression data. In every microarray experiment, experimental randomness and systematic variations [139] are the two main sources of variance. For example, a well-known systematic variation originates the biases associated with the different fluorescent dyes. If two identical mRNA samples are labeled with different dyes and hybridized to the same slide, it is rare to have the dye intensities equal across all spots between these two samples [139].

Since we are looking at expression ratios, we expect the patterns in an asymmetrical scale: over-expressions will have values between 1 and infinite while under-expression will between 0 and 1. In order to give the same weight to both over-expressions and under-expressions, we need to transform the scale. A simple and common way is to do log-transformation. Normally this is done by taking log_2 of the ratio, such as $log_2(Cy5/Cy3)$. Besides, considering data in log-space can also help reduce the effects of outliers [85].

In order to minimize systematic variations in gene expression levels of two co-hybridized mRNA samples, normalization should be conducted for spotted cDNA microarrays. This will help easily distinguish biological differences between two samples and make the comparison of expression levels across slides reasonable. There are several ways to conduct normalization. For example, in one of the general methods, the intensity values are normalized according to the formula: $NV = (V - Min)/(Max - Min)$, where NV is the normalized value, V the raw value, Min (Max) the minimum (maximum) intensity among all samples for the gene. After the normalization, each intensity value is to fall within the range of 0 to 1. Another common practice is to center the data by the median or mean ratio, and possibly to scale the data by the standard deviation [85]. Recently, Yang *et al* proposed a composite normalization procedure in [139], based on robust local regression, to account for intensity and spatial dependence in dye biases for different types of cDNA microarray experiments. They constructed a novel control sample named MSP including all genes present on the microarray, and titrated it over the intensity range of a microarray experiment. Under the composite idea, low intensity values will be normalized based on all genes in the corresponding intensity range while higher values will be normalized based on the MSP titration series.

When we illustrate our work on some gene expression profilings one by one in the next chapter, we will indicate whether a preprocessing (log-transformation, normalization and so on) has been conducted on a particular data set. However, basically, as stated in [85], the normalization is not technically required, though it will help reduce the effects of varying dynamic range from sample to sample for cDNA microarray data.

4.1.2 Missing value management

One of the characteristics of the gene expression profile is the existence of missing values in the data set. There are diverse reasons that cause missing values, including insufficient resolution, image corruption, or simply due to dust or scratches on the slide [125]. In practice, missing data also occur systematically as a result of the robotic methods used to create them. Unfortunately, many data analysis algorithms require a complete matrix of gene array values as input [125]. For example, standard hierarchical clustering methods and k-means clustering are not robust to the excess of missing values since the calculations in the algorithms are based on a distance matrix. Even with a few missing values, they may lose effectiveness. More strictly, some methods like principal components analysis can not deal with missing values at all. Therefore, methods for imputing missing data are needed, not only to minimize the effect of incomplete data on further analyses, but also to increase the range of data sets to which learning algorithms will be applied.

There are some general solutions to impute missing values, though there is not a large literature that were specific to gene expression data. Here, we list four commonly used strategies: (1) filling blanks with zeros; (2) replacing with the gene's average expression levels over all experiments; (3) replacing with the median of the gene's expression levels over all experiments; (4) using weighted k-NN imputation method. The k-NN-based method is to use the k-nearest neighbours to estimate the missing values, where k is a user-defined parameter. The selection of "neighbours" can be done via calculating certain similarity metric between genes, such as widely used Euclidean distance, Pearson correlation, variance minimization and etc [125]. For example, if gene A has one missing value in experiment 1, the k-NN-based method will find k other genes, which have a value present in experiment 1 and have most similar expression values to A in other experiments. The values of these k nearest genes in experiment 1 are then averaged by a weight metric and used as the estimated value of gene A in experiment 1. In the weighted average, the contribution of each gene is weighted by similarity of its expression levels to gene A.

Troyanskaya *et al* [125] compared three missing value imputation methods by testing them on three microarray data sets. Three imputation methods were simple gene average, weighted k-NN and their proposed singular value decomposition (SVD) based method. The mechanism of SVD-based algorithm is to (1) use singular value decomposition to obtain a set of mutually orthogonal expression patterns that can be linearly combined to approximate the expression of all genes in the data set, (2) refer these patterns as eigengenes (like principal components) and select k most significant eigengenes by sorting their corresponding eigenvalue, (3) estimate a missing value in gene A by regressing gene A against the k eigen genes and then use the coefficients of the regression to reconstruct a replacement value from a linear combination of the k eigen genes. Their results showed that weighted k-NN appeared to be the most accurate and robust method, and both weighted k-NN and SVD-based techniques surpass the commonly used simple average method. This conclusion is very natural since the winning methods take advantage of the correlation structure of the data to estimate missing expression values.

Although we can efficiently handle missing values in microarray data by using weighted k-NN imputation method, the method itself requires that we have enough complete genes (clones) (i.e. genes with no missing values) in the data set so that finding real neighbours can be ensured. When there are too many missing values in an original data set, one can consider to filter some genes based on amount of missing elements. For example, in a study on diffuse large-B-cell lymphoma addressed in [60], genes (clones) having more than 20% missing values were removed before any analysis being conducted. Please note that, in [6, 140], the missing values in the gene expression data sets were excluded in the analyses.

4.1.3 A web-based preprocessing tool

An interactive web-based software for preprocessing microarray gene expression data was introduced in [49], which was implemented in a Perl CGI script. Besides the functions mentioned above, such as log-transformation, normalization and missing values management, it also provides a way to handle replicate. The replicate here means the same cDNA clone that spotted several times or different cDNAs representing the same gene on the cDNA array. The usage of replicates is mainly for quality checking. Generally, in an experiment, several expression values of a replicated gene will be output, though only one is needed in the further analysis. How to derive a proper expression level from several output values? The provided solution is quite simple: using the average or the median value of all the replicates upon checking the consistency among them. During the consistency checking, the median of all the values is calculated and then the replicates whose expression value is beyond the threshold from the median are removed. The threshold is a user-defined value. The web interface of this tool is at `http://gepas.bioinfo.cnio.es/cgi-bin/preprocess`.

4.2 Gene Identification and Supervised Learning

Supervised learning algorithms are used to establish models to classify samples in different classes of gene expression profiles while gene identifications answer which genes are differentially expressed between the classes, i.e. feature selection. Generally, gene identification is carried out before learning algorithms are used.

4.2.1 Gene identification

In a pioneer study in 1999, Golub *et al* [41] analysed gene expression profiles of 27 Acute Lymphoblastic Leukemia (ALL) samples and 11 Acute Myeloid Leukemia (AML) samples. They identified genes with differential expression between ALL and AML samples using the signal-to-noise measure that we introduced in Section 3.2.1 of Chapter 3. According to signal-to-noise statistic, the coefficient correlation between gene g and classes, $s(g)$, is defined as:

$$s(g) = \frac{\mu^{\mathcal{A}}(g) - \mu^{\mathcal{B}}(g)}{\delta^{\mathcal{A}}(g) + \delta^{\mathcal{B}}(g)} \tag{4.1}$$

where $\mu(g)$ and $\delta(g)$ are the mean and standard deviation of the gene expression values for gene g for all the patients of class $\mathcal{A}$ (ALL) or class $\mathcal{B}$ (AML). Large positive value of $s(g)$ indicates strong correlation with class $\mathcal{A}$ whereas large negative value of $s(g)$ indicates strong correlation with class $\mathcal{B}$ [41]. Then an equal number of genes with positive and with negative correlation values were selected to integrate into the learning algorithm. The number of informative genes they chose was 50, but they stated in the paper that "the (prediction) results were insensitive to the particular choice: predictors based on between 10 and 200 genes were all found to be 100% accurate, reflecting the strong correlation of genes with the AML-ALL distinction".

Similar to Golub *et al*, there were some other researchers who used statistical tools to discover differentially expressed genes between sample classes, such as t-statistic and its variation (like signal-to-noise, Fisher criterion score), Wilcoxon rank sum test and so on. For examples, in [12], genes selected by t-statistic were fed to a Bayesian probabilistic framework for sample classification. Olshen *et al* [85] suggested to combine t-statistic, Wilcoxon rank sum test or the $\mathcal{X}^2$-statistic with a permutation-based model to conduct gene selection. In their model, the significance of genes is determined by the associated statistic and a critical value calculated on the same statistic using the permuted labels. The permutation of sample class labels were conducted for a few thousands times. Wilcoxon rank sum test is another measure favored by researchers mainly due to its non-parametric characteristic.

Park *et al* built a scoring system in [87] to assign each gene a score based on training samples. For a gene, they first sorted training samples according to the expression levels of this gene — from the smallest expression level to the largest one. Second, they swap the class labels of the samples to make the gene into a perfectly discriminating marker — all high expression values belong to one class of samples and all low expression values belong to the other class. Then the score of the gene was the minimum number of the necessary swaps. Finally, a small set of differently expressed genes, which had smaller score, were discovered. They claimed that this scoring approach was robust to outliers and different normalization schemes because it used ranks rather than actual expression levels. Essentially, this score is identical to Wilcoxon rank sum test statistic [51]. Some researchers also conducted comparisons between Wilcoxon rank sum test and some other statistical measures on gene selection. For example, Troyanskaya *et al* compared t-statistic, Wilcoxon rank sum test and a heuristic method based on Pearson correlation in [126]. Their results showed that overall speaking, the rank sum test appeared most conservative, which may be advantageous if the further biological or clinical usages of the identified genes are taken into account.

Jaeger *et al* [51] designed three pre-filtering methods to retrieve groups of similar genes. Two of them are based on clustering and one is on correlation. A statistical test then was applied to these groups to finally select genes. The statistical tests used in their study included Fisher criterion score, signal-to-noise, Wilcoxon rank sum test, t-statistic and TnoM (Thresholded-number-of-Misclassifications), which calculates a minimal error decision boundary and counts the number of misclassifications done with this boundary. Based on the test results on three public gene expression data sets using the selected genes and support vector machines classification algorithm, they concluded that feature selection can greatly help improve the classification accuracy, but there is no absolute winner among their proposed pre-filtering methods and the five statistical tests. Another comparison of using different statistics in gene identification was conducted by Thomas *et al* in [121], they presented a statistical regression modeling approach to discover genes that are differentially expressed between two classes of samples. Their modeling approach used known sample group membership to focus on expression profiles of individual genes. They tested their methodology on the AML-ALL leukemia data set of Golub [41] and compared their results with those obtained using t-statistic or Wilcoxon rank sum test. Their model made no distributional assumptions about the data and accounted for high false-positive error rate. However, in practice, the Z-scores they proposed are expected to be similar to t-

statistics, when the distribution of expression levels can be approximated by the normal distribution. In a recent review of several statistical methods in term of their effectiveness to discover differentially expressed genes, Pan [86] compared t-statistic, the regression modeling approach against a mixture model approach proposed by him. Different from t-statistic and the above regression modeling approach that sets strong assumptions on the null distribution of the test statistics, the mixture model estimated the null distribution directly. He pointed out that although the three methods were all based on using the two-sample t-statistic or its minor variations, they differed in how to associate a statistical significance level to the corresponding statistic so that large differences in the resulting significance levels and the numbers of genes discovered were possible [86]. The Bonferroni method described in Section 3.4 of Chapter 3 was used in his study to adjust the significant level.

SAM (Significance Analysis of Microarrays), a software developed at Stanford University (`http://www-stat.stanford.edu/˜tibs/SAM/`), is designed to find significant genes in a set of microarray experiments based on strong statistical study on genes [127]. SAM first computes a statistic to each gene on the basis of change in gene expression relative to the standard deviation of repeated measurements for the gene. Then for those genes whose statistic is greater than an adjustable threshold, SAM uses permutations of the data to estimate the percentage of genes identified by chance (known as false discovery rate (FDR)). The threshold for significance is determined by a *tuning parameter* δ, chosen by the user based on FDR, or a *fold change* parameter to ensure that the selected genes change at least a pre-specified amount [26]. Besides gene expression profiles for phenotype classification, SAM can be applied to other types of experimental data [127]. For example, to identify genes whose expression correlates with survival time, the assigned score is defined in terms of Cox's proportional hazards function, which is a popular method for assessing a covariate's effect on patients remain alive or censored during the follow-up at the time of the study. To identify genes whose expression correlates with a quantitative parameter (e.g. a numeric type class label), such as tumor stage, the assigned score can be defined in terms of the Pearson correlation coefficient.

Besides statistical measures, other dimension reduction methods were also adopted to select genes from expression data. Nguyen *et al* [82] proposed an analysis procedure for gene expression data classification, involving dimension reduction using partial least squares (PLS) and classification using logistic discrimination (LD) and quadratic discriminant analysis (QDA). They compared PLS to the well known dimension reduction method of principal components analysis (PCA). PCA reduced the high dimensional data to only a few gene components which explained as much of the observed total gene expression variation as possible and PLS chose components to maximize the sample covariance between the class and a linear combination of the genes. The essential difference between these two methods is that PLS is a supervised method while PCA is an unsupervised method since it selects features without regard to the class information of the samples. For more about PCA, please refer to Section 3.2.5 in Chapter 3. After applying PLS to original high dimension data, a simple t-statistics was used to conduct a further gene selection. Finally, 50 genes were provided to the classification step.

4.2.2 Supervised learning to classify samples

Various machine learning algorithms have been applied to conduct classification from gene expression data. Let's still start with the AML-ALL leukemia study conducted by Golub *et al* in [41]. The classification method they proposed was a weighted gene voting scheme, which was a combination of multiple "univariate" classifiers [43]. In detail, they defined $a_g = s(g)$ (reflects the correlation between the expression levels of gene g and distinction), and $b_g = [\mu^{\mathcal{A}}(g) + \mu^{\mathcal{B}}(g)]/2$ (the average of the mean expression values in the two classes). The $s(g)$ was the signal-to-noise measure of gene g that they used to select genes. When doing prediction for a new sample T, let t_g denote the expression value of gene g in the sample. The vote of gene g was $V_g = a_g * (t_g - b_g)$, with a positive value indicating a vote for class $\mathcal{A}$ and a negative value indicating a vote for class $\mathcal{B}$. The total vote for class $\mathcal{A}$ was obtained by adding up the absolute values of the positive votes over the selected informative genes, while the total vote for class $\mathcal{B}$ was obtained by adding up the absolute values of the negative votes. In order to avoid arbitrary prediction when the margin of victory is slight, they defined "prediction strength" (PS) to measure the margin of a winner class. A threshold of PS was established to minimize the chance of making an incorrect prediction.

Dudoit *et al* [35] conducted a comparison of using some discriminant methods for classification of gene expression data. These well-known classification methods included Fisher linear discriminant analysis (FLDA), maximum likelihood discriminant rules (such as linear discriminant analysis (LDA), diagonal quadratic discriminant analysis (DQDA) and diagonal linear discriminant analysis (DLDA, also known as naive Bayes)), k-nearest neighbours (k-NN) classifier, classification and regression trees (CART) and aggregating CART trees by boosting procedure. Before classification, a gene filtering was performed based on the ratio of genes between-group to within-group sums of squares. For a gene j, this ratio, $BW(j)$, was given by

$$BW(j) = \frac{\sum_i \sum_k I(y_i = k)(\bar{x}_{kj} - \bar{x}_j)^2}{\sum_i \sum_k I(y_i = k)(x_{ij} - \bar{x}_{kj})^2} \tag{4.2}$$

where y_i was the class label of sample i and $I(\bullet)$ was an indicator function — equaling 1 if the condition in the following parentheses was true and 0 otherwise; $\bar{x}_j$ and $\bar{x}_{kj}$ were the average expression level of gene j across all the samples and across samples belonging to class k only [35]. Then a certain number of genes with the largest BW ratios were selected for classification. They did experiments on three data sets. Their results showed that k-NN classifiers and DLDA had the lowest error rates, whereas FLDA had the highest. CART-based classifiers performed intermediately, with aggregated classifiers being more accurate than a single tree. They explained that the poor performance of FLDA was most likely caused by the fact that data sets contained a large number of genes but a limited number of samples. Under such a situation, the ratios of between-group and within-group sums of squares and cross-products became quite unstable and provided poor estimates of the corresponding population quantities. They also showed that the performance of FLDA improved when the number of selected genes was decreased to 10. Although CART-based classifiers did not achieve the best performance, they could exploit and reveal interactions between genes as well as relationship between genes and phenotypes. Most importantly, decision trees/rules output by these methods are easy to interpret and understand. In addition, their results also demonstrated that the unstableness of a single classification tree on prediction could be greatly improved when it was used in combination with aggregation techniques.

As mentioned previously in Chapter 2, support vector machines (SVM) have been extensively used in biological data analysis. It is also playing a very active role in classifying gene expression data. SVM has many mathematical features that make it attractive for gene expression analysis, such as its flexibility in choosing a similarity function, sparseness of solution when dealing with large data sets, the ability to handle large feature spaces, and the ability to identify outliers [23]. For example, in an early work done by some researchers in MIT [80], a linear

SVM classifier with a rejection level based on confidence values was applied to classify Golub's AML-ALL subtypes leukemia disease. They achieved a better performance on this task than Golub *et al* did [41]. Furey *et al* [39] further tested the efficiency of SVM on several other gene expression data sets and also obtained good results. Both of them selected discriminatory genes via signal-to-noise measure.

Besides the above techniques, Bayes model, a classical and effective method, has been also applied to gene expression study. For example, two new Bayesian classification algorithms were investigated in Li *et al* [68] which automatically incorporated a feature selection process. The fundamental technique of the algorithms was a Baysian approach named automatic relevance determination (ARD), which was employed to construct a classifier that was sparse in the number of samples, i.e. the relevance vector machine (RVM). Li Y. *et al* [68] adopted the idea of ARD to gene expression study. They developed two algorithms. One was the standard RVM with sparsity obtained in the feature set. Another performed feature selection by isolating the feature dependence in the log-marginal likelihood function. The conclusion they obtained was that these algorithms had comparable performance to SVM when dealing with gene expression data.

4.2.3 Combing two procedures — wrapper approach

In some studies, procedures of gene selection and supervised learning were not separated distinctly. Similar to the wrapper approach illustrated in Chapter 3, identification of significant genes were incorporated with learning process. For example, Weston *et al* [131] integrated feature selection into the learning procedure of SVM. The feature selection techniques they used included Pearson correlation coefficients, Fisher criterion score, Kolmogorov-Smirnov test and generalization selection bounds from statistical learning theory. Going a step further, Guyon *et al* [43] presented an algorithm called recursive feature elimination (RFE), by which features were successively eliminated during the training of a sequence of SVM classifiers.

There are some other examples of using the wrapper idea in gene expression data analysis. Gene selection was performed in [50] by a sequential search engine, evaluating the goodness of each gene subset by a wrapper method. The method executed the supervised algorithm to obtain its accuracy estimation by a leave-one-out process. The supervised classification algorithms reported in this paper included IB1 (i.e. 1-NN), Naive-Bayes, C4.5 and CN2. The paper demonstrated that the accuracy of all these learning algorithms was significantly improved by using the gene selection procedure. Another example of using the wrapper method was [67], where Li *et al* combined a genetic algorithm (GA) and the k-NN method to identify a subset of genes that could jointly discriminate between different classes of samples. First, GA was used to obtain many such “near optimal” subsets of differentially expressed genes independently. Then, the relative importance of genes for sample classification were chosen by examining the frequency of membership of the genes in these sets.

Culhane *et al* [31] applied Between-Group Analysis (BGA) to microarray data. BGA was based on conducting an ordination of groups of samples, using a standard method such as correspondence analysis (COA) or principal components analysis (PCA). For N groups, BGA could find $N - 1$ eigenvectors (or axes) to maximize the between-group variance. Each of eigenvectors could be used as a discriminator to separate one of the groups from the rest. After a BGA, the samples are separated along axes. The genes that were most responsible for separating the groups were those with the highest or lowest coordinates along these axes. One advantage of BGA is that it can be safely used with any combinations of numbers of genes and samples so that no advanced gene selection is necessary.

PAM (Prediction Analysis for Microarrays), developed at Stanford University (`http://www-stat.stanford.edu/~tibs/PAM/`), is a class prediction software for genomic expression data mining. It performs sample classification from gene expression data based on the *nearest shrunken centroid* method proposed by Tibshirani *et al* [123]. This method computes a standardized centroid for each class — the average gene expression for each gene in each class divided by the within-class standard deviation for that gene. This standardization has the effect of giving higher weight to genes whose expression is stable within samples of the same class [123]. The main feature of nearest shrunken centroid classification from standard nearest centroid classification is that it ”shrinks” each of the class centroids toward the overall centroid for all classes by an amount named *threshold*. The selection of the threshold can be determined by the results of cross-validation for a range of candidate values. When classifying a new sample, it follows the usual nearest centroid rule, but using the shrunken class centroids. The idea of shrinkage has two advantages: (1) it achieves better performance by reducing the effect of noisy genes, and (2) it does automatic gene selection. In particular, if a gene is shrunk to zero for all classes, then it will be removed from further consideration. PAM has been applied to several DNA microarray data sets to do classification [123, 124], such as small round blue cell tumor data of childhood [54], diffuse large B-cell lymphoma [6], AML-ALL leukemia [41]. Recently, PAM is also used to classify patients into appropriate clinical subgroups (e.g. high risk and low risk groups) identified by clustering algorithms on gene expression profiles [11].

4.3 Applying Clustering Techniques to Analyse Data

Another early work on analyzing gene expression data was done by Alon *et al* [7]. Their data contained the expression of the 2000 genes with highest minimal intensity across 62 tissues, including 22 normal and 40 colon cancer. Their study was based on top down hierarchical clustering, a method of unsupervised learning. They demonstrated two kinds of groupings that (1) genes of related functions could be grouped together by clustering according to similar temporal evolution under various conditions, and (2) different tissues formed different clusters, i.e. most normal samples clustered together while most cancer samples clustered together. Although they showed that some genes are correlated with the normal versus cancer separation, they do not suggest a specific method of gene selection in the paper.

Since [7], quite a few researchers have applied clustering techniques to gene expression data, including self organizing maps, simulated annealing and graph theoretic approaches. In [111], the input data was represented as a weighted graph, where vertices corresponded to samples and edge weights reflected pairwise similarity between the corresponding samples. Then the weight of an edge was believed to reflect the likelihood that its endpoints originated from the same clustering under some simplified probabilistic assumptions [111]. An algorithm named CLICK (CLuster Identification Connectivity Kernels) was invented to partition the graph using a minimum-cut algorithm, which minimizes the sum of the weights of the edges joining the two parts. However, one disadvantage of this approach is that there is little guarantee that the algorithm will not go astray and generate partitions that are highly unbalanced. To avoid this, Xing *et al* [136] proposed CLIFF (CLustering via Interactive Feature Filtering) to combine clustering and feature selection in a bootstrap-like process. Their algorithm interacted between feature filtering process and clustering process in a such way that each process used the output of the other process as an approximate input. They applied Approximate Normalized Cut, a graph partition algorithm, to generate a dichotomy of samples during each iteration. In the feature selection process, they used the unsupervised independent feature modeling technique to rank all features in terms of their power to discriminate. Then an initial partition based on the k most discriminative features was generated (value k was pre-defined). Based on this partition, they applied supervised algorithms, information gain ranking and Markov blanket filtering, to determine feature subset from which new partition would be generated. In turn, the newly generated partition could be used to further improve the feature selection. CLIFF was applied by another paper [135] to select genes from the leukemia data set [41] and good classification results were obtained via three learning algorithms: a Gaussian classifier, a logistic regression classifier and a nearest neighbour classifier.

In a recent work conducted by Xu *et al* [137], gene expression data was presented as a Minimum Spanning Tree (MST), a concept from graph theory. By this presentation, each cluster of the expression data corresponded to one subtree of the MST, which rigorously converted a highly computationally intensive multi-dimensional clustering problem to a simplified tree partitioning problem. Based on the MST representation, they developed a number of efficient clustering algorithms and integrated them into a software named EXCAVATOR (EXpression data Clustering Analysis and VisualizATion Resource).

4.4 Patient Survival Analysis

Gene expression profiles with clinical outcome data enable monitoring of disease progression and prediction of patient survival at the molecular level. A few published studies have shown promising results for outcome prediction using gene expression profiles for certain diseases [102, 14, 129, 140, 88, 60].

Cox proportional hazard regression [30, 74] is a common method to study patient outcomes. It has been used by Rosenwald *et al* to analyse survival after chemotherapy for diffuse large-B-cell lymphoma (DLBCL) patients [102], and by Beer *et al* to predict patient out of lung adenocarcinoma [14]. With this method, genes most related to survival are first identified by a univariate Cox analysis, and a risk score is then defined as a linear weighted combination of the expression values of the identified genes.

Ando *et al* [9] fed gene expression profiles to a fuzzy neural network (FNN) system to predict survival of patients. Their method contained several steps. (1) Predicting the outcome of each patient using one gene at one time. (2) Ranking genes by their accuracy — the gene with the highest prediction accuracy had the highest rank. (3) Selecting partner genes for highest ranked gene. They fixed the gene with the highest rank (named as "1st gene") and used a similar prediction method to select a partner gene (named as "2nd gene") who gave the highest accuracy in combination with the "1st gene". Similarly, they fixed "1st gene" and the "2nd gene" to find a 3rd gene. This procedure stopped after six rounds or when there was no gain on accuracy. (4) Applying the procedure described in (3) to the ten highest ranked genes. (5) Using each of the ten highest ranked genes and its selected partner genes to do prediction. (6) Optimizing the resulting ten FNN models built on the combinatorial genes by the back-propagation method.

Park *et al* [88] linked gene expression data to patient survival times using the partial least squares regression technique, which is a compromise between principal component analysis and ordinary least squares regression. Shipp *et al* [114] employed the weighted voting algorithm to identify cured versus fatal for outcome of diffuse large B-cell lymphoma. The algorithm calculated the weighted combination of selected informative marker genes to make a class distinction.

In a recent publication [60], LeBlanc *et al* developed a gene index technique to identify the associations between gene expression levels and patient outcome. Genes were ordered based on linking their expression levels both to patient outcome and to a specific gene of interest. To select such a reference gene, one was recommended to consider the gene that had been identified to be most strongly related to the outcome or suggested from external data such as a protein analysis or other experimental work. The core of their proposal was to combine the correlation between genes with the correlation between genes and patient outcome as well as class membership. They demonstrated their method using the DLBCL data set collected by Rosenwald *et al* consisting of 160 patients [102].

4.5 Chapter Summary

Using gene expression data to analyse human malignancies has attracted many researchers these years. In this chapter, we did an extensive review on the technologies applied to gene expression studies, focusing on data preprocessing, gene selection and sample supervised learning. The operations of data preprocessing mainly include scale transformations, data normalization, missing value management, replicate handling, and flat pattern filtering. In the studies of gene selection, statistical methods were widely adopted while feature wrapper idea and clustering algorithms also demonstrated their efficiency. To solve the classification problem arising from gene expression data, many traditional and newly invented supervised learning approaches have been applied to distinguish tumor from non-tumor samples, one subtype from other subtypes of certain disease and so on. From the extensive literature review in this chapter, we can see that approaches to gene expression data analysis were not uniform; indeed, almost every paper presented a different method or described a novel manner/procedure to analysis.

Chapter 5

Experiments on Microarray Data — Phenotype Classification

In this chapter, our proposed gene selection process ERCOF (detailed technology description can be found in Section 3.3 of Chapter 3) and the ensemble of decision trees method CS4 (Section 2.3.4 of Chapter 2) will be applied to some bench-mark microarray gene expression and proteomic data sets to classify phenotypes. Phenotype classification is typically performed on binary type, such as tumor against non-tumor (i.e. normal). For each data set, experimental results using some other related feature filtering methods and classification algorithms will also be presented, so that reasonable comparisons can be addressed.

5.1 Experimental Design

We test our methodology on several high-dimensional data sets, which were published recently in *Science, Nature,* and other prestigious journals. All these data sets have been accumulated at `http://sdmc.i2r.a-star.edu.sg/rp/` and transformed into .data, .names format that is widely used by the software programs for data mining, machine learning and bioinformatics. See Appendix B for more detail about this data repository.

5.1.1 Classifiers and their parameter settings

In order to compare CS4 with other ensemble of decision trees methods, Bagging, AdaBoostM1 and Random forests are also run on the same data sets. Widely used state-of-the-art machine learning algorithms in gene expression analysis, support vector machines (SVM) and k nearest neighbours (k-NN are tested as well. The software implementation of these classification algorithms (except CS4) used in the experiments is *Weka* (Bagging, AdaBoostM1, SVM and k-NN in version 3.2 and Random forests in version 3.3.6), a free (under GNU) machine learning software package written in Java and developed at University of Waikato in New Zealand (`http://www.cs.waikato.ac.nz/~ml/weka/`).

For most of the algorithm parameters, we adopt the default setting of *Weka*'s implementation. Particularly, *Weka* implemented SVM using sequential minimal optimization (SMO) algorithm [93] to train the model (see section 2.3.2 for more information about SMO). Other default settings of SVM include: conducting data normalization, using polynomial kernel functions, and transforming the output into probabilities by a standard sigmoid function. Most of the time, the linear kernel function is used unless stated otherwise. As for k-NN, we also use normalized data and set the value of k to 3 (default value is 1)— i.e. 3 nearest neighbours (i.e. 3-NN) will be used in prediction.

Breiman noted in [19] that most of the improvement from bagging is evident within *ten* replications. Therefore, we set 20 (default value is 10) as the number of bagging iterations for Bagging classifier, the number of maximum boost iterations for AdaBoostM1, and the number of trees in the forest for Random forests algorithm. Below, we list the default settings in *Weka* for these three classifiers.

- Bagging. The random seed for resampling is 1.
- AdaBoostM1. Use *boosting by weighting*.
- Random forests. The number of feature candidate to consider is $int(log_2 m + 1)$, where m is the total number of features. The random seed to pick up a feature is 1.

In addition, the implementation of the base classifier C4.5 in *Weka* version 3.2 was based on its revision 8, which was the last public version before it was commercialized. We follow the default settings that tree pruning and subtree raising are conducted. The CS4 algorithm was implemented using *Weka* APIs (Version 3.2) and has been integrated into *Weka* package as one of its classifiers. By default, we also build 20 trees from each time of learning. In case the number of available features is less than 20, the number of trees will be decreased accordingly and automatically.

5.1.2 Entropy-based feature selection

In the feature selections conducted by ERCOF, we select 5% significant level (for Wilcoxon rank sum test) and 0.99 Pearson correlation coefficient threshold. For each data set, besides ERCOF, we also try the following entropy-based filtering scenarios to conduct feature selection.

- **All-entropy**: choose all the features whose value range can be partitioned into intervals by Fayyad's discretization algorithm [36] (also see Section 3.2.4 of Chapter 3), i.e. all the output features from the Phase I of ERCOF.
- **Mean-entropy**: choose features whose entropy measure is smaller than the mean entropy value of all the genes selected by above "all-entropy" strategy [64].
- **Top-number-entropy**: choose a certain number of top-ranked features according to entropy measure, such as top 20, 50, 100 and 200 genes.

In addition, performance on original intact data (i.e. whole feature space, no gene selection) are also obtained and presented under column **All** in the result table of each data set. Please note that the type of features is always numeric.

5.1.3 Performance evaluation

Since the number of samples (i.e. experiments) is small in gene expression profiles, we simply use *number of misclassified samples* in each class as the main evaluator. The format of performance presentation is $Z(X : Y)$, where X (or Y) is number of misclassified samples in the first (or second) class and $Z = X + Y$. Other evaluation measures, such as sensitivity, specificity and precision are also calculated when necessary. In most of cases, we present results obtained from a 10-fold cross validation on all samples of each data set. The samples are shuffled (with

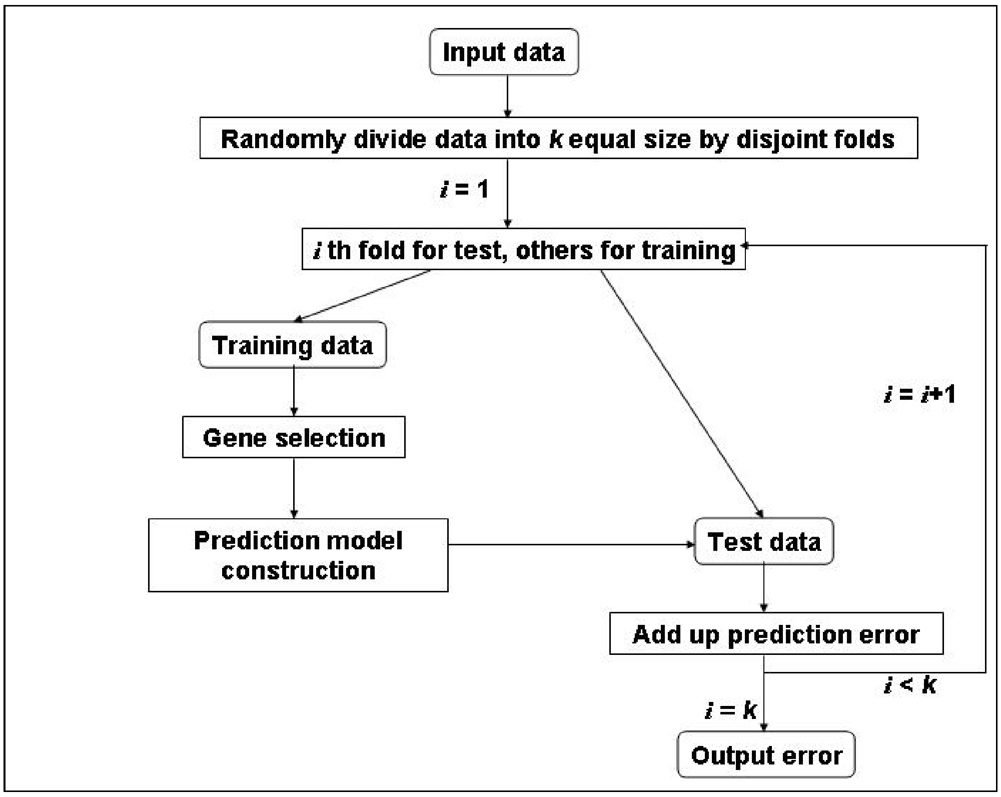

Input data
Randomly divide data into k equal size by disjoint folds
i = 1
i th fold for test, others for training
Training data
Gene selection
Prediction model construction
Test data
Add up prediction error
i = i+1
i < k
i = k
Output error

Figure 5.1: A process diagram for k-fold cross validation.

random seed 1) and stratified by *Weka* program version 3.2. In a 10-fold cross validation, since the feature selection is conducted for each fold independently, the identified genes on same data set will be vary from fold to fold. Figure 5.1 is a diagram of our process to conduct k-fold cross validation on gene expression data. Especially, k-fold cross validation becomes *leave-one-out* cross validation (LOOCV, also known as "jack-knife") when k equals the number of samples.

5.2 Experimental Results

Here, we will present our experimental results of several public gene expression profiles and one proteomic data set.

5.2.1 Colon tumor

This data set was first analysed by Alon *et al* in [7]. Its task is to distinguish cancer from normal tissue using microarray data (Affymetrix oligonucleotide array). 2000 out of around 6500 genes were selected based on the confidence in the measured expression levels. These 2000 genes have

Table 5.1: Colon tumor data set results (22 normal versus 40 tumor) on LOOCV and 10-fold cross validation. Numbers presented in bold type is the best result achieved by the corresponding classifier among 8 gene selection scenarios.

Classifier	All	All-entropy	Mean-entropy	Top-number-entropy				ERCOF
				20	50	100	200	
LOOCV								
SVM	10(5:5)	12(5:7)	9(4:5)	9(4:5)	8(4:4)	9(4:5)	13(8:5)	**7(3:4)**
3-NN	18(11:7)	**10(5:5)**	12(6:6)	**10(6:4)**	12(6:6)	**10(5:5)**	**10(5:5)**	**10(4:6)**
Bagging	10(7:3)	11(7:4)	11(7:4)	11(5:6)	11(5:6)	**10(5:5)**	**10(5:5)**	**10(5:5)**
AdaBoostM1	13(8:5)	**11(6:5)**	13(8:5)	13(8:5)	13(7:6)	13(8:5)	14(9:5)	**11(6:5)**
RandomForests	16(11:5)	15(10:5)	15(10:5)	14(8:6)	14(8:6)	15(8:7)	14(8:6)	**13(8:5)**
CS4	11(7:4)	11(7:4)	11(7:4)	12(7:5)	11(7:4)	11(7:4)	**9(6:3)**	12(4:8)
10-fold cross validation								
SVM	11(5:6)	9(5:4)	9(5:4)	**8(4:4)**	**8(4:4)**	9(5:4)	10(5:5)	**8(4:4)**
3-NN	19(12:7)	**9(5:4)**	11(5:6)	12(8:4)	10(5:5)	10(5:5)	11(6:5)	**9(5:4)**
Bagging	12(7:5)	12(7:5)	10(5:5)	11(5:6)	12(7:5)	10(5:5)	**9(4:5)**	10(5:5)
AdaBoostM1	12(8:4)	10(5:5)	12(8:4)	14(8:6)	13(7:6)	13(8:5)	14(9:5)	**9(5:4)**
RandomForests	12(5:7)	13(6:7)	13(9:4)	15(9:6)	**11(7:4)**	13(8:5)	12(6:6)	12(7:5)
CS4	14(9:5)	11(7:4)	12(7:5)	13(8:5)	12(7:5)	**9(5:4)**	13(8:5)	10(5:5)

highest minimal intensity across the 62 tissues collected from colon-cancer patients, including 40 tumor biopsies from adenocarcinoma and 22 normal biopsies from healthy parts of the colons of the same patients [7]. The raw data can be found at `http://microarray.princeton.edu/oncology/affydata/index.html`.

Table 5.1 shows the performance of different classifiers among total 8 gene selection scenarios. For this data set, since it contains a relatively smaller number of samples, we list out both LOOCV and 10-fold cross validation results.

There are 7 common genes selected by each fold ERCOF feature selection in 10-fold cross validation test. Table 5.2 lists their feature series number, GenBank accession number, sequence and name. Several of these identified features, such as features 377, 625 and 1772, were also highlighted in [68], where Bayesian algorithms incorporating feature selection were applied to the same data set. Particularly, the finding of feature 377, that corresponds to the mRNA for uroguanylin precursor, is consistent with the statement in [84] that "guanylin and uroguanylin are markedly reduced in early colon tumors with very low expression in adenocarcinoma of the colon and also in its benign precursor, the adenoma".

The best performance on LOOCV is achieved by SVM under ERCOF feature selection scenario (7 biopsies are misclassified, including 3 normal and 4 tumor samples). So far, this is also among the best prediction accuracy on this data set when compared with published results.

Table 5.2: 7 common genes selected by each fold of ERCOF in 10-fold cross validation test for colon tumor data set. UTR stands for untranslated region.

Feature number	Accession number	Sequence	Name
377	Z50753	gene	H.sapiens mRNA for GCAP-II/uroguanylin precursor
780	H40095	3' UTR	MACROPHAGE MIGRATION INHIBITORY FACTOR (HUMAN)
513	M22382	gene	MITOCHONDRIAL MATRIX PROTEIN P1 PRECURSOR (HUMAN)
625	X12671	gene	Human gene for heterogeneous nuclear ribonucleoprotein (hnRNP) core protein A1
1582	X63629	gene	H.sapiens mRNA for p cadherin
1771	J05032	gene	Human aspartyl-tRNA synthetase alpha-2 subunit mRNA, complete cds
1772	H08393	3' UTR	COLLAGEN ALPHA 2(XI) CHAIN (Homo sapiens)

Although CS4 performs worse than SVM does in terms of accuracy, it provides some learning rules. For example, Figure 5.2 gives a decision tree output by CS4 on this data set. From this tree, 5 rules can be derived directly:

(1) "If $attribute625 \leq 226.6$, then *the sample is normal*". There are 11 of normal samples (labeled as "positive") that satisfy this rule.

(2) "If $attribute625 > 226.6 \wedge attribute1772 \leq 82.0 \wedge attribute377 \leq 224.1$, then *the sample is tumor*". This rule is true for 10 of the tumor samples (labeled as "negative").

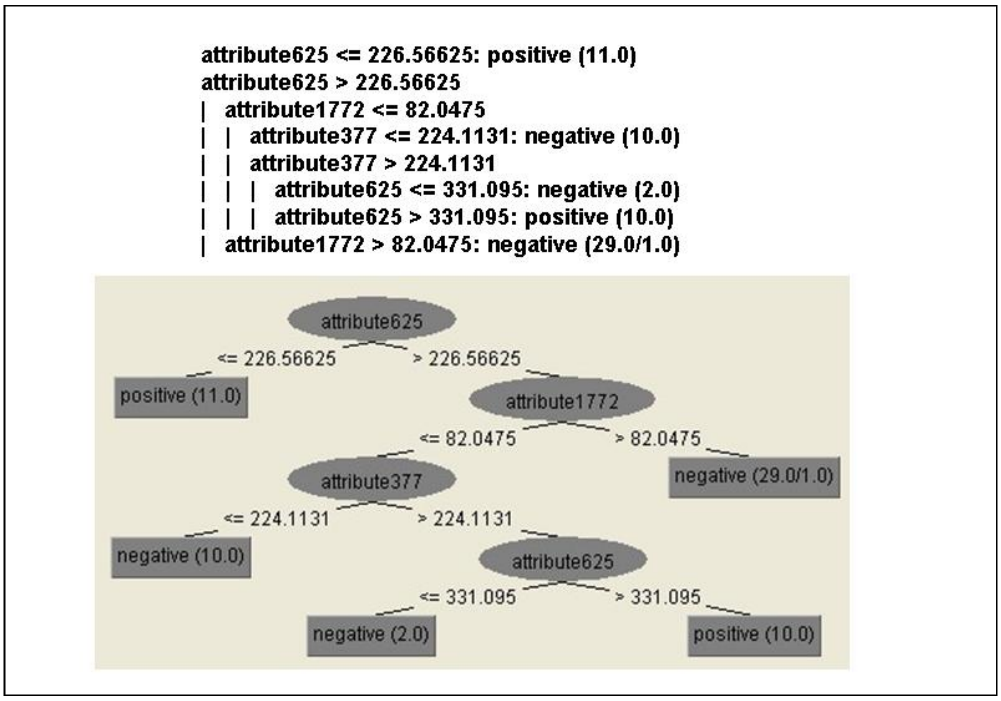
(3) "If $attribute625 > 226.6 \wedge attribute1772 \leq 82.0 \wedge attribute377 > 224.1 \wedge attribute625 \leq 331.1$, then *the sample is tumor*". This rule is true for 2 of the tumor samples.

(4) "If $attribute625 > 226.6 \wedge attribute1772 \leq 82.0 \wedge attribute377 > 224.1 \wedge attribute625 > 331.1$, then *the sample is normal*". There are 10 of the normal samples that satisfy this rule.

(5) "If $attribute625 > 226.6 \wedge attribute1772 > 82.0$, then *the sample is tumor*". This is a dominant rule for tumor samples since it is true for 28 (70%) of them; however, there is also 1 normal sample meets this rule.

5.2.2 Prostate cancer

Prostate tumors are among the most heterogeneous of cancers, both histologically and clinically [115]. Here, we will study gene expression patterns from 52 tumor and 50 normal prostate specimens. The data was obtained from oligonucleotide microarrays containing probs for approximately 12,600 genes and ESTs. According to the supplemental documents of [115], where



```
attribute625 <= 226.56625: positive (11.0)
attribute625 > 226.56625
|   attribute1772 <= 82.0475
|   |   attribute377 <= 224.1131: negative (10.0)
|   |   attribute377 > 224.1131
|   |   |   attribute625 <= 331.095: negative (2.0)
|   |   |   attribute625 > 331.095: positive (10.0)
|   attribute1772 > 82.0475: negative (29.0/1.0)
```

Figure 5.2: A decision tree output from colon tumor data set. The upper part is the tree presented in text format while the lower is the same tree in tree format. The pictures are captured from the result panel of *Weka*. In this chapter, we will mostly use the text format to illustrate a decision tree.

the data was first analysed, all expression files in a given experiment were scaled to a reference file based upon the mean average difference for all genes present on the microarray. All genes with average differences (calculated by Affymetrix GeneChip software) below the minimum threshold of 10 were set at the minimum threshold while the maximum threshold was set at 16,000. The raw data can be downloaded from `http://microarray.princeton.edu/oncology/affydata/index.html`.

Table 5.3 shows our 10-fold cross validation performance on this prostate cancer data set. SVM achieves 95% accuracy (5 errors out of total 102 samples, with 2 misclassified tumor samples and 3 misclassified normal samples) under both ERCOF and top 100 genes selected by entropy measure. CS4 also obtains good accuracy as high as 93% with 7 classification errors. In [115], greater than 90% LOOCV accuracy was claimed by using a small number of genes (from 4 to 256) selected by signal-to-noise measure and k-nearest neighbours classification algorithm. In fact, our LOOCV accuracy under ERCOF is also 95% for SVM and 93% for CS4 using average 500 genes (detailed data not shown).

There are 54 common genes selected by each fold ERCOF feature selection in 10-fold cross

Table 5.3: Prostate cancer data set results (52 tumor versus 50 normal) on 10-fold cross validation.

Classifier	All	All-entropy	Mean-entropy	Top-number-entropy				ERCOF
				20	50	100	200	
SVM	7(5:2)	8(5:3)	6(4:2)	6(4:2)	7(4:3)	**5(3:2)**	7(3:4)	**5(3:2)**
3-NN	18(8:10)	10(6:4)	8(5:3)	9(3:6)	8(4:4)	**7(4:3)**	9(5:4)	8(5:3)
Bagging	10(8:2)	9(7:2)	10(5:5)	8(4:4)	**8(4:4)**	11(5:6)	9(5:4)	9(5:4)
AdaBoostM1	14(7:7)	10(6:4)	**8(5:3)**	9(5:4)	12(6:6)	14(3:11)	10(4:6)	10(6:4)
RandomForests	21(10:11)	11(7:4)	11(6:5)	9(5:4)	10(7:3)	9(5:4)	**7(4:3)**	10(5:5)
CS4	9(7:2)	9(7:2)	8(6:2)	8(4:4)	**7(5:2)**	9(6:3)	9(6:3)	8(6:2)

Table 5.4: Classification errors on the validation set of lung cancer data, consisting of 149 samples (15 MPM versus 134 ADCA).

Classifier	All	All-entropy	Mean-entropy	Top-number-entropy				ERCOF
				20	50	100	200	
SVM	1(0:1)	1(0:1)	**0**	1(0:1)	2(1:1)	1(0:1)	**0**	**0**
3-NN	3(2:1)	1(1:0)	1(1:0)	1(1:0)	1(1:0)	**0**	1(1:0)	1(1:0)
Bagging	**4(0:4)**	5(0:5)	5(0:5)	20(3:17)	12(2:10)	8(1:7)	6(0:6)	6(0:6)
AdaBoostM1	27(4:23)	27(4:23)	27(4:23)	27(4:23)	27(4:23)	27(4:23)	27(4:23)	27(4:23)
RandomForests	5(0:5)	7(0:7)	3(2:1)	8(1:7)	3(1:2)	3(1:2)	**2(0:2)**	**2(0:2)**
CS4	3(1:2)	3(1:2)	3(1:2)	3(1:2)	3(1:2)	3(1:2)	3(1:2)	3(1:2)

validation test. Table A.1 in the Appendix lists their probe number, GenBank accession number, and name. Some of them were also announced by [115] as significant genes to distinguish tumor from normal prostate samples. For examples, AF037643, M17885, AL031228, and X07732 and so on.

5.2.3 Lung cancer

This data set is about the distinction between malignant pleural mesothelioma (MPM) and adenocarcinoma (ADCA) of the lung by using the gene expression profiles on 181 tissue samples (31 MPM and 150 ADCA) obtained from oligonucleotide chips. Each sample is described by 12,533 genes. In [42], where this data was first studied, samples was divided into a training set consisting 16 MPM and 16 ADCA, and a validation set containing the rest 149 samples. The raw data can be found from `http://www.chestsurg.org/microarray.htm.` Table 5.4 shows the errors on test set using our proposed scenarios.

This data set has several features: (1) The size of training set is small, but the number of samples in each class is balanced. Test set contains more than three times samples than those in the training set, and the number of MPM samples is only one ninth of that of ADCA samples. (2)

There are as many as 16 genes having zero entropy value on training samples. This means that using any one of them can separate MPM and ADCA completely. Thus, in the construction of a C4.5 decision tree, a tree will contain only one feature and two rules, one rule for MPM samples and another one for ADCA samples. In this case, since the base classifier C4.5 has no training error, the algorithm of AdaBoostM1 will not proceed to generate new trees and therefore, it is equivalent to C4.5. Unfortunately, none of these genes can 100% classify the samples in the validation set alone — the best one misclassifies 4 samples. Table 5.5 gives the cut point for each of these 16 genes that can separate MPM and ADCA samples in the training set completely, as well as the testing error of C4.5 decision tree built only on that gene. The cut point is the middle point of the gene's boundary expression value in each class. For example: if the maximum expression value of a gene having zero entropy in MPM class samples is 100 while the minimum expression value of the same gene in ADCA samples is 500, then the cut point value of this gene will be 300 and we say the gene has lower expression level in MPM samples and higher level in ADCA samples. (3) Although there is no single gene that can give 100% correct prediction on the testing samples, the combination of all of them will lead to a near perfect accuracy — 99.3% prediction accuracy with only one MPM sample misclassified by SVM and 3-NN. (4) Furthermore, when more genes are considered, 100% accuracy on testing is achieved by SVM using mean-entropy, top 200 entropy or ERCOF selected features, or by 3-NN using top 100 entropy measure genes.

In the study on the data set in [42], marker genes with a highly significant difference ($p < 2 \times 10^{-6}; \geq 8-$fold) in average expression levels between 16 MPM and 16 ADCA training samples were explored. From them, 8 genes with the most statistically significant differences and a mean expression level $>$ 600 in at least one of the two training sample sets were chosen to form 15 expression ratio patterns. The best test accuracy reported was also 99.3% (with 1 error). Among the 8 significant genes, we find 3 of them with zero entropy. They are highlighted with bold font in Table 5.6 where the probe name, GenBank accession number and gene name of those 16 zero entropy genes are listed. The remaining 5 genes also have relatively smaller entropy values, they are X56667 (GenBank accession number), entropy rank 31; X16662, rank 32; AJ011497, rank 33; AB023194, rank 37; and U43203, rank 56.

By the way, we also obtain the 10-fold cross validation results on this data set and list them

Table 5.5: 16 genes with zero entropy measure in the training set of lung cancer data. Cut point is the expression value of the gene that can be used to separate MPM and ADCA samples completely. The column “lower” (or “higher”) indicates the class that all of its samples have their expression values of this gene *not greater than* (or *greater than*) the cut point.

Probe Name	Cut Point	Lower	Higher	Test Error
2047_s_at	571.1	MPM	ADCA	27(4:23)
266_s_at	76.95	MPM	ADCA	20(2:18)
32046_at	103.2	MPM	ADCA	16(3:13)
32551_at	73.45	MPM	ADCA	15(1:14)
33245_at	48.3	MPM	ADCA	12(1:11)
33833_at	453.7	ADCA	MPM	10(2:8)
35330_at	25.3	ADCA	MPM	31(1:30)
36533_at	193.25	ADCA	MPM	8(2:6)
37205_at	78.8	ADCA	MPM	14(3:11)
37716_at	197.75	ADCA	MPM	**4(4:0)**
39795_at	1167	ADCA	MPM	14(1:13)
40936_at	430.6	ADCA	MPM	9(3:6)
41286_at	41.5	MPM	ADCA	28(2:26)
41402_at	54.6	MPM	ADCA	26(2:24)
575_s_at	149.75	MPM	ADCA	8(1:7)
988_at	31	MPM	ADCA	17(2:15)

in Table 5.7. Many scenarios have less than 4 misclassified samples, achieving overall accuracy above 98%. Remarkably, random forests makes no error using mean-entropy selected genes.

5.2.4 Ovarian cancer

Different from other data sets studied in this chapter, this disease analysis is about using *proteomic* spectra generated from mass spectrometer for ovarian cancer detection. The initial publication [92] on this new diagnostic approach was in *Lancet* in February 2002, in which analysis of serum from 50 unaffected women and 50 patients with ovarian cancer were conducted and a proteomic pattern that completely discriminated cancer from non-cancer was identified. As described in [29], when we use proteomic patterns to diagnose disease, the sample drawn from the patient is first applied to a protein chip which is made up of a specific chromatographic surface, and then analysed via mass spectrometry. The result is simply a mass spectrum of the species that bound to and subsequently desorbed from the array surface. The pattern of peaks within the spectrum is studied to diagnose the source of the biological sample. A process diagram of how to diagnose disease using proteomic patterns is captured from [29] and given in Figure 5.3. One obvious advantage of this process is that raw biofluids, such as urine, serum and plasma, can be directly applied to the array surface. On the other hand, as pointed out in [29], there are criticisms

Table 5.6: GenBank accession number and name of 16 genes with zero entropy measure in the training set of lung cancer data. Three genes in bold font were also selected by [42].

Probe name	Accession number	Gene name
2047_s_at	M23410	Human plakoglobin (PLAK) mRNA, complete cds
266_s_at	L33930	Homo sapiens CD24 signal transducer mRNA, complete cds and 3 region
32046_at	D10495	Homo sapiens mRNA for protein kinase C delta-type, complete cds
32551_at	U03877	Human extracellular protein (S1-5) mRNA, complete cds
33245_at	AF004709	Homo sapiens stress-activated protein kinase 4 mRNA, complete cds
33833_at	J05243	Human nonerythroid alpha-spectrin (SPTAN1) mRNA, complete cds
35330_at	AJ012737	Homo sapiens mRNA for filamin, muscle isoform
36533_at	D83402	Homo sapiens gene for prostacyclin synthase
37205_at	AB020647	Homo sapiens mRNA for KIAA0840 protein, partial cds
37716_at	X05323	Human MRC OX-2 gene signal sequence
39795_at	D63475	Human mRNA for KIAA0109 gene, complete cds
40936_at	AI651806	Homo sapiens cDNA, 3'end
41286_at	X77753	H.sapiens TROP-2 gene
41402_at	AL080121	Homo sapiens mRNA; cDNA DKFZp564O0823 (from clone DKFZp564O0823)
575_s_at	M93036	Human (clone 21726) carcinoma-associated antigen GA733-2 (GA733-2) mRNA, exon 9 and complete cds
988_at	X16354	Human mRNA for transmembrane carcinoembryonic antigen BGPa (formerly TM1-CEA)

Table 5.7: 10-fold cross validation results on whole lung cancer data set, consisting of 31 MPM and 150 ADCA samples.

Classifier	All	All-entropy	Mean-entropy	Top-number-entropy				ERCOF
				20	50	100	200	
SVM	**1(1:0)**	**1(1:0)**	**1(1:0)**	**1(1:0)**	2(2:0)	2(2:0)	**1(1:0)**	**1(1:0)**
3-NN	11(11:0)	3(3:0)	2(2:0)	**1(1:0)**	2(2:0)	2(2:0)	2(2:0)	**1(1:0)**
Bagging	6(5:1)	6(5:1)	6(5:1)	7(5:2)	**5(4:1)**	**5(4:1)**	6(5:1)	6(5:1)
AdaBoostM1	6(3:3)	7(3:4)	5(2:3)	3(2:1)	**2(1:1)**	**2(1:1)**	3(2:1)	6(3:3)
RandomForests	2(2:0)	2(2:0)	**0**	1(1:0)	2(2:0)	1(1:0)	1(1:0)	1(1:0)
CS4	2(2:0)	2(2:0)	**1(1:0)**	3(3:0)	**1(1:0)**	2(2:0)	2(2:0)	**1(1:0)**

of using proteomic patterns for diagnostic purpose — mainly because the identity of the proteins or peptides giving rise to the key m/z features is not known. However, this debate is beyond the scope of this thesis.

After the first publication about using proteomic spectra to detect cancer, a series of new data and discussions on proteomic patterns were put on the FDA-NCI Clinical Proteomics Program Databank web site at `http://clinicalproteomics.steem.com/`. Recently (updated in August 2003), an important development about using a higher resolution mass spectrometer to generate proteomic patterns was announced publicly. Compared with the configuration of the old Ciphergen instrument (about 100 to 200 spots), there is a tremendous increase in resolution of the new Q-Star instrument (>9000 at m/z 1500). Besides, mass accuracy is also improved — Q-Star 10ppm versus Ciphergen 1000ppm.

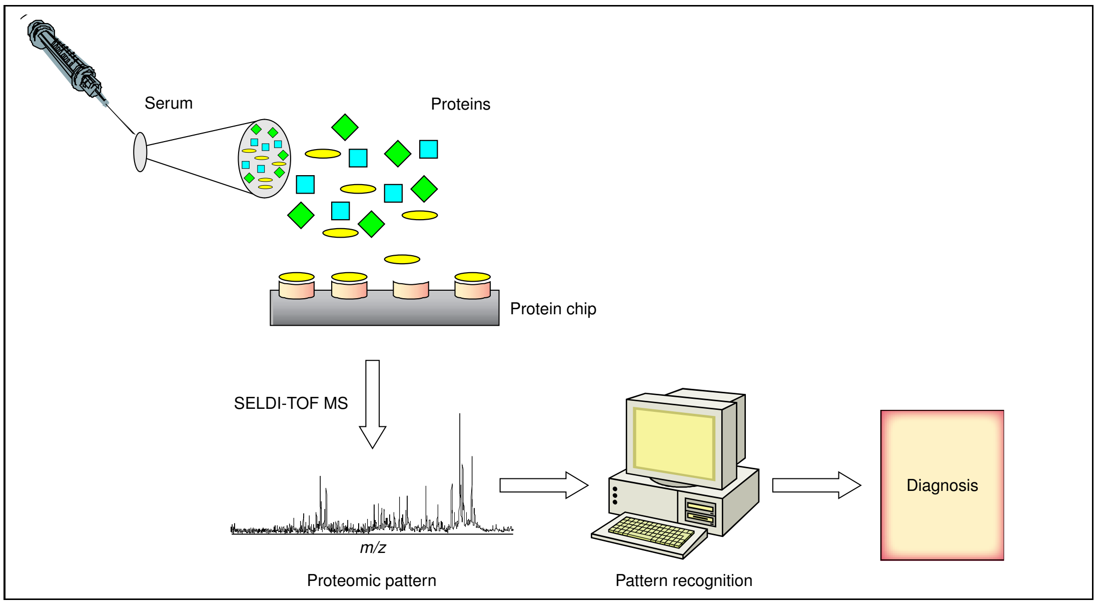


Figure 5.3: Disease diagnostics using proteomic patterns. Picture is from [29]. m/z stands for mass to charge ratio and SELDI-TOF MS for surface-enhanced laser desorption/ionization time-of-flight mass spectrometry.

Here, we apply our proposed feature selection and machine learning approach to an ovarian proteomic data set named “6-19-02”. This sample set included 91 controls and 162 ovarian cancers. The raw SELDI (surface-enhanced laser desorption/ionization) data constructed using the Ciphergen WCX2 ProteinChip had 15154 molecular m/z (mass to charge ratio) identities ranging from 0.0000786 to 19995.513. The relative amplitude of the intensity at each m/z identity was normalized against the most intense and the least intense values in the data stream according to the formula

$$NV = \frac{(V - V_{min})}{(V_{max} - V_{min})} \tag{5.1}$$

where NV is the normalized value, V the raw value, V_{min} the minimum and V_{max} the maximum raw data of the identity across all the samples, respectively. After this linear normalization, all the m/z intensities fell within the range [0,1]. Table 5.8 lists the 10-fold cross validation results on 253 samples with normalized intensities using our proposed scenarios. Notably, both SVM and CS4 achieve 100% accuracy under certain feature selection methods. This may indicate that machine learning technologies can also be used to find proteomic patterns.

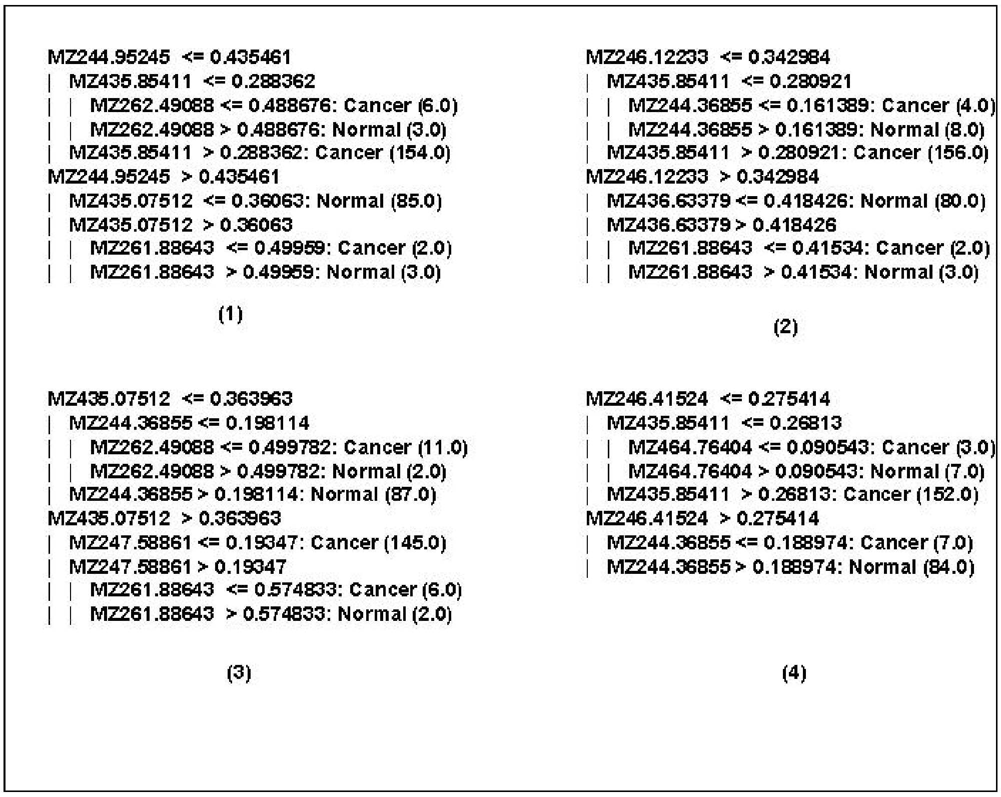
In the above mentioned web site, associated with this “6-19-02” ovarian cancer data, there was also a list of seven key m/z values which was announced as an example of the best models

Table 5.8: 10-fold cross validation results on “6-19-02” ovarian proteomic data set, consisting of 162 ovarian cancer versus 91 control samples.

Classifier	All	All-entropy	Mean-entropy	Top-number-entropy				ERCOF
				20	50	100	200	
SVM	**0**	**0**	**0**	4(1:3)	**0**	**0**	**0**	**0**
3-NN	15(6:9)	11(3:8)	10(3:7)	4(1:3)	**2(0:2)**	3(0:3)	4(0:4)	3(1:2)
Bagging	7(3:4)	6(3:3)	**5(3:2)**	7(4:3)	**5(3:2)**	6(3:3)	**5(3:2)**	6(3:3)
AdaBoostM1	10(4:6)	9(4:5)	8(4:4)	6(4:2)	**4(4:0)**	5(4:1)	6(4:2)	5(4:1)
RandomForests	19(6:13)	8(1:7)	5(0:5)	7(3:4)	**3(0:3)**	4(0:4)	6(1:5)	5(1:4)
CS4	**0**	**0**	1(0:1)	5(2:3)	1(0:1)	**0**	**0**	**0**

found to 100% correctly separate ovarian cancer and non-cancer samples. These m/z identities are: MZ2760.6685, MZ19643.409, MZ465.56916, MZ6631.7043, MZ14051.976, MZ435.4652 and MZ3497.5508. However, among these seven M/Z values, we find 3 of them will be filtered out by the Phase I of ERCOF, i.e. the entropy algorithm can not find cut point for their value ranges. They are: MZ2760.6685, MZ19643.409 and MZ6631.7043. With the remaining 4 identities, SVM can still achieve 100% accuracy on 10-fold cross validation and some simple rules are found to separate cancer and non-cancer samples completely by decision tree method. For example, the simple rule, “if *MZ435.46452>0.335733* ∧ *MZ465.56916<0.666745*, then *the sample is ovarian cancer*”, is true for 148 of 162 cancer samples.

A recent paper presented the work on this data set is [118], which used non-parametric Wilcoxon rank sum test statistics and stepwise discriminant analysis to develop patterns and rules from proteomic profiling. Using Wilcoxon test, the paper reported that 685 out of total 15154 m/z values differing between the cancer and non-cancer populations with a p-value of less than 10^{-6}. On the other hand, refer to our 10-fold cross validation results in Table 5.8, the top 50 entropy measure selected features can lead to a 100% accuracy and we further find there are as many as 39 common m/z values among each time feature selection for 10 folds. These 39 m/z identities are all in the ERCOF selected common features for 10-fold cross validation. In the Appendix, we list in Table A.3 these m/z values, their corresponding Wilcoxon test p-values and entropy measure on the entire data set. The p-values are derived from the supplementary figures of paper [118]. Notably, their Wilcoxon p-values are all very small ($< 10^{-21}$). With these 39 m/z identities, CS4 outputs several decision trees, and each of them can separate cancer from non-cancer completely. Figure 5.4 shows only four of them.



```
MZ244.95245  <= 0.435461
|  MZ435.85411  <= 0.288362
|  |  MZ262.49088 <= 0.488676: Cancer (6.0)
|  |  MZ262.49088 > 0.488676: Normal (3.0)
|  MZ435.85411  > 0.288362: Cancer (154.0)
MZ244.95245  > 0.435461
|  MZ435.07512  <= 0.36063: Normal (85.0)
|  MZ435.07512  > 0.36063
|  |  MZ261.88643  <= 0.49959: Cancer (2.0)
|  |  MZ261.88643  > 0.49959: Normal (3.0)

                    (1)
```

```
MZ246.12233  <= 0.342984
|  MZ435.85411  <= 0.280921
|  |  MZ244.36855 <= 0.161389: Cancer (4.0)
|  |  MZ244.36855 > 0.161389: Normal (8.0)
|  MZ435.85411  > 0.280921: Cancer (156.0)
MZ246.12233  > 0.342984
|  MZ436.63379 <= 0.418426: Normal (80.0)
|  MZ436.63379 > 0.418426
|  |  MZ261.88643  <= 0.41534: Cancer (2.0)
|  |  MZ261.88643  > 0.41534: Normal (3.0)

                    (2)
```

```
MZ435.07512  <= 0.363963
|  MZ244.36855 <= 0.198114
|  |  MZ262.49088 <= 0.499782: Cancer (11.0)
|  |  MZ262.49088 > 0.499782: Normal (2.0)
|  MZ244.36855 > 0.198114: Normal (87.0)
MZ435.07512  > 0.363963
|  MZ247.58861 <= 0.19347: Cancer (145.0)
|  MZ247.58861 > 0.19347
|  |  MZ261.88643  <= 0.574833: Cancer (6.0)
|  |  MZ261.88643  > 0.574833: Normal (2.0)

                    (3)
```

```
MZ246.41524  <= 0.275414
|  MZ435.85411  <= 0.26813
|  |  MZ464.76404 <= 0.090543: Cancer (3.0)
|  |  MZ464.76404 > 0.090543: Normal (7.0)
|  MZ435.85411  > 0.26813: Cancer (152.0)
MZ246.41524  > 0.275414
|  MZ244.36855 <= 0.188974: Cancer (7.0)
|  MZ244.36855 > 0.188974: Normal (84.0)

                    (4)
```

Figure 5.4: Four decision trees output by CS4 using 39 common features selected by top 50 entropy measure on 10-fold cross validation on ovarian cancer proteomic profiling. All these trees are constructed on the entire 253 samples and can separate cancer and non-cancer completely.

5.2.5 Diffuse large B-cell lymphoma

Diffuse large B-cell lymphoma (DLBCL) is the most common subtype of non-Hodgkin's lymphoma. Although around 40% of DLBCL patients are cured with current therapy and have prolonged survival, the remainder succumb to the disease [6]. Recently, DLBCL was widely studied at molecular level using gene expression profilings [6, 102, 114]. Alizadeh *et al* [6] identified two distinct forms of DLBCL which had gene expression patterns indicative of different stages of B-cell differentiation. *Germinal center B-like* DLBCL expresses genes normally seen in germinal center B cells, while *activated B-like* DLBCL expresses genes that are induced during *in vitro* activation of peripheral blood B cells. They showed that patients with germinal center B-like DLBCL had a significantly better overall survival than those with activated B-like DLBCL. Thus, accurately classifying germinal center B-like DLBCL and activated B-like DLBCL will help with survival prediction.

The DLBCL gene expression data studied in [6] contains 4026 genes across 47 samples,

Table 5.9: 10-fold cross validation results on DLBCL data set, consisting of 24 germinal center B-like DLBCL versus 23 activated B-like DLBCL.

Classifier	All	All-entropy	Mean-entropy	Top-number-entropy				ERCOF
				20	50	100	200	
SVM	6(3:3)	3(1:2)	**2(1:1)**	6(4:2)	3(1:2)	4(2:2)	3(1:2)	**2(1:1)**
3-NN	13(1:12)	5(2:3)	5(2:3)	5(3:2)	5(3:2)	**3(1:2)**	5(2:3)	4(2:2)
Bagging	**6(3:3)**	**6(3:3)**	7(3:4)	8(3:5)	8(3:5)	8(3:5)	**6(3:3)**	8(3:5)
AdaBoostM1	11(4:7)	11(5:6)	10(4:6)	**8(4:4)**	9(4:5)	11(5:6)	10(4:6)	10(5:5)
RandomForests	5(4:1)	**1(0:1)**	4(3:1)	3(2:1)	4(3:1)	6(2:4)	3(2:1)	3(2:1)
CS4	5(2:3)	5(2:3)	5(2:3)	6(2:4)	**4(2:2)**	5(2:3)	5(2:3)	5(2:3)

including 24 germinal center B-like DLBCL and 23 activated B-like DLBCL. The data and associated information can be found at `http://llmpp.nih.gov/lymphoma/`. The raw data were originally filtered by several criteria and log-transformed (base 2). For details of data preprocessing, please refer to [6]. Table 5.9 shows the 10-fold cross validation results on this DLBCL data set under our proposed scenarios. The results demonstrate that, overall speaking, germinal center B-like DLBCL and activated B-like DLBCL can be classified. Random forests achieves best cross validation results — having only one sample misclassified using all entropy measure selected genes. SVM still performs well — giving only two misclassified samples in two cases. In addition, using ERCOF as feature selection method, the number of misclassified samples in LOOCV test for SVM, CS4 and random forests are 2(1:1), 4(2:2) and 3(2:1), respectively.

Table 5.10 lists the 9 common genes selected by each fold ERCOF feature selection in the 10-fold cross validation test. All of them are in the "list of best class-predicting genes supporting the GC-B Like v.s. Activated B-Like class distinction" of paper [6] (see supplemental Figure 3 on the data web site given above). Besides, our identified genes are also highly consistent with those reported in [126], where t-test, Wilcoxon rank sum test and a heuristic method were applied to select genes on the same data set. Notably, we find that the GENE1207X (or FLIP), whose products inhibit programmed cell death, highly expressed in activated B-like DLBCL. According to [6], "FLIP is a dominant-negative mimic of caspase 8 (FLICE) which can block apoptosis mediated by Fas and other death receptors.....FLIP is highly expressed in many tumor types and its constitutive expression in activated B-like DLBCLs could inhibit apoptosis of tumor cells induced by host T cells expressing Fas ligand". On the other hand, simply using these 9 genes, 3-NN and Random forests can separate 24 germinal center B-like DLBCL from 23 activated B-like

Table 5.10: 9 common genes selected by each fold of ERCOF in 10-fold cross validation test on DLBCL data set. The third column indicates the DLBCL sub-class that the gene was relatively highly expressed.

GID	Gene name	HighlyExpressed in
GENE3328X	Unknown UG Hs.136345 ESTs; Clone=746300	GC-B Like
GENE3314X	*Unknown; Clone=1353041	GC-B Like
GENE1252X	*Cyclin D2/KIAK0002=3' end of KIAK0002 cDNA; Clone=1357360	activated B-like
GENE3325X	Unknown UG Hs.120245 Homo sapiens mRNA for KIAA1039 protein, partial cds; Clone=1268870	GC-B Like
GENE3946X	*PTP-1B=phosphotyrosyl-protein phosphatase; Clone=472182	activated B-like
GENE2106X	Similar to intersectin=adaptor protein with two EH and five SH3 domains; Clone=1339781	GC-B Like
GENE2291X	Unknown; Clone=1340742	activated B-like
GENE3258X	*JAW1=lymphoid-restricted membrane protein; Clone=815539	GC-B Like
GENE1207X	*FLICE-like inhibitory protein long form=I-FLICE=FLAME-1 =Casper=MRIT=CASH=cFLIP=CLARP; Clone=711633	activated B-like

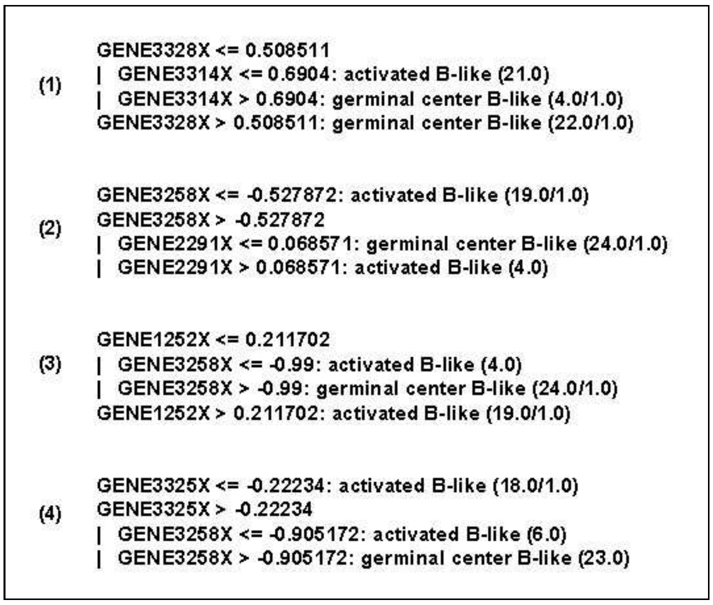
DLBCL completely while both SVM and CS4 only misclassify one activated B-like DLBCL. Figure 5.5 displays some decision trees output from running CS4 on these 9 genes.

5.2.6 ALL-AML leukemia

This leukemia data first reported by Golub *et al* [41] is among the most extensively analysed gene expression profilings. Many researchers have tested their clustering, gene selection and/or classification algorithms on this bench mark data set [39, 136, 87, 123, 86, 85, 31, 68, 82, 51]. The original training data consists of 38 bone marrow samples with 27 ALL (acute lymphoblastic leukemia) and 11 AML (acute myeloid leukemia) from adult patients. The test data set consisted of 24 bone marrow samples and 10 peripheral blood specimens from adults and children, including 20 ALL and 14 AML. The gene expression profile were obtained from Affymetrix high-density oligonucleotide microarrays containing 7129 probes for 6817 human genes. The raw data can be downloaded from `http://www.broad.mit.edu/cgi-bin/cancer/datasets.cgi`.

In Table 5.11, we list results on 34 test samples as well as 10-fold cross validation and LOOCV on entire 72 samples using our proposed gene selection and classification scenarios. Our best result of both testing and cross validation is to misclassify only one sample. In fact, this misclassified AML sample was reported by most of other investigators.

ERCOF selects 280 genes from training set samples. Table A.4 in the Appendix lists the probe and name of these genes. In [41], 50 genes found by signal-to-noise measurement



```
      GENE3328X <= 0.508511
(1)   |  GENE3314X <= 0.6904: activated B-like (21.0)
      |  GENE3314X > 0.6904: germinal center B-like (4.0/1.0)
      GENE3328X > 0.508511: germinal center B-like (22.0/1.0)

      GENE3258X <= -0.527872: activated B-like (19.0/1.0)
(2)   GENE3258X > -0.527872
      |  GENE2291X <= 0.068571: germinal center B-like (24.0/1.0)
      |  GENE2291X > 0.068571: activated B-like (4.0)

      GENE1252X <= 0.211702
(3)   |  GENE3258X <= -0.99: activated B-like (4.0)
      |  GENE3258X > -0.99: germinal center B-like (24.0/1.0)
      GENE1252X > 0.211702: activated B-like (19.0/1.0)

      GENE3325X <= -0.22234: activated B-like (18.0/1.0)
(4)   GENE3325X > -0.22234
      |  GENE3258X <= -0.905172: activated B-like (6.0)
      |  GENE3258X > -0.905172: germinal center B-like (23.0)
```

Figure 5.5: Four decision trees output by CS4 using 9 common features selected by ERCOF on 10-fold cross validation on DLBCL data. All these trees are constructed on the entire 47 samples, including 24 germinal center B-like DLBCL and 23 activated B-like DLBCL.

that most highly correlated with ALL and AML distinction from the training samples were reported. Remarkably, 49 of them are also in our 280 genes list and indicated with bold font in Table A.4. In addition, Olshen and Jain [85] reported 40 significant genes identified by t-test with a permutation-based adjustment. These genes are all included in our list, but some of them (13 out of 40) are not in Golub's 50-gene list. On the other hand, there are 80 common genes selected by ERCOF in each fold of 10-fold cross validation on the entire 72 samples. Fifty of them are in the list of Table A.4 in the Appendix. Based on training set samples, there is one gene (Zyxin) with zero entropy (1017.58 is the cut point and it highly expressed in AML samples). However, with only this one gene, classification algorithms can not achieve good testing results on validation set. At this point, Golub *et al* commented "in any case, we recommend using at least 10 genes ...... Class predictors using a small number of genes may depend too heavily on any one gene and can produce spuriously high prediction strengths".

Using SAM described in Section 4.2.1, a statistical software designed for identifying significant genes in a set of microarray experiments, total of 2857 genes are output with the threshold δ at 0.4789. Table 5.12 lists the classification results on 34 testing samples using different top genes ranked by SAM score. We can see that SVM, k-NN and random forests can not achieve good testing results using SAM selected genes on this data set, but AdaBoostM1 achieves bet-

Table 5.11: ALL-AML leukemia data set results (ALL versus AML) on testing samples, as well as 10-fold cross validation and LOOCV on the entire set.

Classifier	All	All-entropy	Mean-entropy	Top-number-entropy				ERCOF
				20	50	100	200	
				Test				
SVM	5(0:5)	**1(0:1)**	**1(0:1)**	4(0:4)	5(1:4)	**1(0:1)**	**1(0:1)**	**1(0:1)**
3-NN	10(1:9)	6(0:6)	2(0:2)	3(0:3)	4(1:3)	2(0:2)	**1(0:1)**	**1(0:1)**
Bagging	3(0:3)	4(0:4)	4(0:4)	**2(1:1)**	4(0:4)	4(0:4)	4(0:4)	4(0:4)
AdaBoostM1	3(2:1)	3(2:1)	3(2:1)	3(2:1)	3(2:1)	3(2:1)	3(2:1)	3(2:1)
RandomForests	9(0:9)	4(0:4)	6(0:6)	4(1:3)	5(0:5)	**2(0:2)**	**2(0:2)**	3(0:3)
CS4	4(0:4)	4(0:4)	3(0:3)	**2(1:1)**	4(0:4)	3(0:3)	3(0:3)	3(0:3)
				10-fold cross validation				
SVM	**1(0:1)**	2(1:1)	2(1:1)	2(1:1)	5(2:3)	3(2:1)	2(1:1)	2(1:1)
3-NN	10(1:9)	2(0:2)	**1(0:1)**	4(3:1)	4(2:2)	4(2:2)	2(1:1)	2(0:2)
Bagging	5(0:5)	6(0:6)	5(0:5)	**4(0:4)**	6(1:5)	6(1:5)	6(1:5)	6(2:4)
AdaBoostM1	13(6:7)	11(5:6)	12(5:7)	**6(3:3)**	7(4:3)	10(6:4)	10(5:5)	9(4:5)
RandomForests	6(0:6)	5(0:5)	4(1:3)	4(0:4)	4(1:3)	**3(0:3)**	5(0:5)	5(2:3)
CS4	1(0:1)	2(0:2)	2(0:2)	3(1:2)	2(1:1)	**1(0:1)**	2(0:2)	2(1:1)
				LOOCV				
SVM	**1(0:1)**	**1(0:1)**	2(1:1)	4(2:2)	5(2:3)	4(2:2)	2(1:1)	**1(0:1)**
3-NN	10(1:9)	**1(0:1)**	**1(0:1)**	4(3:1)	5(3:2)	2(1:1)	3(2:1)	**1(0:1)**
Bagging	7(3:4)	6(1:5)	**5(0:5)**	**5(0:5)**	**5(0:5)**	6(1:5)	6(1:5)	**5(1:4)**
AdaBoostM1	11(6:5)	10(5:5)	11(5:6)	**6(3:3)**	**6(3:3)**	7(4:3)	10(5:5)	7(4:3)
RandomForests	8(0:8)	6(2:4)	**4(1:3)**	**4(0:4)**	5(2:3)	5(2:3)	6(3:3)	**4(1:3)**
CS4	2(1:1)	2(1:1)	2(1:1)	**1(0:1)**	**1(0:1)**	2(1:1)	2(1:1)	**1(0:1)**

ter results (with top 350, 280 or 200 genes) than using our proposed gene selection schemes. Remarkably, bagging makes no testing error on top 350 SAM selected genes. As for CS4, the performance is relatively stable by using 100 to 350 SAM selected genes. When we compare the genes identified by SAM with those 280 selected by ERCOF, we only find 125 and 17 common genes from all 2857 and top 280 SAM selected genes, respectively.

Using PAM described in Section 4.2.3, a class prediction software for genomic expression data mining based on nearest shrunken centroid method, Tibshirani *et al* reported 2 misclassified

Table 5.12: ALL-AML leukemia data set results (ALL versus AML) on testing samples by using top genes ranked by SAM score. *: the number is approximate to the number of all-entropy selected genes; **: the number is approximate to the number of mean-entropy selected genes; ***: the number is approximate to the number of ERCOF selected genes.

Classifier	2857	800^{*}	350^{**}	280^{***}	200	100	50	20
SVM	**3(0:3)**	4(0:4)	4(1:3)	5(1:4)	6(2:4)	10(4:6)	11(3:8)	11(0:11)
3-NN	11(1:10)	11(1:10)	**10(0:10)**	**10(0:10)**	**10(0:10)**	11(0:11)	13(1:12)	12(0:12)
Bagging	2(0:2)	2(0:2)	**0**	2(1:1)	2(1:1)	2(1:1)	9(0:9)	8(1:7)
AdaBoostM1	3(0:3)	3(1:2)	**1(0:1)**	**1(0:1)**	**1(0:1)**	2(0:2)	12(4:8)	8(1:7)
RandomForests	13(0:13)	9(0:9)	8(0:8)	**6(0:6)**	**6(0:6)**	**6(0:6)**	11(0:11)	11(0:11)
CS4	6(1:5)	4(1:3)	**2(0:2)**	3(0:3)	4(0:4)	**2(1:1)**	8(0:8)	9(2:7)

test samples by using only 21 genes with an amount of shrinkage δ at 4.06 [123]. This δ value is not the optimal one where the minimum cross-validation errors occurs since there will be more than 1000 genes associated with the optimal δ value. Anyway, our classification results are very competitive on this data set — misclassifying only 1 test sample.

5.2.7 Subtypes of pediatric acute lymphoblastic leukemia

Pediatric acute lymphoblastic leukemia (ALL) is the most common form of childhood cancer. However, with modern cancer therapy, its overall long-term event-free survival rates is as high as 80% [140]. Treatment of pediatric ALL is based on the concept of tailoring the intensity of therapy to a patient's risk of relapse. Thus, it becomes very important to accurately assign individual patients into specific risk groups; otherwise, it would cause under-treatment (which causes relapse and eventual death) or over-treatment (which causes severe long-term side-effects). Although current risk assignment is mainly dependent on a variety of clinical and laboratory parameters requiring an extensive range of procedures including morphology, immunophenotyping, cytogenetics, and molecular diagnostics, it has been noticed that the genetic alterations that underlie the pathogenesis of individual leukemia subtypes are also playing important roles [95, 140]. Though it looks identical under the microscope, pediatric ALL is a highly heterogeneous disease, with as many as 6 different subtypes that have widely differing treatment outcome. The purpose of the analysis on this data set is to accurately classify subtypes of pediatric ALL using gene expression profiling so that the correct intensity of therapy can be delivered to ensure that the child would have the highest chance for cure.

The data is a collection of 327 gene expression profiles of pediatric ALL diagnostic bone marrows with Affymetrix oligonucleotide microarrays containing 12,600 probe sets [140]. The raw data can be found from `http://www.stjuderesearch.org/data/ALL1/`. These samples contain all known biologic ALL subtypes, including T lineage leukemias (*T-ALL*), B lineage leukemias that contain t(9;22) (*BCR-ABL*), t(1;19) (*E2A-PBX1*), t(12;21) (*TEL-AML1*), rearrangement in the MLL gene on chromosome 11, band q23 (*MLL*), and a hyperdiploid karyotype (i.e. > 50 chromosomes) (*Hyperdip>50*) [140]. In [140], where the data was first analysed, 327 samples were divided into two groups — a training group consisting of 215 samples and a testing group consisting of 112 samples. Table 5.13 lists the number of samples of each subtype

Table 5.13: Number of samples in each of subtypes in pediatric acute lymphoblastic leukemia data set.

Subtype	Number of training samples	Number of testing samples	total
T-ALL	28	15	43
E2A-PBX1	18	9	27
TEL-AML1	52	27	79
BCR-ABL	9	6	15
MLL	14	6	20
Hyperdip>50	42	22	64
Rest	52	27	79
Total	215	112	327

in training and testing groups, and the diagnostic samples that did not fit into any one of the above subtypes are put under "Rest".

In [140], classification was designed following a decision tree format, in which the first decision was T-ALL (T lineage) versus non-T-ALL (B lineage) and then within the B lineage subset. If a case is decided to be a non-T-ALL, it will be sequentially classified into the known risk groups characterized by the presence of E2A-PBX1, TEL-AML1, BCR-ABL, MLL, and lastly hyperdip>50. A very high prediction accuracy on the blinded test samples was achieved for each ALL subtypes using SVM and genes selected by t-statistic, $\mathcal{X}^2$-statistic or other metrics: 100% on T-ALL, E2A-PBX1 and MLL samples, 99% on TEL-AML1 samples, 97% on BCR-ABL samples, and 96% on Hyperdip>50 samples. However, in this thesis, we will not follow this tree structure to sequentially classify samples; instead, we will treat all subtypes equally and distinguish one subtype samples from all the other samples. Therefore, for each of the 6 classification problems, number of training and testing samples are always 215 and 112, respectively. For example, for subtype BCR-ABL, the 215 training samples consist of 9 BCR-ABL cases versus 206 "OTHERS" while 112 testing samples consist of 6 BCR-ABL cases versus 106 "OTHERS". The samples labeled as "OTHERS" here include all the cases other than BCR-ABL. Next, we will report classification results on the validation samples and 10-fold cross validation on the entire data set under our proposed gene selection and classification scenarios.

T-ALL versus OTHERS

The training set contains 28 *T-ALL* and 187 *OTHERS* samples while the test set contains 15 *T-ALL* and 97 *OTHERS*. Table 5.14 shows the results of this test. Under most of our scenarios, the T-

Table 5.14: Pediatric ALL data set results (T-ALL versus OTHERS) on 112 testing samples, as well as 10-fold cross validation on the entire 327 cases.

Classifier	All	All-entropy	Mean-entropy	Top-number-entropy				ERCOF
				20	50	100	200	
Test								
SVM	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
3-NN	3(3:0)	**0**	**0**	**0**	**0**	**0**	**0**	**0**
Bagging	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
AdaBoostM1	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
RandomForests	4(4:0)	**0**	**0**	1(1:0)	**0**	**0**	1(1:0)	**0**
CS4	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
10-fold cross validation								
SVM	1(1:0)	**0**	**0**	1(1:0)	**0**	**0**	**0**	**0**
3-NN	8(8:0)	3(3:0)	**0**	1(1:0)	1(1:0)	1(1:0)	**0**	1(1:0)
Bagging	**1(1:0)**	**1(1:0)**	**1(1:0)**	**1(1:0)**	**1(1:0)**	**1(1:0)**	**1(1:0)**	**1(1:0)**
AdaBoostM1	**1(1:0)**	**1(1:0)**	**1(1:0)**	**1(1:0)**	**1(1:0)**	**1(1:0)**	**1(1:0)**	**1(1:0)**
RandomForests	11(11:0)	**0**	**0**	**0**	**0**	**0**	1(1:0)	**0**
CS4	1(0:1)	1(1:0)	1(1:0)	**0**	2(1:1)	3(2:1)	2(2:0)	**0**

ALL samples can be distinguished completely from non-T-ALL cases. Remarkably, we find one gene, AA919102 (GenBank accession number), has zero entropy value from training samples with cut point 20062.86 (highly expressed in T-ALL cases) and this gene can also completely separates T-ALL from all other ALL cases in the testing set. This gene was also reported in [140] where other feature selection metrics were used. Besides, the genes selected by ERCOF in each fold testing of 10-fold cross validation are highly consistent, having as many as 253 common genes. However, it seems that using small amount of good features identified by entropy measure is enough to separate T-ALL cases in this application, we list out in Table 5.15 the top 20 genes found from training samples.

E2A-PBX1 versus OTHERS

The training set contains 18 *E2A-PBX1* and 197 *OTHERS* samples while the test set contains 9 *E2A-PBX1* and 103 *OTHERS*. Table 5.16 shows the results of this test. With feature selection, the testing E2A-PBX1 samples can be distinguished completely from other subtypes of ALL cases. Similarly, in 10-fold cross validation test, there are quite a few scenarios that achieve 100% accuracy. There are 5 genes whose entropy value is zero in training samples. With these genes, all the classification algorithms can achieve 100% prediction accuracy on testing samples. In table 5.17, we list all of them. In addition, all these 5 genes are in the “good genes list” reported in [140] to distinguish E2A-PBX1 cases. In the supplemental documents of [140],

Table 5.15: Top 20 genes selected by entropy measure from the training data set of T-ALL versus OTHERS in subtypes of pediatric ALL study. The last column indicates the sample class in which the gene is relatively highly expressed.

Probe	Accession No.	Description	HighlyExp in
38319_at	AA919102	Homo sapiens cDNA, 3' end	T-ALL
1096_g_at	M28170	Human cell surface protein CD19 (CD19) gene	OTHERS
38242_at	AF068180	Homo sapiens B cell linker protein BLNK mRNA, alternatively spliced	OTHERS
41723_s_at	M32578	Human MHC class II HLA-DR beta-1 mRNA (DR2.3), 5' end	OTHERS
32794_g_at	X00437	Human mRNA for T-cell specific protein	T-ALL
37988_at	M89957	Human immunoglobulin superfamily member B cell receptor complex cell surface glycoprotein (IGB) mRNA	OTHERS
37344_at	X62744	Human RING6 mRNA for HLA class II alpha chain-like product	OTHERS
38095_i_at	M83664	Human MHC class II lymphocyte antigen (HLA-DP) beta chain mRNA	OTHERS
38017_at	U05259	Human MB-1 gene	OTHERS
35016_at	M13560	Human Ia-associated invariant gamma-chain gene	OTHERS
36277_at	M23323	Human membrane protein (CD3-epsilon) gene	T-ALL
39318_at	X82240	H.sapiens mRNA for Tcell leukemia/lymphoma 1	OTHERS
38147_at	AL023657	Homo sapiens SH2D1A cDNA, formerly known as DSHP	T-ALL
32649_at	X59871	Human TCF-1 mRNA for T cell factor 1 (splice form C)	T-ALL
38833_at	X00457	Human mRNA for SB classII histocompatibility antigen alpha-chain	OTHERS
33238_at	U23852	Human T-lymphocyte specific protein tyrosine kinase p56lck (lck) abberant mRNA	T-ALL
37039_at	J00194	human hla-dr antigen alpha-chain mrna & ivs fragments	OTHERS
38051_at	X76220	H.sapiens MAL gene exon 1 (and joined CDS)	T-ALL
38096_f_at	M83664	Human MHC class II lymphocyte antigen (HLA-DP) beta chain mRNA	OTHERS
2059_s_at	M36881	Human lymphocyte-specific protein tyrosine kinase (lck) mRNA	T-ALL

good genes identified by the self-organizing map (SOM) and discriminant analysis with variance (DAV) programs to separate each of the six known subtypes ALL were reported.

TEL-AML1 versus OTHERS

The training set contains 52 *TEL-AML1* and 163 *OTHERS* samples while the test set contains 27 *TEL-AML1* and 85 *OTHERS*. Table 5.18 shows the results of this test. Although the validation result on classification of TEL-AML is not as good as that of subtype T-ALL or E2A-PBX1, there are still some proposed scenarios can accurately distinguish TEL-AML and non-TEL-AML cases. Notably, using ERCOF selected features, SVM, 3-NN, Random forests and CS4 achieve 100% prediction accuracy on the testing samples. The number of features selected by ERCOF from training cases is around 400 and they include 37 of 46 genes that reported in [140] to separate TEL-AML1 from other subtypes of ALL cases under their proposed tree structure of

Table 5.16: Pediatric ALL data set results (E2A-PBX1 versus OTHERS) on 112 testing samples, as well as 10-fold cross validation on the entire 327 cases.

Classifier	All	All-entropy	Mean-entropy	Top-number-entropy				ERCOF
				20	50	100	200	
				Test				
SVM	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
3-NN	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
Bagging	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
AdaBoostM1	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
RandomForests	3(0:3)	**0**	**0**	**0**	**0**	**0**	**0**	**0**
CS4	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
				10-fold cross validation				
SVM	1(1:0)	1(1:0)	1(1:0)	**0**	**0**	**0**	**0**	**0**
3-NN	1(1:0)	1(1:0)	1(1:0)	**0**	**0**	1(1:0)	1(1:0)	**0**
Bagging	**1(1:0)**	**1(1:0)**	**1(1:0)**	**1(1:0)**	**1(1:0)**	**1(1:0)**	**1(1:0)**	**1(1:0)**
AdaBoostM1	**1(1:0)**	**1(1:0)**	**1(1:0)**	**1(1:0)**	**1(1:0)**	**1(1:0)**	**1(1:0)**	**1(1:0)**
RandomForests	16(16:0)	3(3:0)	1(1:0)	**0**	**0**	**0**	**0**	**0**
CS4	1(1:0)	1(1:0)	1(1:0)	1(1:0)	1(1:0)	1(1:0)	1(1:0)	**0**

Table 5.17: Five genes with zero entropy measure on the training data set of E2A-PBX1 versus OTHERS in subtypes of pediatric ALL study. The last column indicates the sample class in which the gene is highly expressed (above the mean value across all the samples).

Probe	Accession No.	Description	HighlyExpressed in
32063_at	M86546	H.sapiens PBX1a and PBX1b mRNA	E2A-PBX1
41146_at	J03473	Human poly(ADP-ribose) synthetase mRNA	E2A-PBX1
430_at	X00737	Human mRNA for purine nucleotide phosphorylase (PNP; EC 2.4.2.1)	E2A-PBX1
1287_at	J03473	Human poly(ADP-ribose) synthetase mRNA	E2A-PBX1
33355_at	AL049381	Homo sapiens mRNA; cDNA DKFZp586J2118 (from clone DKFZp586J2118)	E2A-PBX1

classification. In Table A.10 of the Appendix, we list these 37 highlighted genes. In Figure 5.6, we present some decision trees output by CS4 using ERCOF selected features. It can be seen that CS4 makes use of different features as root node and combines them to achieve a perfect prediction accuracy on the testing samples.

BCR-ABL versus OTHERS

The training set contains 9 *BCR-ABL* and 206 *OTHERS* samples while the test set contains 6 *BCR-ABL* and 106 *OTHERS*. Table 5.19 shows the results of this test. Since the number of available BCR-ABL cases is very small, most error predications made are on BCR-ABL samples in almost all the scenarios. This leads to a very low sensitivity, especially in 10-fold cross validation test. However, under ERCOF and some other gene selection methods, SVM and CS4 still can

Table 5.18: Pediatric ALL data set results (TEL-AML1 versus OTHERS) on 112 testing samples, as well as 10-fold cross validation on the entire 327 cases.

Classifier	All	All-entropy	Mean-entropy	Top-number-entropy				ERCOF
				20	50	100	200	
Test								
SVM	10(0:10)	**0**	**0**	2(1:1)	1(0:1)	1(0:1)	1(0:1)	**0**
3-NN	5(4:1)	**0**	**0**	1(1:0)	1(1:0)	1(1:0)	1(1:0)	**0**
Bagging	**1(1:0)**	2(2:0)	2(2:0)	**1(1:0)**	**1(1:0)**	**1(1:0)**	**1(1:0)**	2(2:0)
AdaBoostM1	4(2:2)	4(3:1)	3(2:1)	**2(2:0)**	4(2:2)	4(3:1)	5(2:3)	4(3:1)
RandomForests	11(11:0)	**0**	1(1:0)	1(1:0)	2(2:0)	**0**	1(1:0)	**0**
CS4	2(1:1)	2(1:1)	2(1:1)	3(3:0)	1(1:0)	1(1:0)	1(1:0)	**0**
10-fold cross validation								
SVM	4(1:3)	3(1:2)	4(1:3)	7(2:5)	8(2:6)	5(2:3)	5(2:3)	**2(0:2)**
3-NN	14(5:9)	4(0:4)	4(0:4)	8(3:5)	6(2:4)	7(3:4)	4(1:3)	**3(0:3)**
Bagging	12(5:7)	11(5:6)	**10(4:6)**	11(5:6)	**10(4:6)**	11(5:6)	11(5:6)	**10(4:6)**
AdaBoostM1	9(4:5)	13(7:6)	14(9:5)	**8(3:5)**	**8(5:3)**	13(10:3)	13(8:5)	10(4:6)
RandomForests	20(17:3)	7(3:4)	7(3:4)	5(0:5)	5(1:4)	**4(0:4)**	6(2:4)	**4(1:3)**
CS4	6(2:4)	6(2:4)	6(2:4)	10(5:5)	**5(1:4)**	6(2:4)	6(2:4)	**5(1:4)**

correctly predict most of the testing samples with only one BCR-ABL case misclassified. This misclassified BCR-ABL sample was also reported by [140]. The number of features selected by ERCOF from training cases is around 70 and they include 11 of 21 genes that reported in [140] to separate BCR-ABL from other subtypes of ALL cases under their proposed tree structure of classification. In Table 5.20, we list these 11 highlighted genes.

MLL versus OTHERS

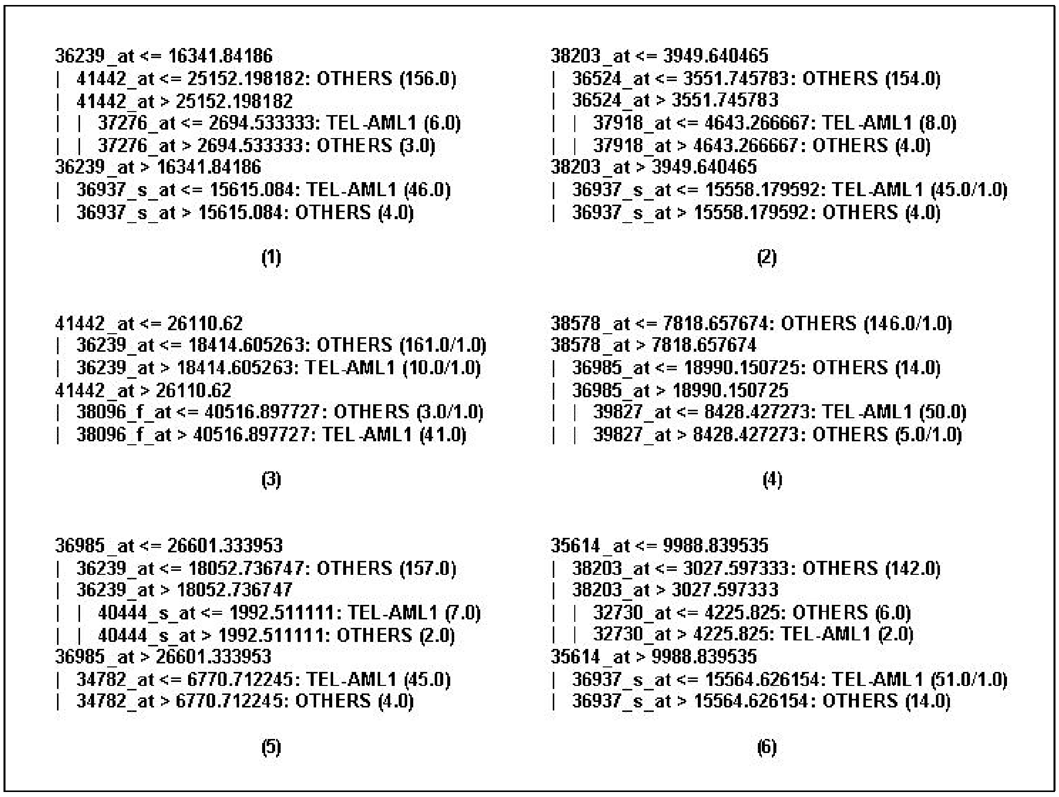
The training set contains 14 *MLL* and 201 *OTHERS* samples while the test set contains 6 *MLL* and 106 *OTHERS*. Table 5.21 shows the results of this test. Most of our scenarios achieve 100% accuracy on testing samples to separate MLL from other subtypes of ALL cases. Using only 20 genes selected by entropy measure, SVM, 3-NN, Bagging and CS4 can make perfect prediction. These genes can be found in Table A.11 of the Appendix. When we apply Pearson correlation coefficient to the 20 genes (all of them can pass Wilcoxon rank sum test) , we filter out only one gene . With the remaining 19 genes, 100% prediction can also be achieved. On the other hand, there are 34 genes reported in [140] to be significant to separate MLL from other ALL subtypes under their proposed tree structure classification. Among them, 24 genes are also selected by our ERCOF and we list them in Table A.12 of the Appendix, where the genes with bold font are also appear in Table A.11.

Table 5.19: Pediatric ALL data set results (BCR-ABL versus OTHERS) on 112 testing samples, as well as 10-fold cross validation on the entire 327 cases.

Classifier	All	All-entropy	Mean-entropy	Top-number-entropy				ERCOF
				20	50	100	200	
Test								
SVM	4(4:0)	**1(1:0)**	2(1:1)	2(1:1)	2(1:1)	**1(1:0)**	**1(1:0)**	**1(1:0)**
3-NN	6(6:0)	3(3:0)	2(2:0)	**1(1:0)**	4(4:0)	4(4:0)	4(4:0)	2(2:0)
Bagging	5(5:0)	3(3:0)	2(2:0)	**1(1:0)**	4(4:0)	3(3:0)	3(3:0)	3(3:0)
AdaBoostM1	8(4:4)	**5(1:4)**	**5(1:4)**	**5(1:4)**	**5(1:4)**	**5(1:4)**	**5(1:4)**	**5(1:4)**
RandomForests	6(6:0)	6(6:0)	2(2:0)	**1(1:0)**	2(2:0)	6(6:0)	4(4:0)	2(2:0)
CS4	6(6:0)	**1(1:0)**	**1(1:0)**	**1(1:0)**	**1(1:0)**	**1(1:0)**	**1(1:0)**	**1(1:0)**
10-fold cross validation								
SVM	12(12:0)	8(6:2)	8(7:1)	6(5:1)	9(7:2)	**4(4:0)**	8(6:2)	6(5:1)
3-NN	15(14:1)	9(9:0)	10(9:1)	9(7:2)	10(9:1)	8(8:0)	10(10:0)	**7(7:0)**
Bagging	13(13:0)	12(11:1)	12(11:1)	**10(10:0)**	12(11:1)	11(11:0)	12(11:1)	**10(10:0)**
AdaBoostM1	22(13:9)	18(11:7)	15(10:5)	**8(7:1)**	15(10:5)	16(9:7)	16(10:6)	16(12:4)
RandomForests	15(15:0)	10(10:0)	7(7:0)	**6(6:0)**	7(7:0)	11(11:0)	12(12:0)	9(9:0)
CS4	8(8:0)	7(7:0)	6(6:0)	8(7:1)	**5(5:0)**	6(6:0)	7(7:0)	7(6:1)

Table 5.20: Eleven genes selected by ERCOF on training samples and reported in [140] to separate BCR-ABL from other subtypes of ALL cases in pediatric ALL study. All these genes are relatively highly expressed (above the mean value across all the samples) in BCR-ABL samples.

Probe	Accession No.	Description
37600_at	U68186	Human extracellular matrix protein 1 mRNA
38312_at	AL050002	Homo sapiens mRNA; cDNA DKFZp564O222
39730_at	X16416	Human c-abl mRNA encoding p150 protein
40051_at	D31762	Human mRNA for KIAA0057 gene
40504_at	AF001601	Homo sapiens paraoxonase (PON2) mRNA
34362_at	M55531	Human glucose transport-like 5 (GLUT5) mRNA
36591_at	X06956	Human HALPHA44 gene for alpha-tubulin, exons 1-3
40196_at	D88153	Homo sapiens mRNA for HYA22
1635_at	U07563	Human proto-oncogene tyrosine-protein kinase (ABL) gene, exon 1a and exons 2-10
1636_g_at	U07563	Human proto-oncogene tyrosine-protein kinase (ABL) gene, exon 1a and exons 2-10
330_s_at	HG2259-HT2348	Tubulin, Alpha 1, Isoform 44



```
36239_at <= 16341.84186
|   41442_at <= 25152.198182: OTHERS (156.0)
|   41442_at > 25152.198182
|   |   37276_at <= 2694.533333: TEL-AML1 (6.0)
|   |   37276_at > 2694.533333: OTHERS (3.0)
36239_at > 16341.84186
|   36937_s_at <= 15615.084: TEL-AML1 (46.0)
|   36937_s_at > 15615.084: OTHERS (4.0)
```

(1)

```
38203_at <= 3949.640465
|   36524_at <= 3551.745783: OTHERS (154.0)
|   36524_at > 3551.745783
|   |   37918_at <= 4643.266667: TEL-AML1 (8.0)
|   |   37918_at > 4643.266667: OTHERS (4.0)
38203_at > 3949.640465
|   36937_s_at <= 15558.179592: TEL-AML1 (45.0/1.0)
|   36937_s_at > 15558.179592: OTHERS (4.0)
```

(2)

```
41442_at <= 26110.62
|   36239_at <= 18414.605263: OTHERS (161.0/1.0)
|   36239_at > 18414.605263: TEL-AML1 (10.0/1.0)
41442_at > 26110.62
|   38096_f_at <= 40516.897727: OTHERS (3.0/1.0)
|   38096_f_at > 40516.897727: TEL-AML1 (41.0)
```

(3)

```
38578_at <= 7818.657674: OTHERS (146.0/1.0)
38578_at > 7818.657674
|   36985_at <= 18990.150725: OTHERS (14.0)
|   36985_at > 18990.150725
|   |   39827_at <= 8428.427273: TEL-AML1 (50.0)
|   |   39827_at > 8428.427273: OTHERS (5.0/1.0)
```

(4)

```
36985_at <= 26601.333953
|   36239_at <= 18052.736747: OTHERS (157.0)
|   36239_at > 18052.736747
|   |   40444_s_at <= 1992.511111: TEL-AML1 (7.0)
|   |   40444_s_at > 1992.511111: OTHERS (2.0)
36985_at > 26601.333953
|   34782_at <= 6770.712245: TEL-AML1 (45.0)
|   34782_at > 6770.712245: OTHERS (4.0)
```

(5)

```
35614_at <= 9988.839535
|   38203_at <= 3027.597333: OTHERS (142.0)
|   38203_at > 3027.597333
|   |   32730_at <= 4225.825: OTHERS (6.0)
|   |   32730_at > 4225.825: TEL-AML1 (2.0)
35614_at > 9988.839535
|   36937_s_at <= 15564.626154: TEL-AML1 (51.0/1.0)
|   36937_s_at > 15564.626154: OTHERS (14.0)
```

(6)

Figure 5.6: Six decision trees output by CS4 using ERCOF selected features on TEL-AML subtype classification of pediatric ALL data.

Hyperdip$>$50 versus OTHERS

The training set contains 42 *Hyperdip*$>$ 50 and 173 *OTHERS* samples while the test set contains 22 *Hyperdip*$>$ 50 and 90 *OTHERS*. Table 5.22 shows the results of this test. Although the cross validation results is not very encouraging, some of our scenarios still achieve 100% prediction accuracy on the testing samples, such as SVM using all-entropy, mean-entropy, top 200 entropy and ERCOF selected features. Based on training cases, ERCOF selects around 300 genes, which include 19 of the 26 genes that reported in [140] to separate Hyperdip$>$50 from other subtype ALL cases. These 19 highlighted genes are listed in Table A.13 of the Appendix.

A brief summary

As mentioned, different from [140] where the pediatric ALL data set was first analysed and the classification was based on a given tree structure to sequentially classify a new case into a subtype of ALL, our study focused on distinguishing a subtype of samples from all other cases. Therefore, the number of samples in the different classes are more unbalanced in both training

Table 5.21: Pediatric ALL data set results (MLL versus OTHERS) on 112 testing samples, as well as 10-fold cross validation on the entire 327 cases.

Classifier	All	All-entropy	Mean-entropy	Top-number-entropy				ERCOF
				20	50	100	200	
Test								
SVM	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
3-NN	2(2:0)	**0**	**0**	**0**	1(1:0)	**0**	**0**	**0**
Bagging	2(2:0)	1(1:0)	**0**	**0**	**0**	**0**	**0**	**0**
AdaBoostM1	4(2:2)	**1(0:1)**	**1(0:1)**	2(1:1)	2(1:1)	**1(0:1)**	**1(0:1)**	**1(0:1)**
RandomForests	5(5:0)	2(2:0)	1(1:0)	**0**	1(1:0)	**0**	1(1:0)	1(1:0)
CS4	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
10-fold cross validation								
SVM	7(7:0)	2(2:0)	2(1:1)	7(6:1)	2(1:1)	**0**	2(1:1)	2(2:0)
3-NN	9(9:0)	5(5:0)	**4(3:1)**	8(7:1)	7(6:1)	8(8:0)	5(4:1)	**4(2:2)**
Bagging	10(9:1)	9(8:1)	8(7:1)	8(7:1)	9(8:1)	9(8:1)	8(7:1)	**5(5:0)**
AdaBoostM1	13(7:6)	14(9:5)	18(13:5)	14(12:2)	**12(10:2)**	14(12:2)	18(13:5)	13(6:7)
RandomForests	18(18:0)	10(10:0)	**7(7:0)**	9(8:1)	**7(6:1)**	8(7:1)	9(9:0)	9(9:0)
CS4	7(6:1)	5(4:1)	6(5:1)	7(6:1)	10(7:3)	5(4:1)	5(4:2)	**4(4:0)**

Table 5.22: Pediatric ALL data set results (Hyperdip>50 versus OTHERS) on 112 testing samples, as well as 10-fold cross validation on the entire 327 cases.

Classifier	All	All-entropy	Mean-entropy	Top-number-entropy				ERCOF
				20	50	100	200	
Test								
SVM	18(18:0)	**0**	**0**	4(1:3)	4(1:3)	1(0:1)	**0**	**0**
3-NN	4(3:1)	2(0:2)	**0**	5(1:4)	1(1:0)	2(2:0)	2(1:1)	3(1:2)
Bagging	6(4:2)	6(4:2)	**5(4:1)**	6(4:2)	7(4:3)	9(4:5)	8(4:4)	6(3:3)
AdaBoostM1	10(4:6)	12(3:9)	10(4:6)	5(3:2)	**2(1:1)**	3(1:2)	**2(2:0)**	10(4:6)
RandomForests	9(8:1)	3(2:1)	3(2:1)	5(2:3)	3(2:1)	3(2:1)	2(1:1)	**1(1:0)**
CS4	4(3:1)	4(3:1)	3(2:1)	8(3:5)	5(1:4)	**2(1:1)**	**2(1:1)**	3(2:1)
10-fold cross validation								
SVM	11(8:3)	9(6:3)	11(9:2)	15(8:7)	15(10:5)	15(10:5)	18(11:7)	**8(6:2)**
3-NN	21(16:5)	13(9:4)	16(13:3)	15(9:6)	14(9:5)	17(11:6)	**12(9:3)**	**12(8:4)**
Bagging	19(16:3)	19(15:4)	20(16:4)	**17(11:6)**	**17(12:5)**	18(12:6)	18(14:4)	19(15:4)
AdaBoostM1	23(14:9)	24(16:8)	**14(10:4)**	17(9:8)	17(10:7)	**14(9:5)**	17(11:6))	**14(9:5)**
RandomForests	31(28:3)	19(15:4)	15(12:3)	17(11:6)	14(10:4)	18(13:5)	15(10:5)	**13(9:4)**
CS4	14(10:4)	14(10:4)	15(11:4)	20(10:10)	17(9:8)	17(10:7)	**12(9:3)**	14(10:4)

Table 5.23: Total number of misclassified testing samples over six subtypes of pediatric ALL study. Number with bold font in each row indicates the best result achieved by the corresponding classifier.

Classifier	All	All-entropy	Mean-entropy	Top-number-entropy				ERCOF
				20	50	100	200	
SVM	32	**1**	2	8	8	3	2	**1**
3-NN	20	5	**2**	8	7	7	6	5
Bagging	14	12	9	**8**	12	13	12	11
AdaBoostM1	26	22	19	**13**	**13**	**13**	**13**	20
RandomForests	38	11	7	8	8	9	9	**4**
CS4	12	7	6	12	7	**4**	**4**	**4**

and testing sets, which is easier to cause bias in prediction. However, some of our proposed classification algorithms and feature selection methods still achieved excellent testing results on all the six known subtypes classification. In addition, for 10-fold cross validation on the entire data set we also obtained very good results on classification of subtypes T-ALL, E2A-PBX1 and MLL.

In Table 5.23, for each of our scenarios, we add up the number of misclassified testing samples over all six known subtypes. Remarkably, SVM, Random forests and CS4 achieved their best prediction accuracy under ERCOF — misclassified 1, 4 and 4 testing samples, respectively. In addition, we also demonstrated the advantage of CS4 by presenting some of the decision trees output by the algorithm.

5.3 Comparisons and Discussions

We have conducted more than one thousand experiments on six gene expression profiles and one proteomic data set using proposed feature selection and classification methods. From the large amount of results presented above, we can address various comparisons and discussions. In the following comparisons, we will use the results of these 20 tests: (1) 10-fold cross validation on colon tumor, prostate cancer, lung cancer, ovarian disease, DLBCL, ALL-AML leukemia and six subtypes of pediatric ALL (total 12 tests); (2) validation on the testing samples of lung cancer, ALL-AML leukemia and six subtypes of pediatric ALL (total 8 tests).

5.3.1 Classification algorithms

We employed six classification algorithms in the experiments, four ensemble of decision trees methods, SVM and k-NN.

Comparison among ensemble of decision trees methods

First, let's do a comparison among the four ensemble of decision trees classification methods — Bagging, AdaBoostM1, Random forests and CS4. Table 5.24 shows the best classifier(s) (of these four methods) for each experiment under different feature selection scenarios. From the summary in the last row of the table, We can see that under every proposed feature selection scenario, the performance of CS4 was much superior than that of Bagging and AdaBoostM1. Besides, CS4 performed much better than Random forests did under four feature selection scenarios and did equally good under the other cases. On the other hand, because CS4 only makes use of unchanged original training samples (in contrast to bootstrapped data), the decision trees/rules output are more reliable. This concern is crucial in bio-medical applications, such as understanding and diagnosis of a disease.

Note that, AdaBoostM1 performed poorly in these experiments. The main reason is that, when its base classifier C4.5 makes no training error, AdaBoostM1 only constructs a single tree and thus loses the power of combining different decision trees. A typical example can be found in the prediction on lung cancer validation samples where AdaBoostM1 made 27 misclassified predictions under every feature selection scenario, which in fact, is the same as C4.5. Recall that, with the training samples of this data set, there are 16 genes having zero entropy value. This leads to a very simple decision tree consisting only one feature and having 100% training accuracy. Unfortunately, this feature is not good enough to give good prediction on testing samples. By the way, let's explore more on the prediction power of combining decision trees in CS4.

Power of combining decision trees in CS4

In each experiment, CS4 built 20 decision trees using different features as the root node. First, let's look at the prediction power of each single decision tree. Table 5.25 illustrates the number of misclassified training and testing samples of each single tree in the experiments on the pediatric ALL data to classify TEL-AML1 and Hyperdip$>$50 subtypes using ERCOF selected genes.

Table 5.24: Comparison among four ensemble of decision trees methods under different feature selection scenarios using the results of the 640 (=4x8x20) experiments on the six gene expression profiles and one proteomic data set. Symbol "C" stands for Bagging classifier, "D" for AdaBoostM1, "E" for Random forests, and "F" for CS4. Each cell indicates the symbol(s) of the classifier(s) that achieved best performance in the relevant experiment under the corresponding feature selection scenario. For each feature selection scenario, the last row indicates the total number of experiments that individual decision tree based classifier achieved best prediction accuracy (including tie cases). If we add up the numbers across eight feature selection scenarios, the final result is — Bagging 42, AdaBoostM1 36, random forests 72 and CS4 108.

Experiment	All	All-entropy	Mean-entropy	Top-number-entropy				ERCOF
				20	50	100	200	
ColonTumor	C,D,E	D	C	C	E	F	C	D
Prostate	F	C,F	D,F	C,F	F	E,F	E	F
Lung test	F	F	E,F	F	E,F	E,F	E	E
Lung	E,F	E,F	E	E	E	E	E	E,F
Ovarian	F	F	F	F	F	F	F	F
DLBCL	E,F	E	E	E	E,F	F	E	E
ALLAML test	C,D	D	D,F	C,F	D	E	E	D,E,F
ALLAML	F	F	F	F	F	F	F	F
Pediatric ALL data — test								
T-ALL	C,D,F	C,D,E,F	C,D,E,F	C,D,F	C,D,E,F	C,D,E,F	C,D,F	C,D,E,F
E2A-PBX1	C,D,F	C,D,E,F	C,D,E,F	C,D,E,F	C,D,E,F	C,D,E,F	C,D,E,F	C,D,E,F
TEL-AML1	C	E	E	C,E	C,F	E	C,E,F	E,F
BCR-ABL	C	F	F	C,E,F	F	F	F	F
MLL	F	F	C,F	C,E,F	C,F	C,E,F	C,F	C,F
Hyperdip>50	F	F	E,F	D,F	D	F	D,E,F	E
Pediatric ALL data — 10-fold cross validation								
T-ALL	C,D,F	E	E	E,F	E	E	E	E
E2A-PBX1	C,D,F	C,D,F	C,D,E,F	E	E	E	E	E,F
TEL-AML1	F	F	F	E	E,F	F	E	E
BCR-ABL	F	F	F	E	F	F	F	F
MLL	F	F	F	F	E	F	F	F
Hyperdip>50	F	F	D	C,D,E	C,D,F	D	F	E
Sum	C: 8	C: 4	C: 5	C: 9	C: 5	C: 3	C: 5	C: 3
	D: 6	D: 5	D: 6	D: 4	D: 5	D: 3	D: 3	D: 4
	E: 3	E: 6	E: 9	E: 11	E: 10	E: 10	E: 11	E: 12
	F: 16	F: 15	F: 14	F: 12	F: 13	F: 14	F: 11	F: 13

From the figures displayed in the table, we can observe that: (1) these single decision trees constructed using different good features as root node possess similar merit with little difference. Although some single trees gave much better prediction (e.g. tree no. 5) while other trees did very bad (e.g. tree no. 2) on the same test (TEL-AML1 versus OTHERS), there are no rules that can be drawn across experiments. Especially, the first tree is not always the best one. (2) A single tree can achieve good training accuracy, but poor testing results, such as tree no. 1 and tree no. 2 of the test on Hyperdip>50. (3) A single tree contains fewer number of features so that it is easy to be understood and interpreted. However, as shown in Figure 5.4, 5.5 and 5.6, different trees produce different rules so that the *single coverage constraint* problem in a single tree can

Table 5.25: The training and testing errors of 20 single decision trees generated by CS4 using ERCOF selected features on testings of TEL-AML1 versus OTHERS and Hyperdip>50 versus OTHERS in pediatric ALL data set. The row "No. features" gives the number of features used in the tree.

Tree No.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
TEL-AML1 versus OTHERS — testing																				
Train	0	3	0	1	1	2	2	3	0	3	0	1	1	0	1	1	3	0	2	2
Test	8	12	3	5	1	5	8	8	4	6	5	11	8	4	9	4	11	2	7	7
No. features	4	3	4	4	4	3	4	4	5	3	5	5	5	6	5	5	3	6	4	4
Hyperdip>50 versus OTHERS — testing																				
Train	0	0	2	1	2	1	3	3	3	1	1	1	0	0	1	2	3	1	0	1
Test	10	10	11	11	7	15	13	11	6	10	8	15	7	13	7	7	12	9	9	8
No. features	7	7	8	8	8	8	6	6	5	6	8	8	8	8	8	8	7	7	8	8

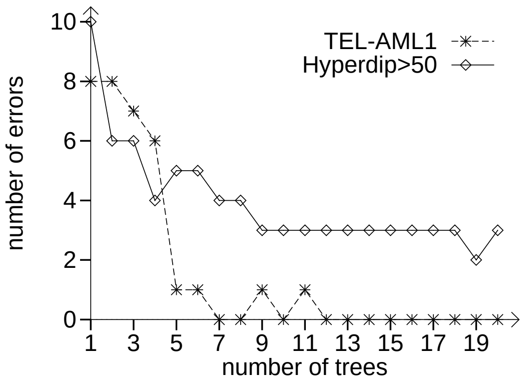


Figure 5.7: Power of ensemble trees in CS4 — number of combined trees versus number of misclassified testing samples.

be rectified. Single coverage constraint means every training sample is covered by exactly one rule [66].

Secondly, we examine the performance of combining these single trees. For the test on TEL-AML1 versus OTHERS, if we combine the first four trees, CS4 will make 6 mistakes; if we combine the first five trees, the number of mistake predictions drops to 1 and to 0 when using the first seven trees. In Figure 5.7, we plot the curves of number of combined trees versus number of misclassified testing samples for TEL-AML1 and Hyperdip>50 subtypes prediction. The curves show an obvious decreasing trend on the number of testing errors when first several trees are combined to give prediction and after that, the accuracy tends to be stable. Therefore, intuitively, we see the power of using our ensemble of decision trees method CS4. Besides, the two curves also demonstrate that 20 trees are enough to give good prediction.

Table 5.26: Comparison between CS4 and SVM under different feature selection scenarios using the results of the 320 (=2x8x20) experiments on the six gene expression profiles and one proteomic data set. Symbol "A" stands for SVM classifier and "F" for CS4. Each cell indicates the symbol(s) of the classifier(s) that achieved best performance in the relevant experiment under corresponding feature selection scenario. For each feature selection scenario, the last row indicates the total number of experiments in which SVM performed better, CS4 did better and the tie case. If we add up the numbers across feature selection scenarios, the final result is — SVM won 86, CS4 won 22, and tie 52.

Experiment	All	All-entropy	Mean-entropy	Top-number-entropy				ERCOF
				20	50	100	200	
ColonTumor	A	A	A	A	A	A	A	A
Prostate	A	A	A	A	A,F	A	A	A
Lung test	A	A	A	A	A	A	A	A
Lung	A	A	A,F	A	F	A,F	A	A,F
Ovarian	A,F	A,F	A	A	A	A,F	A,F	A,F
DLBCL	F	A	A	A,F	A	A	A	A
ALLAML test	F	A	A	F	F	A	A	A
ALLAML	A,F	A,F	A,F	A	F	F	A,F	A,F
			Pediatric ALL data — test					
T-ALL	A,F	A,F	A,F	A,F	A,F	A,F	A,F	A,F
E2A-PBX1	A,F	A,F	A,F	A,F	A,F	A,F	A,F	A,F
TEL-AML1	F	A	A	A	A,F	A,F	A,F	A,F
BCR-ABL	A	A	F	F	F	A,F	A,F	A,F
MLL	A,F	A,F	F	A,F	A,F	F	A,F	A,F
Hyperdip>50	F	A	A	A	A	A	A	A
			Pediatric ALL data — 10-fold cross validation					
T-ALL	A,F	A	F	F	A	A	A	A,F
E2A-PBX1	A,F	A,F	A	A	A	A	A	A,F
TEL-AML1	A	A	A	A	F	A	A	A
BCR-ABL	F	F	A	A	F	A	F	A
MLL	A,F	A	A,F	A,F	A	A	A	A
Hyperdip>50	A	A	A	A	A	A	F	A
Sum	A:7	A:13	A:12	A:12	A:9	A:12	A:11	A:10
	F:5	F:1	F:3	F:3	F:6	F:2	F:2	F:0
	Tie:8	Tie:6	Tie:5	Tie:5	Tie:5	Tie:6	Tie:7	Tie:10

Comparison of CS4 with SVM and k-NN

First, we compare CS4 with SVM. Table 5.26 lists the classifier(s) (of SVM and CS4) that achieved best validation accuracy for each experiment. Overall speaking, the performance of SVM is superior to that of CS4.

Secondly, similarly, we compare CS4 with k-NN. Table 5.27 lists the classifier(s) (of k-NN and CS4) that achieved best validation accuracy for each experiment. Among 160 cases, k-NN won 48, CS4 won 55, and tie 57. The performance of CS4 is slightly better than that of k-NN.

SVM is the representative of the classifiers built on kernel functions while k-NN is the most typical instance-based learning algorithm. Different from decision tree methods which use only

Table 5.27: Comparison between CS4 and k-NN under different feature selection scenarios using the results of the 320 (=2x8x20) experiments on the six gene expression profiles and one proteomic data set. Symbol "B" stands for k-NN classifier and "F" for CS4. Each cell indicates the symbol(s) of the classifier(s) that achieved best performance in the relevant experiment under corresponding feature selection scenario. For each feature selection scenario, the last row indicates the total number of experiments in which K-NN performed better, CS4 did better and the tie case. If we add up the numbers across feature selection scenarios, the final result is — k-NN won 48, CS4 won 55, and tie 57.

Experiment	All	All-entropy	Mean-entropy	Top-number-entropy				ERCOF
				20	50	100	200	
ColonTumor	F	B	B	B	B	B	B	B
Prostate	F	F	B,F	F	F	B	B,F	B,F
Lung test	B,F	B	B	B	B	B	B	B
Lung	F	F	F	B	F	B,F	B,F	B,F
Ovarian	F	F	F	B	F	F	F	F
DLBCL	F	B,F	B,F	B	F	B	B,F	B
ALLAML test	F	F	B	F	B,F	B	B	B
ALLAML	F	B,F	B	F	F	F	B,F	B,F
Pediatric ALL data — test								
T-ALL	F	B,F	B,F	B,F	B,F	B,F	B,F	B,F
E2A-PBX1	B,F	B,F	B,F	B,F	B,F	B,F	B,F	B,F
TEL-AML1	F	B	B	B	B,F	B,F	B,F	B,F
BCR-ABL	B,F	F	F	B,F	F	F	F	F
MLL	F	B,F	B,F	B,F	F	B,F	B,F	B,F
Hyperdip>50	B,F	B	B	B	B	B,F	B,F	B,F
Pediatric ALL data — 10-fold cross validation								
T-ALL	F	F	B	F	F	B	B	F
E2A-PBX1	B,F	B,F	B,F	B	B	B,F	B,F	B,F
TEL-AML1	F	B	B	B	F	F	B	B
BCR-ABL	F	F	F	F	F	F	F	B,F
MLL	F	B,F	B	F	B	F	B,F	B,F
Hyperdip>50	F	B	F	B	B	B,F	B,F	B
Sum	B:0	B:6	B:9	B:10	B:6	B:6	B:5	B:6
	F:15	F:7	F:5	F:6	F:10	F:6	F:3	F:3
	Tie:5	Tie:7	Tie:6	Tie:4	Tie:4	Tie:8	Tie:12	Tie:11

a small subset of features, SVM and k-NN involve all the features in their classification models. Although the overall performance of SVM is better than that of CS4, prediction models built by SVM are difficult to understand, interpret and apply to practical disease diagnosis. In this aspect, CS4 has its big advantage over SVM.

SVM — linear versus quadratic kernel

"Will quadratic polynomial kernel functions perform better?" To answer this question, we apply SVM with the quadratic polynomial kernel function to the data sets. The results show that in most of cases, SVM with quadratic kernel function performs the same as that with the simple linear kernel; and in some cases, it even does worse. For the experiments described in this chapter, quadratic kernel seldom performs better than linear kernel. For example, among the twenty experiments using ERCOF selected features, linear kernel achieved better accuracy in 7 of them while they tied in the rest 13 cases (detailed data is not shown). Note that, quadratic kernels need much more time on training process, especially for high-dimensional data.

5.3.2 Feature selection methods

The experimental results show that for all the classifiers, in most of cases, they performed better (or not worse) with the selected features than they did with the original feature space. In the following discussions, we will focus on comparing our ERCOF with other proposed entropy-based feature selection methods.

Comparison of ERCOF with all-entropy and mean-entropy

Since ERCOF is built on all-entropy (the Phase I feature filtering of ERCOF), first, let's compare the performance of ERCOF and all-entropy. Table 5.28 lists the feature selection method(s) (of ERCOF and all-entropy) that achieved best validation accuracy for each experiment. Among 120 cases, all-entropy won 4, ERCOF won 60, and tie 56. Obviously, the performance of ERCOF is better than that of all-entropy.

In our previous work presented in [64], mean-entropy method was claimed to be superior to all-entropy on high-dimensional biomedical data. Next, we will compare the performance of ERCOF with mean-entropy. Table 5.29 lists the feature selection method(s) (of ERCOF and mean-entropy) that achieved best validation accuracy for each experiment. Among 120 cases, mean-entropy won 18, ERCOF won 42, and tie 60. Overall speaking, the performance of ERCOF is better than that of mean-entropy, though among 50% of cases they had equal performance.

Note that, compared with all-entropy, mean-entropy and ERCOF use fewer features. To have an intuitive sense of amount of features selected by these three methods, in Table 5.30 we list number of features in the original data sets as well as in the dimensional-reduced data sets. From the table, we can see that:

(1) Feature reduction is mostly done by the entropy measure. As our "base" selection method, entropy measure (i.e. All-entropy) on the average filters out as many as 88.5% (=1-11.5%)

Table 5.28: Comparison between ERCOF and all-entropy under six different classifiers using the results of the 240 (=2x6x20) experiments on the six gene expression profiles and one proteomic data set. Symbol "A" stands for feature selection using all-entropy method and "C" for ERCOF. Each cell indicates the symbol(s) of the feature selection method(s) that achieved minimum number of misclassified samples in the relevant experiment using relevant classifier. For each classifier, the last row indicates the total number of experiments in which all-entropy performed better, ERCOF did better and the tie case. If we add up the numbers across classifiers, the final result is — all-entropy won 4, ERCOF won 60, and tie 56.

Experiment	SVM	3-NN	Bagging	AdaBoostM1	RandomForests	CS4
ColonTumor	C	A,C	C	C	C	C
Prostate	C	C	A,C	A,C	C	C
Lung test	C	A,C	A	A,C	C	A,C
Lung	A,C	C	A,C	C	C	C
Ovarian	A,C	C	A,C	C	C	A,C
DLBCL	C	C	A	C	A	A,C
ALLAML test	A,C	C	A,C	A,C	C	C
ALLAML	A,C	A,C	A,C	C	A,C	A,C
Pediatric ALL data — test						
T-ALL	A,C	A,C	A,C	A,C	A,C	A,C
E2A-PBX1	A,C	A,C	A,C	A,C	A,C	A,C
TEL-AML1	A,C	A,C	A,C	A,C	A,C	C
BCR-ABL	A,C	C	A,C	A,C	C	A,C
MLL	A,C	A,C	C	A,C	C	C
Hyperdip>50	A,C	A	A,C	C	C	C
Pediatric ALL data — 10-fold cross validation						
T-ALL	A,C	C	A,C	A,C	A,C	C
E2A-PBX1	C	C	A,C	A,C	C	C
TEL-AML1	C	C	C	C	C	C
BCR-ABL	C	C	C	C	C	A,C
MLL	A,C	C	C	C	C	C
Hyperdip>50	C	C	A,C	C	C	A,C
Sum	A:0	A:1	A:2	A:0	A:1	A:0
	C:8	C:12	C:5	C:10	C:14	C:11
	Tie:12	Tie:7	Tie:13	Tie:10	Tie:5	Tie:9

of the features in original data. From the above performance analysis, this round of heavy dimensionality reduction not only brings us much faster speed of classification, but also leads to more accurate predictions.

(2) During the second phase of filtering in ERCOF, 33% of all-entropy selected features are further removed by Wilcoxon rank sum test. After this round of narrow down, the remaining features become sharply discriminating.

(3) After the correlation checking in Phase III, the final ERCOF keeps only 4.5% representative features of original data. This means that 40% of the highly correlated features left in Phase II are deducted in this round of filtering.

Table 5.29: Comparison between ERCOF and mean-entropy under six different classifiers using the results of the 240 (=2x6x20) experiments on the six gene expression profiles and one proteomic data set. Symbol "B" stands for feature selection using mean-entropy method and "C" for ERCOF. Each cell indicates the symbol(s) of the feature selection method(s) that achieved minimum number of misclassified samples in the relevant experiment using relevant classifier. For each classifier, the last row indicates the total number of experiments in which mean-entropy performed better, ERCOF did better and the tie case. If we add up the numbers across classifiers, the final result is — mean-entropy won 18, ERCOF won 42, and tie 60.

Experiment	SVM	3-NN	Bagging	AdaBoostM1	RandomForests	CS4
ColonTumor	C	C	B,C	C	C	C
Prostate	C	B,C	C	B	C	B,C
Lung test	B,C	B,C	B	B,C	C	B,C
Lung	B,C	C	B,C	B	B	B,C
Ovarian	B,C	C	B	C	B,C	C
DLBCL	B,C	C	B	B,C	C	B,C
ALLAML test	B,C	C	B,C	B,C	C	B,C
ALLAML	B,C	B	B	C	B	B,C
Pediatric ALL data — test						
T-ALL	B,C	B,C	B,C	B,C	B,C	B,C
E2A-PBX1	B,C	B,C	B,C	B,C	B,C	B,C
TEL-AML1	B,C	B,C	B,C	B	C	C
BCR-ABL	C	B,C	B	B,C	B,C	B,C
MLL	B,C	B,C	B,C	B,C	B,C	B,C
Hyperdip>50	B,C	B	B	B,C	C	B,C
Pediatric ALL data — 10-fold cross validation						
T-ALL	B,C	B	B,C	B,C	B,C	B,C
E2A-PBX1	C	C	B,C	B,C	C	C
TEL-AML1	C	C	B,C	C	C	C
BCR-ABL	C	C	C	B	B	B
MLL	B,C	B,C	C	C	B	C
Hyperdip>50	C	C	C	B,C	C	C
Sum	B:0	B:3	B:6	B:4	B:4	B:1
	C:7	C:9	C:4	C:5	C:10	C:7
	Tie:13	Tie:8	Tie:10	Tie:11	Tie:6	Tie:12

(4) Number of features selected by mean-entropy is very close to that by ERCOF — only 40% of all-entropy selected features are kept. Note that, overall speaking, both mean-entropy and ERCOF performed better than all-entropy did.

Comparison of ERCOF with top-number-entropy

For each data set, we also did experiments on some numbers of top features selected by entropy measure. Here, for each test, we will pick up the best one of them to compare with ERCOF. Table 5.31 shows that among 120 comparisons, ERCOF won 17, the best top-number-entropy won 45, and they did equally good in more than 50% of them. However, the best top-number-entropy is different from data to data and from classifier to classifier. There is no regular pattern

Table 5.30: Number of features selected by each method. For a cross validation, the average number of the selected features in each fold's test is used. Column *All* gives the number of features in the original intact data. Under ERCOF, the number of remaining features after Wilcoxon rank sum test (i.e. Phase II) is given in Column *after RST* while the final number of selected features (i.e. Phase III) is in Column *Final*. The percentage of the selected features on original feature space is indicated in the brackets. The last row *Average* is the average percentage across the total 20 tests.

Experiment	All	All-entropy	Mean-entropy	ERCOF	
				after RST	Final
ColonTumor	2000	131(6.6%)	58(2.9%)	77(3.9%)	58(2.9%)
Prostate	12600	1429(11.3%)	528(4.2%)	963(7.6%)	516(4.1%)
Lung test	12533	2173(17.3%)	777(6.2%)	1116(8.9%)	673(5.4%)
Lung	12533	4530(36.1%)	1747(13.9%)	3169(25.3%)	1728(13.8%)
Ovarian	15154	5930(39.1%)	2752(18.2%)	4016(26.5%)	2847(18.8%)
DLBCL	4026	392(9.7%)	141(3.5%)	199(4.9%)	112(2.8%)
ALLAML test	7129	866(12.1%)	350(4.9%)	519(7.3%)	280(3.9%)
ALLAML	7129	890(12.5%)	397(5.6%)	618(8.7%)	322(4.5%)
Pediatric ALL data — test					
T-ALL	12558	1309(10.4%)	415(3.3%)	869(6.9%)	458(3.7%)
E2A-PBX1	12558	718(5.7%)	235(1.9%)	404(3.2%)	235(1.9%)
TEL-AML1	12558	1309(10.4%)	427(3.4%)	721(5.7%)	461(3.7%)
BCR-ABL	12558	84(0.6%)	31(0.2%)	84(0.6%)	76(0.6%)
MLL	12558	327(2.6%)	124(0.9%)	147(1.2%)	86(0.6%)
Hyperdip>50	12558	914(7.3%)	328(2.6%)	691(5.5%)	315(2.5%)
Pediatric ALL data — 10-fold cross validation					
T-ALL	12558	1667(13.3%)	731(5.8%)	1329(10.6%)	695(5.5%)
E2A-PBX1	12558	1021(8.1%)	401(3.2%)	604(4.8%)	326(2.6%)
TEL-AML1	12558	1563(12.4%)	698(5.6%)	1351(10.8%)	748(5.6%)
BCR-ABL	12558	147(1.2%)	56(0.5%)	96(0.7%)	50(0.4%)
MLL	12558	519(4.1%)	147(1.2%)	350(2.8%)	196(1.6%)
Hyperdip>50	12558	1222(9.7%)	536(4.3%)	1013(8.1%)	787(6.3%)
Average		11.5%	4.6%	7.7%	4.5%

to follow. To further illustrate this point, for each of six classifiers, in Figure 5.8, we draw the plots of top number of entropy selected features versus number of prediction errors on the testing samples of the ALL-AML leukemia data set and Hyperdip>50 subtype of the pediatric ALL data set. From the plots, there is no optimal number of features can be found.

After above comparisons, we can claim that ERCOF is an efficient way to select features from high-dimensional gene expression data. In Phase I of ERCOF, entropy-based method eliminates those genes who do not separate the samples well, but you have observed that using all-entropy selected genes does not give the best results. This implies that some features with small entropy are not useful. To avoid restricting ourselves to an arbitrary cut off (like top-number-entropy), in Phase II of ERCOF, we resort to a non-parametric statistical test to help decide which of the genes left in Phase I are more relevant than others. The Phase III of ERCOF corre-

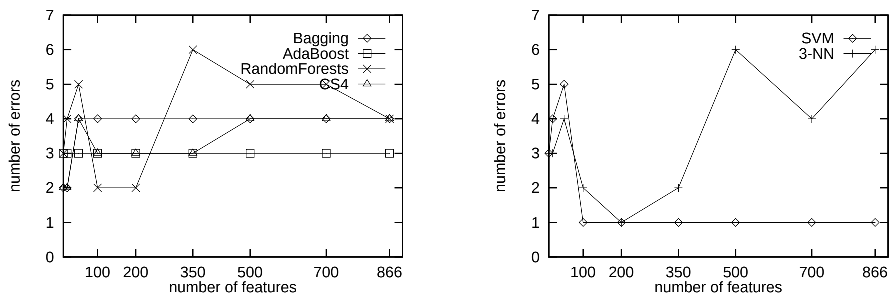
Table 5.31: Comparison between ERCOF and top-number-entropy (i.e. top 20, 50, 100 and 200) under six classifiers using the results of the 600 (=5x6x20) experiments on the six gene expression profiles and one proteomic data set. Symbol "C" stands for ERCOF and "D" for the best feature selection of top-number-entropy. Each cell indicates the symbol(s) of the feature selection method(s) that achieved best performance in the relevant experiment using relevant classifier. For each classifier, the last row indicates the total number of experiments in which ERCOF performed better, top-number-entropy did better and the tie case. If we add up the numbers across classifiers, the final result is — ERCOF won 17, the best top-number-entropy won 45, and tie 58.

Experiment	SVM	3-NN	Bagging	AdaBoostM1	RandomForests	CS4
ColonTumor	C,D	C	C	C	D	D
Prostate	C,D	D	D	D	D	D
Lung test	C,D	D	C,D	C,D	C,D	C,D
Lung	C,D	C,D	D	D	C,D	C,D
Ovarian	C,D	D	D	D	D	C,D
DLBCL	C	D	D	D	C	D
ALLAML test	C,D	C,D	D	C,D	D	D
ALLAML	C,D	C,D	D	D	D	D
Pediatric ALL data — test						
T-ALL	C,D	C,D	C,D	C,D	C,D	C,D
E2A-PBX1	C,D	C,D	C,D	C,D	C,D	C,D
TEL-AML1	C	C	D	D	C,D	C
BCR-ABL	C,D	D	D	C,D	D	C,D
MLL	C,D	C,D	C,D	C,D	D	C,D
Hyperdip>50	C,D	D	C,D	D	C	D
Pediatric ALL data — 10-fold cross validation						
T-ALL	C,D	C,D	C,D	C,D	C,D	C,D
E2A-PBX1	C,D	C,D	C,D	C,D	C,D	C
TEL-AML1	C	C	C,D	D	C,D	C,D
BCR-ABL	D	C	C,D	D	D	D
MLL	D	C	C	D	D	C
Hyperdip>50	C	C,D	D	C,D	C	D
Sum	C:4	C:5	C:2	C:1	C:2	C:3
	D:2	D:7	D:9	D:10	D:9	D:8
	Tie:14	Tie:8	Tie:9	Tie:9	Tie:9	Tie:9

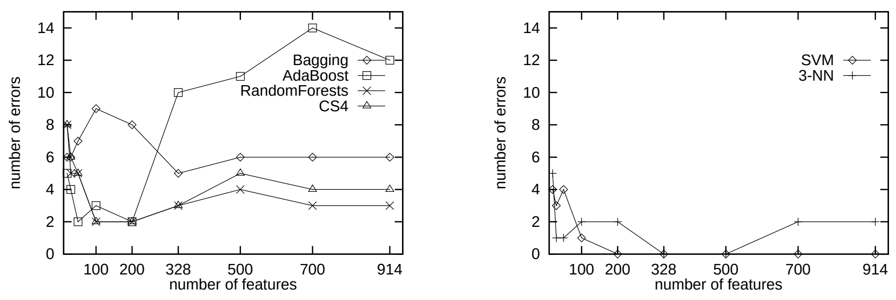
sponds to some biological considerations — sorting out the features into pathways and for each pathway, picking out sufficient number of genes to represent that pathway.

5.3.3 Classifiers versus feature selection

Here, we will discuss two issues: (1) which feature selection method is in favour of which classification algorithm, and (2) sensitivity of the classifiers to the feature selection methods. To have an overall and intuitive feeling of the relationship between the performance of classifiers and feature selection methods, for each of the classifiers, we count for each of the feature selection methods (including *All* where all features were used) the total number of winning times and



(A) ALL-AML data



(B) Hyperdip>50 subtype of pediatric ALL data

Figure 5.8: Plots of top number of features versus number of errors made on testing samples of (A) ALL-AML leukemia data and (B) Hyperdip>50 subtype of pediatric ALL data. In (A), mean-entropy and all-entropy selected 350 and 866 features from training data, respectively. In (B), mean-entropy and all-entropy selected 328 and 914 features, respectively. The two plots on the left side are drawn for four ensemble of decision trees classifiers while the two on the right side are for SVM and 3-NN.

misclassified samples across the 20 validation tests on the six gene expression profiles and one proteomic data set. The results are summarized in Table 5.32.

Now, we start to address the first issue. In terms of both total winning times and number of misclassified samples, among eight feature selection methods, ERCOF is the best for SVM, 3-NN, Random forests and CS4. Besides, under ERCOF, Bagging achieved its smallest total number of misclassified samples. AdaBoostM1 performed relatively better using top 20 features selected by entropy measure, but compared with the other five classifiers, its performance is not good. As mentioned earlier, the main reason might be that boosting loses its power of using multiple decision trees when the single C4.5 tree has no error on training samples. When this happens, boosting is equivalent to the single tree C4.5 method. Unfortunately, in high-dimensional data, we often see that single C4.5 trees have perfect training accuracy. A special case is that there are features having zero entropy value on training samples, such as lung cancer data and T-ALL,

Table 5.32: A summary of the total winning times (including tie cases) of each classifier (under different feature selection methods) across the 20 validation tests on the six gene expression profiles and one proteomic data set. The number with bold font in each row indicates the feature selection method that owns most winning times for the relevant classifier. In the brackets, there is the total number of misclassified samples across the same 20 validation tests. Similarly, the figure with bold font in the brackets in each row is the minimum number of total misclassified samples among feature selection methods for the classifier.

Classifier	All	All-entropy	Mean-entropy	Top-number-entropy				ERCOF
				20	50	100	200	
SVM	4(100)	9(52)	11(48)	6(76)	6(74)	11(52)	11(59)	**16**(**38**)
3-NN	1(187)	5(87)	8(77)	6(88)	4(81)	6(77)	5(73)	**12**(**61**)
Bagging	7(123)	5(117)	8(115)	**11**(123)	**11**(122)	7(122)	9(114)	8(**112**)
AdaBoostM1	5(191)	8(181)	8(166)	**11**(**138**)	10(144)	10(157)	9(162)	10(154)
RandomForests	0(228)	5(111)	5(93)	6(96)	7(83)	8(96)	5(90)	**9**(**80**)
CS4	5(87)	6(77)	6(76)	7(101)	10(81)	9(74)	8(74)	**12**(**66**)

E2A-PBX1 subtypes of pediatric ALL data.

Let's move to the second issue. From Table 5.32, we observe that some classifiers are sensitive to the feature selection. The good examples are SVM and k-NN — their classification performances were improved significantly by using selected features; however, on the other hand, they could not achieve good performance either if the feature space is too small. Thus, feature selection is important to SVM and k-NN when dealing with high-dimensional biomedical data. This conclusion is in consistent with the principles of the both algorithms that all the features are used in the classification models. Again, ERCOF is a suitable feature selection method for these two classifiers. Different from SVM and k-NN, decision tree methods do not use all the input features in their classification models (i.e. decision trees) so that they are relatively not sensitive to the feature selection. For example, Bagging and CS4 performed quite reasonably well on the original intact data. As illustrated in Table 5.25, a decision tree often contains very few number of features, say around 5 in each tree. We called these features as *built-in features* [65]. The selection of these built-in features is dependent on the individual decision tree algorithms. For example, information gain ratio measure is employed by C4.5, our base decision tree classifier. This round of feature selection conducted by the classifier itself might be one of the main reasons that classifiers based on decision tree are relatively resistant to other "pre-feature-selections". Nevertheless, properly selecting features can also help improve the performance of ensemble of decision trees methods — Random forests and CS4 achieved very good classification results using ERCOF selected features.

5.4 Chapter Summary

In this chapter, we applied some entropy-based feature selection methods and classification techniques to six gene expression profiles and one proteomic data. These data sets are described by at least 2000 features and some of them are by more than 12,000 features. For each data set, we carried out various experiments and compared our results with the published ones (where available). The large amount of experimental results showed that in most of cases, our proposed methods achieved comparable or better classification performance than those previously reported. Besides, we also listed the good features (i.e. genes for all the data sets except ovarian disease) identified by our method, compared them with literature, and related some of them with the disease studied. To emphasize the advantages of decision trees methods in bio-medical domain, we presented many simple, explicit and comprehensible trees learned from the data.

We also addressed various comparisons among classifiers and feature selection methods. In the aspect of classifiers, SVM demonstrated its power on classification accuracy and our ensemble of decision trees method CS4 also achieved good results. Among the decision tree methods, the performance of CS4 is superior to Bagging, AdaBoostM1 and Random forests. The main advantage of CS4 over SVM is that its output is easy to be interpreted and applied to practical disease diagnosis. We also clearly observed the performance improvements of all the classifiers under the proposed feature selection scenarios. Among the various entropy-based feature selection methods, ERCOF has demonstrated its efficiency and robustness when dealing with high-dimensional gene expression data.

Chapter 6

Experiments on Microarray Data — Patient Survival Prediction

In this chapter, a new computational process for patient survival prediction using microarray gene expression data will be presented. Different from all previous works, in the first step, we carefully form the training set samples by selecting only *short-term survivors* who died within a short period and *long-term survivors* who were still alive after a relatively long follow-up time. This idea is motivated by our belief that short-term and long-term survivors are more informative and reliable (than those cases in between) for building and understanding the relationship between genes and patient survival. In the second step, ERCOF is used to identify genes most associated with survival. In the third step, a linear kernel support vector machine (SVM) is trained on the selected samples and genes to build a scoring model. The model assigns each validation sample a risk score to predict patient survival.

6.1 Methods

We will describe in detail the new method for patient survival prediction, focusing on selecting an informative subset of training samples and building SVM-based scoring function.

6.1.1 Selection of informative training samples

One of the main features of our new method is to select informative training samples. Since our focus is on the relationship between gene expression and survival, the survival time associated with each sample plays an important role here — two types of extreme cases, patients who died in a short period (termed as "*short-term survivors*") and who were alive after a long period (termed as "*long-term survivors*"), should be more valuable than those in the "middle" status. Thus, we use only a part of samples in training and this is clearly different from other approaches that use all training samples.

Formally, for a sample T, if its follow-up time is $F(T)$ and its status at the end of follow-up time is $E(T)$, then

$$T\ is \begin{cases} \text{short-term survivor,} & \text{if } F(T) < c_1 \wedge E(T) = 1 \\ \text{long-term survivor,} & \text{if } F(T) > c_2 \\ \text{others,} & \text{otherwise} \end{cases} \tag{6.1}$$

$E(T) = 1$ stands for "dead" or an unfavorable outcome, $E(T) = 0$ stands for "alive" or a favorable outcome, c_1 and c_2 are two thresholds of survival time for selecting short-term and long-term survivors. Note that long-term survivors also include those patients who died after the specified long period.

The two thresholds, c_1 and c_2, can vary from disease to disease, from data set to data set. For example, in the survival study of early-stage lung adenocarcinomas that will be presented later, we choose short-term survivors as those who died within one follow-up year (i.e. c_1 is 1 year) and long-term survivors as those who were alive after five follow-up years (i.e. c_2 is 5 years). There are total 31 extreme training samples (10 short-term survivors and 21 long-term survivors) among a total of 86 available primary lung adenocarcinomas. These 21 long-term survivors include 2 patients whose status at the end of follow-up time was "dead", but follow-up times were 79.5 months and 84.1 months, respectively. Our basic guide lines for the selection of c_1 and c_2 are that the informative subset should (1) contain enough training samples for learning algorithms to learn (typically $>$15 samples in each class and total is between one third and one half of all available samples), but (2) not have too many samples to avoid including non-extreme cases.

After choosing informative training samples, we apply ERCOF to them to identify genes most associated with survival status. With the selected samples and genes, in the next step, we will build a scoring function to estimate the survival risk for every patient.

6.1.2 Construction of an SVM scoring function

The regression scoring function proposed for survival risk estimation is based on support vector machines (SVM) described in Section 2.3.2 of Chapter 2. Recall that the final discriminant function $f(T)$ for a test sample T given in Formula (2.3) of Chapter 2. If the linear kernel function is used, $f(T)$ will become a linear combination of the expression values of the identified genes. In this study, we map class label of "short-term survivors" to 1 and "long-term survivors" to -1. Note that $f(T) > 0$ if the sample T is more likely to be a "short-term survivor", and $f(T) < 0$ if the sample T is more likely to be a "long-term survivor".

To normalize $f(T)$, we use a transformation function $s(T)$ defined as:

$$s(T) = \frac{1}{1 + e^{-f(T)}} \tag{6.2}$$

Thus, $f(T)$ is normalized by $s(T)$ into the range $(0, 1)$. Note that the smaller the $s(T)$ value is, the better survival the patient corresponding to sample T will have. We term $s(T)$ the risk score of T.

If one only categorizes patients into high risk or low risk groups, the value 0.5 is a natural cutoff for $s(T)$, where if $s(T) > 0.5$ then the patient corresponding to sample T will have higher risk; otherwise, the patient will have lower risk. If more than two risk groups are considered — such as high, intermediate, and low — then other cutoffs can be set based on the risk scores of training samples. E.g., in training set, if most of short-term survivors have a risk score greater than r_1 and most of long-term survivors have a risk score smaller than r_2, then,

$$T\ is \begin{cases} \text{high risk,} & \text{if } s(T) > r_1 \\ \text{low risk,} & \text{if } s(T) < r_2 \\ \text{intermediate risk,} & \text{if } r_2 \leq s(T) \leq r_1 \end{cases} \tag{6.3}$$

Generally, $r_1 > 0.5$, $r_2 < 0.5$, and they can be derived from the risk scores assigned to the training samples.

To evaluate the results, after assigning patients into different risk groups, we draw Kaplan-

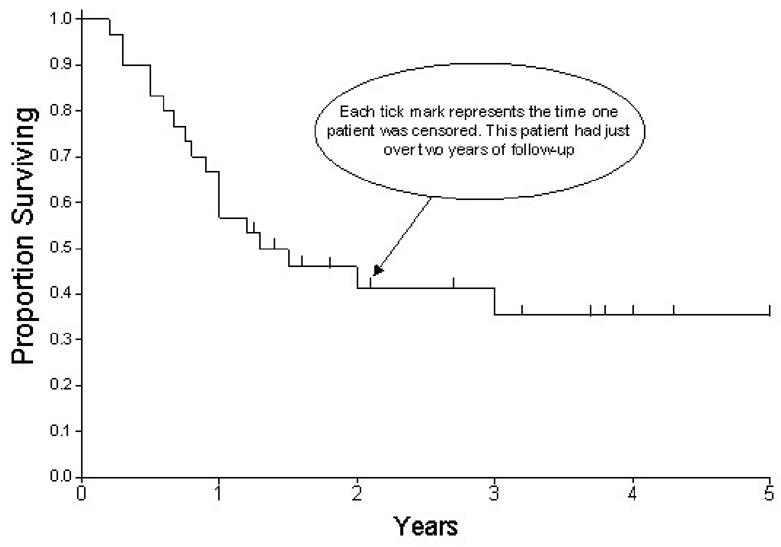
Meier plots [8] to compare the survival characteristics between groups.

6.1.3 Kaplan-Meier analysis

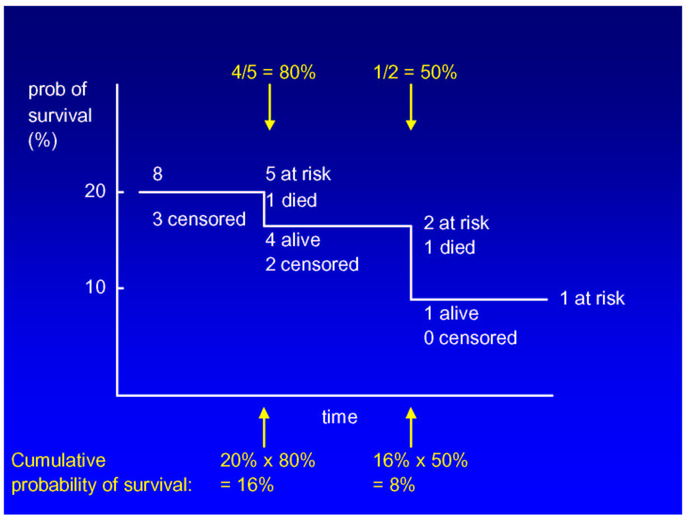
Kaplan-Meier analysis estimates a population survival curve from a set of samples. A survival curve illustrates the fraction (or percentage) survival at each time. Since in realistic clinical trial it often takes several years to accumulate the patients for the trial, patients being followed for survival will have different starting times. Then the patients will have various length of follow-up time when the results are analysed at one time. Therefore, the survival curve can not be estimated simply by calculating the fraction surviving at each time. For example, in the following study of lung adenocarcinomas, the patients follow-up time is varying from 1.5 months to 110.6 months.

A Kaplan-Meier analysis allows estimation of survival over time, even when patients drop out or are studied for different lengths of time [1]. For example, an alive patient with 3 years follow-up time should contribute to the survival data for the first three years of the curve, but not to the part of the curve after that. Thus, this patient should be mathematically removed from the curve at the end of 3 years follow-up time and this is called "censoring" the patient. On a Kaplan-Meier survival curve, when a patient is censored, the curve does not take a step down as it does when a patient dies; instead, a tick mark is generally used to indicate where a patient is censored and each death case after that point will cause a little bit larger step down on the curve. An alternative way to indicate a censored patient is to show the number of remaining cases "at risk" at several time points. Patients who have been censored or died before the time point are not counted as "at risk". In Figure 6.1, picture (A) is a complete sample of Kaplan-Meier survival curve with a tick mark representing a censored patient (captured from `http://www.cancerguide.org/scurve_km.html`), while picture (B) illustrates how to calculate the fraction of survival at a time (captured from [1]).

To compare the survival characteristics between different risk groups for our survival prediction study, we draw Kaplan-Meier survival curves of the risk groups in one picture and use *logrank test* to compare the curves. The logrank test generates a p-value testing the null hypothesis that the survival curves are no difference between two groups. The meaning of p-value is that "if the null hypothesis is true, what is the probability of randomly selecting samples whose survival curves are different from those actually obtained?". In this chapter, all the Kaplan-Meier



(A)



(B)

Figure 6.1: Samples of Kaplan-Meier survival curves. (A) is an example of a Kaplan-Meier survival curve. This group of patients has a minimum follow-up of a little over a year. (B) is an illustration on how to calculate the fraction of survival at a time.

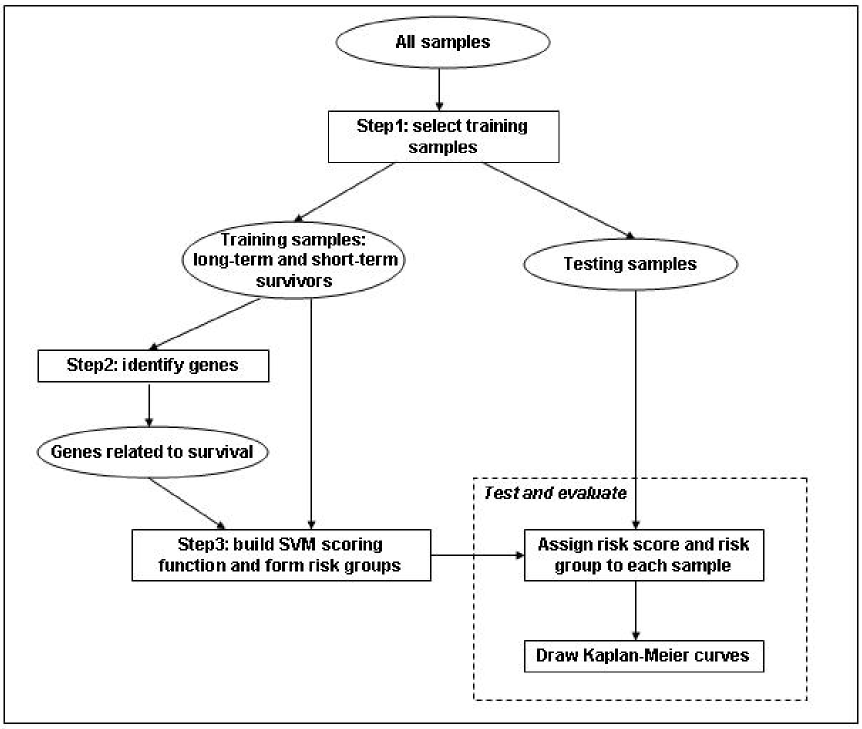

All samples
Step1: select training samples
Training samples: long-term and short-term survivors
Testing samples
Step2: identify genes
Genes related to survival
Test and evaluate
Step3: build SVM scoring function and form risk groups
Assign risk score and risk group to each sample
Draw Kaplan-Meier curves

Figure 6.2: A process diagram of patient survival study, including three training steps as well as testing and results evaluation.

survival curves are generated by *GraphPad Prism* (`http://www.graphpad.com`) and we always indicate the two-tailed p-value. Figure 6.2 shows a diagram of patient survival prediction using our proposed method.

6.2 Experiments and Results

We apply the procedure of survival study above to two gene expression data sets.

6.2.1 Lymphoma

Survival after chemotherapy for diffuse large-B-cell lymphoma (DLBCL) patients was previously studied by Rosenwald *et al* [102] using gene expression profiling and Cox proportional-hazards model. In that study, expression profiles of biopsy samples from 240 patients were used [102]. The data include a preliminary group consisting of 160 patients and a validation group of 80 patients, each of them is described by 7399 microarray features.

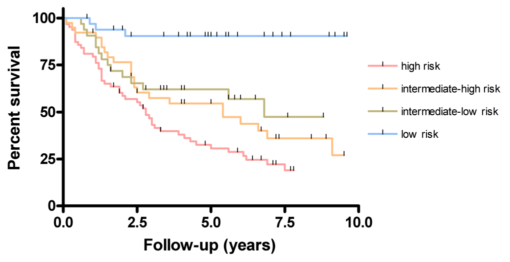
Survival curves showing clear distinction

As an initial step, we pre-process the data to remove those genes that are absent in more than 10% of the experiments in the preliminary group. There remains 4937 features after having 2462 genes removed.

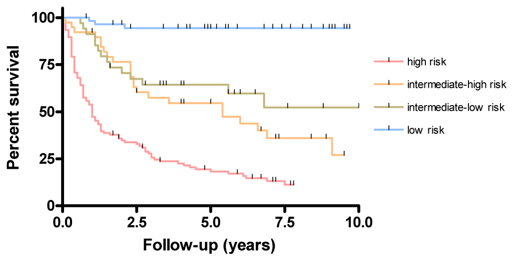
Then, we select short-term survivors and long-term survivors to construct an informative subset of training samples. For this study, we set $c_1 = 1$ year and $c_2 = 8$ years in Formula (6.1). Among the preliminary 160-patient group, 47 short-term survivors (who died within one follow-up year) and 26 long-term survivors (who were alive after eight follow-up years) are thus chosen. So, a total of 73 samples are in this informative subset of training samples (46% of the preliminary group) .

In the second step, we apply ERCOF to these 73 samples and identify 78 genes that are related to patient survival status at 5% significant level (for Wilcoxon rank sum test) and 0.99 Pearson correlation coefficient threshold. Some of our selected genes are also listed in Table 2 of [102], where these genes were found to be significantly associated with survival ($p < 0.01$). E.g., AA805575 (GenBank accession number) is in *germinal-center B-cell signature*, X00452 and M20430 in *MHC class II signature*, and D87071 is in *lymph-node signature*. The gene signatures were formed by a hierarchical clustering algorithm in [102]. Besides, some top-ranked genes (with smaller entropy value) identified by ERCOF are also in one of these gene signatures. E.g., BC012161, AF061729 and U34683 are in *proliferation signature*, BF129543 is in *germinal-center B-cell signature*, and K01144 and M16276 are in *MHC class II signature*.

In the third step, an SVM model is trained on the 73 extreme training samples with the 78 identified features. We find that the well-learned linear kernel SVM can separate the 47 short-term survivors and 26 long-term survivors completely — the lowest risk score assigned to the short-term survivors is above 0.7 and most of the long-term survivors has risk score lower than 0.3. Then, we calculate risk scores for all the other samples, namely the remaining (non-extreme) 87 samples in the original preliminary group and the 80 samples in the validation group. These 167 samples are treated as our test set.



(A) Test cases ($p < 0.0001$)



(B) All cases ($p < 0.0001$)

Figure 6.3: Kaplan-Meier plots illustrate the estimation of overall survival among different risk DLBCL patients in the testing set containing 167 samples (Panel (A)) and all 240 samples (Panel (B)). The risk groups are formed on our SVM-based scoring function. A tick mark on the plot indicates that one sample is censored at the corresponding time. The 5-year overall survival for high risk versus low risk groups of patients for testing samples is 32% versus 91%, for all samples is 20% versus 95%.

We categorized patients into four risk groups as follows:

$$T\ is \begin{cases} \text{high risk,} & \text{if } S(T) > 0.7 \\ \text{intermediate-high risk,} & \text{if } 0.5 < S(T) \leq 0.7 \\ \text{intermediate-low risk,} & \text{if } 0.3 \leq S(T) \leq 0.5 \\ \text{low risk,} & \text{if } S(T) < 0.3 \end{cases} \tag{6.4}$$

where the threshold 0.5 is the mean value of 0.7 and 0.3. The Kaplan-Meier curves of overall survival are drawn in Figure 6.3, where we can see clear differences at the five-year survival rates for the high risk and low risk groups, in both testing sample set (Panel (A)) and all samples (Panel (B)). Although we cannot see distinct overall survival between the two intermediate groups, the 5-year survival rates of these two groups are obviously different from that in the high risk group or the low risk group. This also suggests that three or two risk groups would be sufficient for these DLBCL samples. So in the rest of this study, we simply merge high and intermediate-high risk patients into a single high risk category, and low and intermediate-low risk patients into a single low risk category.

Having the risk score, when a new case comes, we will be able to assign it to the corresponding risk group easily. This kind of prediction was not addressed in [102] where the DLBCL patients were ranked by their gene-expression-based outcome-predictor score but divided into several groups with equal number of samples. For an example: 80 samples in the validation group were stratified according to the quartiles of the scores with each of quartiles consisting of 20 patients. With that kind of categorization, one cannot find an explicit measure to evaluate a new case.

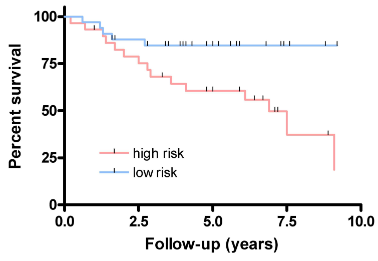
Comparison with International Prognostic Index

Various clinical features — such as stage, performance status, lactate dehydroginase levels — which are known to be strongly related to patient survival, have been combined to form the International Prognostic Index (IPI) [113]. The IPI has been effectively adopted to separate aggressive lymphomas into several groups with significantly different responses to therapy and survival. Since IPI is only built on the consideration of clinical factors, it provides little insight into disease biology [60].

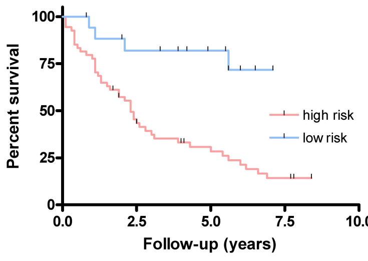
The risk score obtained from our method is based on gene expression in biopsy specimens of the lymphoma, so it is an independent predictor from IPI. In fact, we find that patients in the high IPI group — and similarly for the intermediate and the low IPI groups — when partitioned by our risk score into high risk and low risk categories, have significantly different outcomes. In Figure 6.4, Kaplan-Meier plots show significant difference on overall survival for our high risk and low risk groups among the patients with IPI low (and similarly for intermediate and high) risk index. In particular, among 21 IPI high risk patients in our testing set, 15 of them are assigned by our method to the high risk category and 6 of them to the low risk category. When we check the survival status of these patients, we find 14 of the 15 patients belonging to our high risk category are indeed dead while only 2 of the 6 patients belonging to our low risk category are dead. Similarly, for all 32 patients in the whole data set with high IPI, 23 of them (22 dead) are assigned by our method to the high risk category and 9 (5 dead) of them are assigned to low risk category. This suggests that our method may be a more effective predictor of DLBCL survival outcome than the IPI.

6.2.2 Lung adenocarcinoma

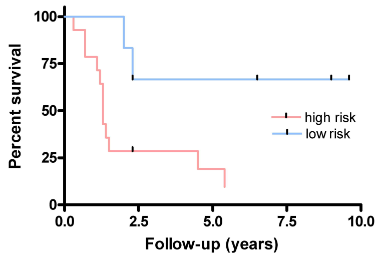
Adenocarcinoma is the major histological subtype of non-small cell lung cancer (NSCLC). There is a need to better predict tumor progression and clinical outcome in lung adenocarcinoma. The lung adenocarcinoma data set contains 86 primary lung adenocarcinomas. These experiments include 67 stage I and 19 stage III tumors, each of them is described by 7129 genes. The data set was first analysed in [14] where a risk index was derived based on the top 50 good genes



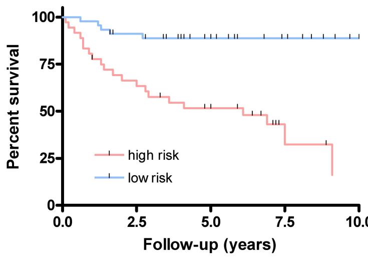
(A) IPI low - test cases ($p = 0.0063$)



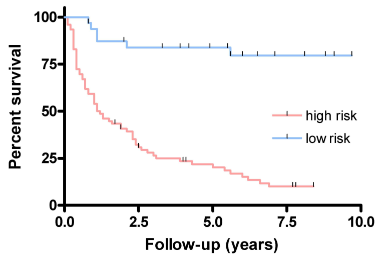
(B) IPI intermediate - test cases ($p = 0.0003$)



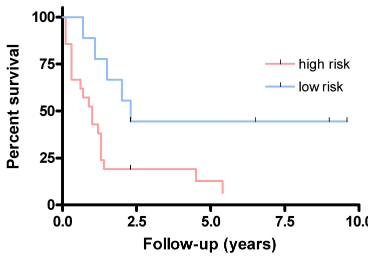
(C) IPI high - test cases ($p = 0.0195$)



(D) IPI low - all cases ($p < 0.0001$)

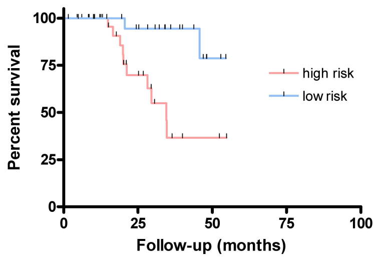


(E) IPI intermediate - all cases ($p < 0.0001$)

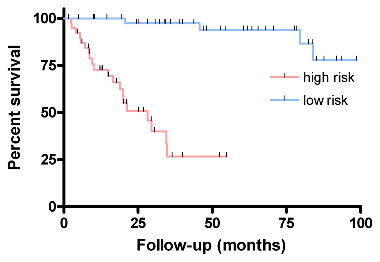


(F) IPI high - all cases ($p = 0.0182$)

Figure 6.4: Kaplan-Meier Estimates of survival among high risk and low risk DLBCL patients (according to our method) in each IPI defined group. Plots (A), (B) and (C) are based on 167 testing samples while (D), (E) and (F) are for all 240 cases.



(A) Test cases ($p = 0.0036$)



(B) All cases ($p < 0.0001$)

Figure 6.5: Kaplan-Meier plots illustrate the estimation of overall survival among high risk and low risk lung adenocarcinoma patients in the testing set containing 55 samples (Panel (A)) and all 86 samples (Panel (B)).

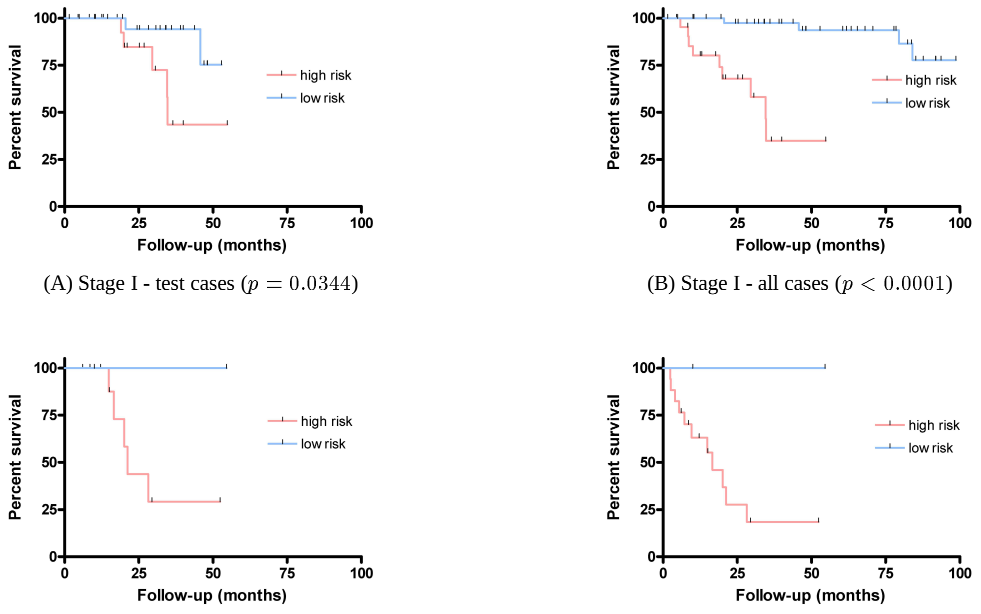
that were identified to be most related to survival by univariate Cox analysis. In that study, tests were conducted by randomly splitting 86 samples into equal sized training and testing sets and by "leave-one-out" cross validation.

First, we form our training set by setting $c_1 = 1$ year and $c_2 = 5$ years in Formula (6.1). 10 short-term survivors and 21 long-term survivors are thus chosen. Applying ERCOF to these 31 training samples, we find 402 genes that are related to outcome. Our top-ranked feature by entropy measure, the ATRX gene, is a putative transcription regulator. It is also reported by Borczuk *et al* in their recent paper [17] on NSCLC. Our second-ranked gene, ENPP2, is part of stress pathways involved in oncogenesis. Yang *et al* [138] also detected it in NSCLC.

Then we train a linear kernel SVM to obtain the weight for each identified gene based on the training data. The trained SVM can separate these 31 samples very well, assigning very high risk scores to short-term survivors (lowest score is as high as 0.73) while very low risk scores to long-term survivors (highest score is as low as 0.25).

After training, we calculate risk score for each of the remaining 55 samples which are used for test purpose. These samples are then classified as high risk group consisting samples T with $s(T) > 0.5$, or as low risk group consisting samples T with $s(T) \leq 0.5$. The Kaplan-Meier curves in Figure 6.5 show clear difference of survival for patients in our high and low risk groups for both testing cases and all cases. Since we pick out all short-term and long-term survivors to form the training set, there is no "death" event happened in the first 12 months time and no sample censored after 60 months time in the plot drawn only on the test cases (Panel (A)).

In order to understand the relationship between our prediction and tumor stage (I or III). We



(A) Stage I - test cases ($p = 0.0344$)

(B) Stage I - all cases ($p < 0.0001$)

(C) Stage III - test cases ($p = 0.3007$)

(D) Stage III - all cases ($p = 0.1622$)

Figure 6.6: Kaplan-Meier plots illustrate the estimation of overall survival among high risk and low risk lung adenocarcinoma patients conditional on tumor stage.

also draw Kaplan-Meier curves to delineate survival difference between our high and low risk patients conditioned on tumor stage. From Figure 6.6, we can see that outcomes of patients with stage I lung adenocarcinoma in our high and low risk groups differ from each other, for both test cases (Panel(A)) and all cases (Panel(B)). Again remarkably, for 13 stage III cases in the testing set, we assigned 11 (5 dead, 6 alive) of them to high risk group, and the 2 of them assigned to low risk group were all alive at the end of the follow-up time. Among all 19 stage III cases, 17 (11 dead, 6 alive) of them were assigned to high risk group according to our risk score.

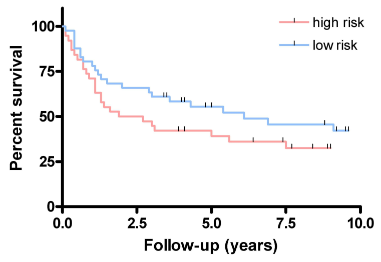
6.3 Discussions

In the step of training set construction, we select only two extreme cases — long-term and short-term survivors. See Table 6.1 for size change trends from the original training samples to the informative training samples on DLBCL and lung adenocarcinomas data sets. The figures illustrate that we used a small part of samples as training.

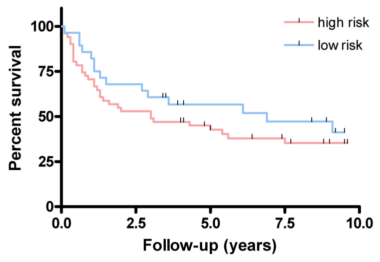
On the other hand, if we do not select those extreme cases, and instead use all available training samples, then what will be the results? To illustrate this, we select genes and train SVM model on the 160 samples in the preliminary group of DLBCL study. Although the training

Table 6.1: Number of samples in original data and selected informative training set. (*): there are 48 DLBCL samples, whose relevant patient was dead at the end of follow-up time, are selected as informative, 47 of them are short-term survivors while 1 of them is long-term survivor. (**): there are 12 lung adenocarcinomas, whose corresponding patients were dead, are selected as informative, 10 of them are short-term survivors while 2 of them are long-term survivors.

Application	Data set	Status		Total
		Dead	Alive	
DLBCL	Original	88	72	160
	Informative	47+1(*)	25	73
Lung adenocarcinoma	Original	24	62	86
	Informative	10+2(**)	19	31



(A) All genes ($p = 0.21$)



(B) ERCOF selected genes ($p = 0.38$)

Figure 6.7: Kaplan-Meier plots illustrate no clear difference on the overall survival among high risk and low risk DLBCL patients formed by the 80 validation samples based on their risk scores that assigned by our regression model built on all 160 training samples. (A) Using all genes. (B) Using genes selected by ERCOF.

accuracy is good, Kaplan-Meier plots do not show significant survival difference between the high and low risk groups formed by the 80 validation samples based on their risk scores that assigned by the trained SVM model. In detail, using all 4937 genes, the p value of the survival curves is 0.21 ((A) in Figure 6.7); using 40 genes selected by ERCOF, the p value is 0.38 ((B) in Figure 6.7). Therefore, we claim that our proposed idea of selecting informative training samples is an effective method.

As pointed out in Section 6.1.1, we have some basic guide lines to determine the thresholds c_1 and c_2 that defined in Formula (6.1). Bearing these minimum constraints in mind, we try several c_1 and c_2 values in our study. In Table 6.2, p-value (of the logrank test) associated with the Kaplan-Meier survival curves of validation samples under different selections of the c_1 and c_2 from DLBCL study are listed. All results are based on ERCOF selected genes. We can see that (1) for a range of c_1 and c_2 (i.e. c_1 less than 3 years and c_2 greater than 8 years), we can

Table 6.2: Results for different thresholds c_1 (years) and c_2 (years) on DLBCL study. All results are based on ERCOF selected genes and on validation samples only.

c_1	c_2	p-value	No. short-term survivors	No. long-term survivors	No. genes
1	5	0.2962	47	57	121
1	7	0.0110	47	36	79
1	8	0.0067	47	26	78
1	9	0.0570	47	22	40
2	8	**0.0049**	61	26	55
3	8	0.0761	76	26	51

achieve better predictions by selecting extreme samples. (2) A better p-value (0.0049) obtained at c_1 =2 years and c_2 =8 years than that we reported in Section 6.2.1 for $c_1 = 1$ year and $c_2 =$ 8 years. However, when we trace back to the risk scores of training samples, one of the long-term survivors selected under c_1 =2 years and c_2 =8 years has a risk score as high as 0.73. In addition, the number of selected short-term survivors is 2.4 times of the number of long-term survivors under this choice. In any case, the selection of c_1 and c_2 can be further refined by running cross-validation on training samples.

In the step of gene identification, built on statistical knowledge, our three-phase filtering process discards many unrelated genes and only keeps a small number of informative representatives. According to our experience on gene expression data analysis, generally, entropy measure can filter out about 90-95% of the total number of genes [64]. This point has been verified again in this study on survival prediction: entropy measure retains only 132 genes in DLBCL study (there are around 5000 genes after removing missing values) and 884 genes in lung adenocarcinoma study (original data contain 7129 genes). After further filtering by Wilcoxon rank sum test and Pearson correlation coefficient test, the final selected genes are with smaller size and less correlated with each other. Table 6.3 shows the number-change trend of features from the entropy selection, to Wilcoxon test, and to correlation coefficient selection. It can be seen that the feature reduction is mostly by the entropy selection.

For comparison, in DLBCL study, we also do experiments using all the 4937 genes, the 132 genes output from the Phase I of ERCOF, and the 84 genes output from the Phase II of ERCOF. The results show that in each of these cases, the overall survival difference between the high and low risk groups formed by our risk scores on the testing samples can be seen as well. In Figure 6.8, we draw the corresponding Kaplan-Meier survival curves. Although the model built

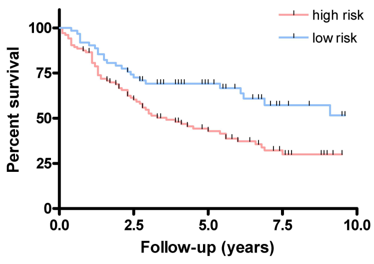
Table 6.3: Number of genes left after feature filtering for each phase of ERCOF. The percentage in the brackets indicates the proportion of the remaining genes on original feature space. (*): the number is after removing genes who were absent in more than 10% of the experiments.

Gene selection	DLBCL	Lung adenocarcinoma
Original	4937(*)	7129
Phase I	132 (2.7%)	884 (12.4%)
Phase II	84 (1.7%)	591 (8.3%)
Phase III	78 (1.6%)	402 (5.6%)

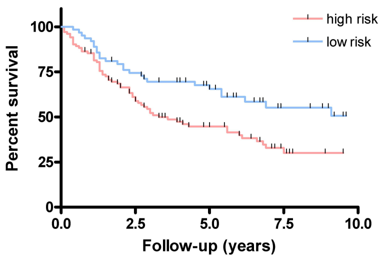
on 3-phase ERCOF makes use the smallest number of genes, it achieves best p value. Again, the good results also demonstrate the effectiveness of selection the informative samples. In addition, in the study of lung adenocarcinoma, using all genes (i.e. without gene selection) cannot predict outcome at all ($p > 0.1$).

In the step of prediction, a simple linear kernel SVM is trained on the selected samples and genes to build a regression model. The model then assigns each validation sample a risk score to predict patient outcome. Based on the training results, we can derive explicit thresholds (e.g., 0.5, 0.3, 0.7) of our risk score to categorize patients into different risk groups. Thus, when a new case comes, we are able to assign it to the corresponding risk group easily according to its risk score. This prediction ability is important in patient survival study.

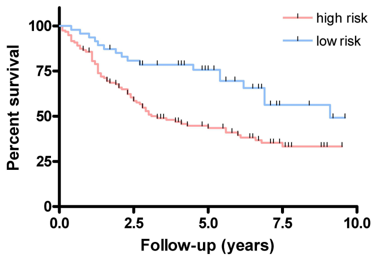
For both studies on DLBCL and lung adenocarcinoma, we associate our results with some clinical features. For example, in the DLBCL study, our high and low risk groups also demonstrate significantly different outcomes in the analysis of patients with low or intermediate risk according to their International Prognostic Index (IPI) scores constructed on some clinical features. E.g., for patients having high IPI, we assign most of them into our high risk category and some into our low risk category, and our assignment is better correlated to survival outcome of these patients. Some of the genes identified to have strong association with survival by ERCOF also fall within four biologic groups defined on the basis of gene expression signatures. In the lung adenocarcinoma study, most of the samples are from stage I tumors. Among these samples, although our high and low risk groups differ significantly from each other, we put quite a few of them into high risk group. This finding "*indicates the important relationship between gene expression profiles and patient survival, independent of disease stage*", which is one of the conclusions drawn in [14].



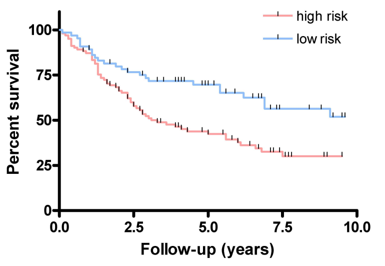
(A) All genes ($p = 0.0095$)



(B) Genes of Phase I ERCOF ($p = 0.0147$)



(C) Genes of Phase II ERCOF ($p = 0.009$)



(D) Genes of Phase III ERCOF ($p = 0.0067$)

Figure 6.8: Kaplan-Meier plots illustrate the estimation of overall survival among high risk and low risk patients in the validation group of DLBCL study. (A) Using all 4937 genes. (B) Using 132 genes output from the Phase I of ERCOF. (C) Using 84 genes output from the Phase II of ERCOF. (D) Using 78 genes output from the Phase III of ERCOF.

6.4 Chapter Summary

In this chapter, we have applied statistical and machine learning technologies to predict patient survival using gene expression profiles. Different from other works, we first picked out extreme cases to form the training set, consisting of only short-term survivors and long-term survivors. Naturally, if there are genes indeed associated with outcome, then the different expression values of these genes should be monitored by analysing these two types of samples. Secondly, ERCOF was applied to the selected informative samples to identify genes most associated with survivals. Thirdly, linear kernel SVM was trained on the selected samples and genes to form a regression model, which can calculate a risk score to each sample. Our proposed methodology was tested on two gene expression profiles: diffuse large-B-cell lymphoma and lung adenocarcinoma. For both studies, the Kaplan-Meier plots showed clear survival difference on high and low risk group patients that formed by the assigned risk scores.

Chapter 7

Recognition of Functional Sites in Biological Sequences

Not all biomedical data contain explicit signals or features as those in the classification problems arised by gene expression profilings. For example, DNA sequences and protein sequences represent the spectrum of biomedical data that possess no explicit features. Generally, a genomic sequence is just a string consisting of the letters "A", "C", "G", and "T" in a "random order". Yet a genomic sequence possesses biologically meaningful functional sites, which play important roles in the process of protein synthesis from DNA sequences. Figure 7.1 shows a picture of this process (captured from the "bioinformatics class notes" of Dr. Nina Rosario L. Rojas at `http://aegis.ateneo.net/nrojas/`). This process can be divided into two stages: transcription and translation.

1. **Transcription**. In this stage, the information in DNA is passed on to RNA. This takes place when one strand of the DNA double helix is used as a template by the RNA polymerase to create a messenger RNA (mRNA). Then this mRNA moves from the nucleus to the cytoplasm. In fact, in the cell nucleus, the DNA with all the exons and introns of the gene is first transcribed into a complementary RNA copy named "nuclear RNA" (nRNA). This is indicated as "primary transcription" in the picture of Figure 7.1. Secondly, non-coding sequences of base pairs (introns) are eliminated from the coding sequences (exons) by RNA *splicing*. The resulting mRNA is the edited sequence of nRNA after splicing. The coding mRNA sequence can be described in terms of a unit of three nucleotides called a

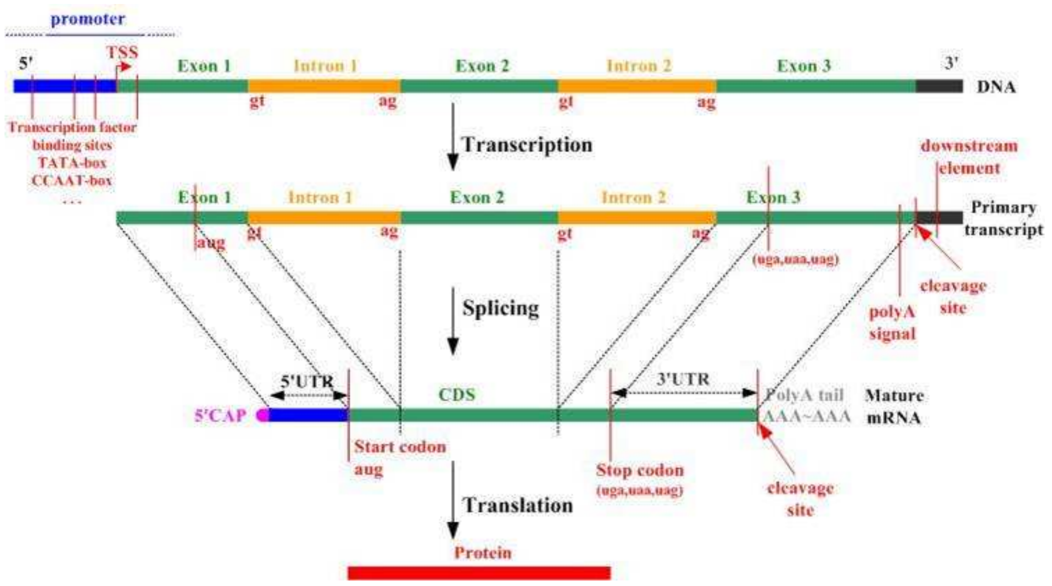


Figure 7.1: Process of protein synthesis.

codon.

2. **Translation**. In this stage, the information that has been passed to RNA from DNA is used to make proteins. At the *initiation* phase of translation, ribosome binds to the mRNA when it reaches an AUG (adenine, uracil, guanine) sequence on the RNA strand in a suitable context. The ribosome is made of protein and ribosomal RNA (rRNA). The start codon AUG is called translation initiation site (TIS) and is only recognized by the initiator tRNA (transfer RNA). After binding to the mRNA, the ribosome proceeds to the *elongation* phase of protein synthesis by sequentially binding to the appropriate codon in mRNA to form base pairs with the anticodon of another tRNA molecule. Hence, with the ribosome moving from codon to codon along the mRNA, amino acids are added one by one, translated into polypeptide sequences. At the end, the newly formed strand of amino acids (complete polypeptide) is released from the ribosome when a release factor binds to the stop codon. This is the *termination* phase of translation.

The functional sites in DNA sequences include transcription start site (TSS), translation initiation site (TIS), coding region, splice site, polyadenylation (cleavage) site and so on that are associated with the primary structure of genes. Recognition of these biological functional sites in a genomic sequence is an important bioinformatics application [72].

In order to apply traditional machine learning techniques to above functional sites recognition problem, we propose a 3-step work flow as follows. In the first step, candidate features are generated using k-gram nucleotide acid or amino acid patterns and then sequence data are transformed with respect to the newly generated feature space. In the second step, a small number of good features are selected by a certain algorithm. In the third step, a classification model is built to recognize the functional site.

7.1 Method Description

The first and the most important step of our method is to generate a new feature space under which the original sequences can be transformed to the format to which general machine learning tools can be easily applied.

7.1.1 Feature generation

We generate the new feature space using k-gram ($k = 1, 2, 3, ...$) *nucleotide* or *amino acid patterns*. A k-gram is simply a pattern of k consecutive letters, which can be amino acid symbols or nucleic symbols [143, 72]. We use each k-gram nucleotide or amino acid pattern as a new feature. For example, nucleotide acid pattern "TCG" is a 3-gram pattern while amino acid pattern "AR" is a 2-gram pattern constituted by an alanine followed by an arginine. Our aim is to recognize functional site in a sequence by analysing k-gram patterns around it. Generally, up-stream and down-stream k-gram patterns of a candidate functional site (for example, every ATG is a candidate of translation initiation site) are treated as different features. Therefore, if we use nucleotide patterns, for each k, there are 2×4^k possible combinations of k-gram patterns; if we use amino acid patterns, since there are 20 standard amino acids plus 1 stop codon symbol, there are 2×21^k possible k-gram patterns for each k. If the position of each k-gram pattern in the sequence fragment is also considered, then the number of features will increase dramatically. We call these features as position-specific k-gram patterns. Besides, k-gram can also be restricted those *in-frame* ones.

The *frequency* of a k-gram pattern is used as the value of this feature. For example,

1. UP-X (DOWN-X), which counts the number of times the letter X appears in the up-stream

(down-stream) part of a functional site in its nucleotide acid or amino acid sequence.

2. UP-XY (DOWN-XY), which counts the number of times the two letters XY appear as a substring in the up-stream (down-stream) part of a functional site in its nucleotide acid or amino acid sequence.

where X and Y range over the 4 nucleotide acid letters or the standard 20 amino acid letters and the special stop codon symbol.

In the framework of the new feature space, the initial nucleotide sequences need to be transformed. The transformation is constructed as follows. Given a DNA nucleotide sequence, a sequence window is set aside for each candidate functional site with it in the center and certain bases up-stream (named as *up-stream window size*) and certain bases down-stream (named as *down-stream window size*). If a candidate functional site does not have enough up-stream or down-stream context, we pad the missing context with the appropriate number of dont-care ("?") symbols.

If features are made from amino acid patterns, we will code every triplet nucleotides, at both up-stream and down-stream of the centered candidate functional site in a sequence window, into an amino acid using the standard codon table. A triplet that corresponds to a stop codon is translated into a special "stop" symbol. Thus, every nucleotide sequence window is coded into another sequence consisting of amino acid symbols and "stop" symbol. Then the nucleotide or amino acid sequences are converted into frequency sequence data under the description of our new features. Later, the classification model will be applied to the frequency sequence data, rather than the original cDNA sequence data or the intermediate amino acid sequence data.

7.1.2 Feature selection and integration

In most cases, the number of candidate features in the feature space is relatively big. It is reasonable to expect that some of the generated features would be irrelevant to our prediction problem while others are indeed good signals to identify the functional site. Thus, in the second step, feature selection is applied to the feature space to find those signals most likely to help in distinguishing the true functional site from a large number of candidates. Besides, feature selection also greatly speeds up the classification and prediction process, especially when the number of samples is large. Among the many feature selection techniques presented in Chapter 3, we employ the simple *entropy measure* (Section 3.2.4) in our following two applications. As used in gene expression data analysis (with name “all-entropy”), we choose all the features whose value range can be partitioned into intervals by Fayyad’s discretization algorithm [36] (Section 3.2.4 of Chapter 3).

To achieve the ultimate goal of predicting the true functional site, our next step is to integrate the selected features by a classification algorithm. At this step, in the following two applications, we will focus on the results achieved by support vector machines (SVM) (with linear or quadratic polynomial kernel function) and our ensemble method CS4. Detailed techniques of SVM and CS4 can be found in Section 2.3.2 and Section 2.3.4 of Chapter 2, respectively.

In the following two sections, we will make use of our proposed work flow to predict translation initiation site and polyadenylation signals.

7.2 Translation Initiation Site Prediction

7.2.1 Background

The translation initiation site (TIS) prediction problem is about how to correctly identify TIS in mRNA, cDNA, or other types of genomic sequences. At the translation stage of protein synthesis process, in eukaryotic mRNA, the context of the start codon (normally “AUG”) and the sequences around it are crucial for recruitment of the small ribosome subunit. Thus, the characterization of the features around TIS will be helpful in a better understanding of translation regulation and accurate gene predication of coding region in genomic and mRNA/cDNA sequences. This is an important step in genomic analysis to determine protein coding from nucleotide sequences.

Since 1987, the recognition of TIS has been extensively studied using biological approaches, data mining techniques, and statistical models [56, 57, 58, 89, 59, 103, 83, 145, 90, 48, 142]. Pedersen and Nielsen [89] directly fed DNA sequences into an artificial neural network (ANN) for training the system to recognize true TIS. They achieved a result of 78% sensitivity on start ATGs (i.e. true TISs) and 87% specificity on non-start ATGs (i.e. false TISs) on a vertebrate data set, giving an overall accuracy of 85%. Zien *et al* [145] studied the same vertebrate data set, but replaced ANN with support vector machines (SVM) using different kinds of kernel functions. They believe that carefully designed kernel functions are useful for achieving higher TIS prediction accuracy. One of their kernel functions is called "locality-improved" kernel, which emphasizes correlations between any two sequence positions that are close together, and a span of 3 nucleotides up- and down-stream is empirically determined as optimal. Recently, Hatzigeorgiou [48] built a multi-step ANN system named "DIANA-TIS" to study the recognition problem. This ANN system combines a consensus ANN and a coding ANN with the ribosome scanning model. They obtained an overall accuracy of 94% on a data set containing full-length human cDNA sequences. All of these methods use nucleotide sequence data directly; they do not generate any new and explicit features for the differentiation between true and false TISs.

There are some related works that use statistical features. The program *ATGpr* [103] uses a linear discriminant function that combines some statistical features derived from the sequence. Each of those features is proposed to distinguish true TIS from false TIS. In a more recent work [83], an improved version of *ATGpr* called *ATGpr_sim* was developed, which uses both statistical information and similarities with other known proteins to obtain higher accuracy of fullness prediction for fragment sequences of cDNA clones. In our previous study [72], the same vertebrate data set was analyzed by generating features using nucleotide acid patterns.

7.2.2 Data

We collected three data sets for this study.

The first data set (data set I) is provided by Dr. Pedersen. It consists of vertebrate sequences extracted from GenBank (release 95). The sequences are further processed by removing possible introns and joining the remaining exon parts to obtain the corresponding mRNA sequences [89]. From these sequences, only those with an annotated TIS, and with at least 10 up-stream nucleotides as well as 150 down-stream nucleotides are considered in our studies. The sequences are then filtered to remove homologous genes from different organisms, sequences added multiple times to the database, and those belonging to same gene families. Since the data are processed DNA, the TIS site is ATG — that is, a place in the sequence where "A", "T", and "G" occur in consecutive positions in that order. We are aware that some TIS sites may be non-ATG; however, this is reported to be rare in eukaryotes [59] and is not considered in this study.

An example entry from this data set is given in Figure 7.2. There are 4 ATGs in this

```
299 HSU27655.1 CAT U27655 Homo sapiens
CGTGTGTGCAGCAGCCTGCAGCTGCCCCAAGCCATGGCTGAACACTGACTCCCAGCTGTG 80
CCCAGGGCTTCAAAGACTTCTCAGCTTCGAGCATGGCTTTTGGCTGTCAGGGCAGCTGTA 160
GGAGGCAGATGAGAAGAGGGAGATGGCCTTGGAGGAAGGGAAGGGGCCTGGTGCCGAGGA 240
CCTCTCCTGGCCAGGAGCTTCCTCCAGGACAAGACCTTCCACCCAACAAGGACTCCCCT
............................................................  80
................................iEEEEEEEEEEEEEEEEEEEEEEEEEEE 160
EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE 240
EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE
```

Figure 7.2: An example annotated sequence from data set I. The 4 occurrences of ATG are underlined. The second ATG is the TIS. The other 3 ATGs are non-TIS. The 99 nucleotides up-stream of the TIS are marked by an overline. The 99 nucleotides down-stream of the TIS are marked by a double overline. The ".", "i", and "E" are annotations indicating whether the corresponding nucleotide is up-stream (.), TIS (i), or down-stream (E).

example. The second ATG is the TIS. The other 3 ATGs are non-TIS (false TIS). ATGs to the left of the TIS are termed *up-stream ATGs*. So the first ATG in the figure is an up-stream ATG. ATGs to the right of the TIS are termed *down-stream ATGs*. So the third and fourth ATGs in the figure are down-stream ATGs. The entire data set contains 3312 sequences. In these sequences, there are a total number of 13375 ATGs, of which 3312 ATGs (24.76%) are true TISs, while 10063 (75.24%) are false. Of the false TISs, 2077 (15.5%) are up-stream ATGs.

The second data set (data set II) is provided by Dr. Hatzigeorgiou. The data collection was first made on the protein database Swissprot. All the human proteins whose N-terminal sites are sequenced at the amino acid level were collected and manually checked [48]. Then the full-length mRNAs for these proteins, whose TIS had been indirectly experimentally verified, were retrieved. The data set consists of 480 human cDNA sequences in standard FASTA format. In these sequences, there are as many as 13581 false TIS, 96.59% of total number of ATGs. However, only 241 (1.8%) of them are up-stream ATGs.

Besides these two data sets that have been analyzed by others, we also formed our own genomic data set (data set III) by extracting a number of well-characterized and annotated human genes of Chromosome X and Chromosome 21 from Human Genome Build30 [70]. Note that we eliminated those genes that were generated by other prediction tools. The resulting set consists of 565 sequences from Chromosome X and 180 sequences from Chromosome 21. These 745 sequences containing true TIS are used as positive data in our experiment. Meanwhile, in order to get negative data, we extracted a set of sequences around all ATGs in these two chromosomes but excluded annotated ones.

7.2.3 Feature generation and sequence transformation

As every 3 nucleotides code for an amino acid, in this study, we use k-gram ($k = 1, 2$) amino acid patterns as candidate features. Thus, there are 924 ($= (21 + 21^2) \times 2$) possible amino acid patterns, i.e. new features.

In the sequence transformation, we set both up-stream window size and down-stream window size to 99 bases — given a cDNA or mRNA nucleotide sequence containing ATGs, a window is set for each ATG with the ATG in the center and 99 bases up-stream and 99 bases down-stream (excluding the ATG itself) aside. As such, for data set I, we get 3312 sequence windows containing true TIS and 10063 containing false TIS; for data set II, 480 sequence windows containing true TIS and 13581 containing false TIS. All the windows have same size, i.e. containing 201 nucleotides. For ease of discussion, given a sequence window, we refer to each position in the sequence window relative to the target ATG of that window. The "A" in the target ATG is numbered as +1 and consecutive down-stream positions — that is, to the right — from the target ATG are numbered from +4 onwards. The first up-stream position — that is, to the left — adjacent to the target ATG is –1 and decreases for consecutive positions towards the 5' end — that is, the left end of the sequence window [72]. These sequence windows containing nucleotide letters are further transformed to amino acid sequences by coding every triplet nucleotides into an amino acid or a stop codon. At last, the amino acid sequences are converted into frequency sequence data under the description of feature space.

Apart from the k-gram amino acid patterns, we also derive three new features from some known bio-knowledge: two are based on the famous Kozak's consensus matrix and one is on the scanning model. From the original work for the identification of the TIS in cDNA sequences, Kozak developed the first weight matrix from an extended collection of data [56]. The consensus motif from this matrix is GCC[**AG**]CC$\underline{\text{ATG}}$**G**, where (1) a G residue tends to follow a true TIS, which indicates that a "G" appears in position +4 of the original sequence window; (2) a purine (A or G) tends to be found 3 nucleotides up-stream of a true TIS, which indicates that an "A" or a "G" appears in position -3 of the original sequence window. Also, according to the ribosome scanning model [27, 57, 4], an mRNA sequence is scanned from left (5') to right (3'), and the scanning stops as soon as an ATG is recognized as TIS. The rest of the ATGs in the mRNA sequence to the right of this ATG are then treated as non-TIS. To incorporate these knowledge to our feature space, we add three Boolean features “DOWN4-G”,“UP3-AorG” and “UP-ATG”. Here, UP-ATG means whether an in-frame up-stream ATG exists. In a nucleotide sequence window extracted for each candidate TIS, we call those 3-grams in positions $\cdots$, -9, -6, and -3, the in-frame up-stream 3-gram patterns; and those 3-grams in positions +4, +7, +10, $\cdots$, the in-frame down-stream 3-gram patterns. Finally, there are 927 features in the new feature space.

After this process of feature generation and data transformation, we get 3312 true TIS samples and 10063 false TIS samples from data set I, 480 true TIS samples and 13581 false TIS samples from data set II. Each sample is a vector of 924 integers and three boolean values. Figure 7.3 presents a diagram for the data transformation with respect to our new feature space.

7.2.4 Experiments and results

To verify the effectiveness of our method from different aspects, we designed a series of experiments on the three data sets:

a. Conducting computational cross validations in data set I and data set II separately.

b. Selecting features and building classification model using data set I. Applying the well-trained model to data set II to obtain a blind testing accuracy.

c. Incorporating the idea of ribosome scanning into the classification model.

d. Applying the model built in experiment-b to genomic sequences.

Validation in different data sets

To strictly compare with the results presented in [142, 72], we conduct the same 3-fold cross validation. Table 7.1 shows our results on the data set I and data set II using the features selected by the entropy-based algorithm. With the simple linear kernel function, SVM achieves accuracy of 92.04% at 81.13% sensitivity and 95.63% specificity on data set I. This is better than the accuracy of 89.4% at 74.0% sensitivity and 94.4% specificity, which is the previous best result reported on the same data set [142]. On data set II, SVM achieves an accuracy of 98.42% at

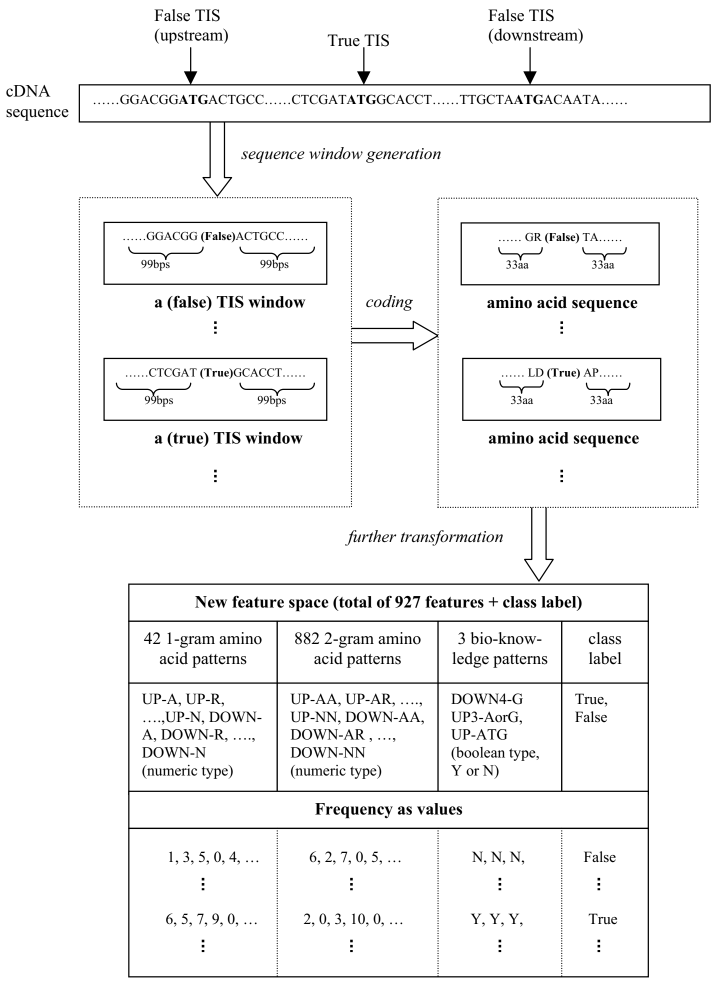


Figure 7.3: A diagram for data transformation aiming for the description of the new feature space.

63.33% sensitivity and 99.66% specificity. Note that we can not find previously reported results on this data set under similar cross validation.

Validation across two data sets

The good cross validation results achieved within the individual data set encourage us to extend our study to span the two data sets. In this experiment, we use the whole data set I as training data to select features and build the classification model, then we evaluate the well-trained model on data set II to get a test accuracy.

To reduce the *similarity* between the training and testing data, a *BLAST* search between the

Table 7.1: The results by 3-fold cross validation on the two data sets (experiment-a). SVM(linear/quad) means the classification model is built by linear/quadratic polynomial kernel function.

Data	Algorithm	Sensitivity	Specificity	Precision	Accuracy
I	SVM(linear)	81.13%	95.63%	85.93%	92.04%
	SVM(quad)	80.19%	96.17%	87.34%	92.22%
	CS4	76.18%	96.14%	86.67%	91.20 %
II	SVM(linear)	63.33%	99.66%	86.86%	98.42%
	SVM(quad)	71.25%	99.42%	81.24%	98.46%
	CS4	83.54%	97.67%	55.93%	97.19%

Table 7.2: Classification accuracy when using data set I as training and data set II as testing (experiment-b). The row of II** is the testing accuracy on data set II before similar sequences being removed.

Data	Algorithm	Sensitivity	Specificity	Precision	Accuracy
I (train)	SVM(linear)	80.68%	96.75%	89.10%	92.77%
	SVM(quad)	86.05%	98.14%	93.84%	95.15%
	CS4	85.54%	97.91%	93.10%	94.85%
II (test)	SVM(linear)	96.28%	89.15%	25.31%	89.42%
	SVM(quad)	94.14%	90.13%	26.70%	90.28%
	CS4	92.02%	92.71%	32.52%	92.68%
II** (test)	SVM(linear)	95.21%	89.74%	24.69%	89.92%
	SVM(quad)	94.38%	89.51%	24.12%	89.67%
	CS4	87.70%	93.26%	28.60%	92.11%

data set I and II is performed. Two sequences are considered similar if they produce a BLAST hit with an identity $> 75\%$. We find 292 similar sequences and removed them from data set II. As a result, after being removed similar sequences, data set II contains 188 real TIS, while there are total number of 5111 candidates [70].

We train SVM model on data set I and obtain training accuracy 92.77% at 80.68% sensitivity and 96.75% specificity. Using this model, we get a test accuracy of 89.42% at 96.28% sensitivity and 89.15% specificity on data set II. We note that the testing accuracy on the original data set II (without the removal of the similar sequences) is quite similar. See Table 7.2 for a summary of these results.

Remarkably, this cross-validation spanning the two data sets achieves a much better sensitivity on data set II than that obtained in the 3-fold cross-validation on this data set. A reason may be that only 3.41% of candidate ATGs in data set II are true TISs, which leads to an extremely unbalanced numbers of samples between the two classes. However, this bias is rectified significantly by the model built on data set I where the population size of true TIS versus false TIS is more balanced.

Incorporation of scanning model

Hatzigeorgiou [48] reported a high accuracy on data set II by an integrated method which combines a consensus ANN with a coding ANN together with a ribosome scanning model. The model suggests to scan from the 5' end of a cDNA sequence and predicts TIS at the first ATG in a good context [27, 57, 4]. The rest of the ATGs in the cDNA sequence to the right of this ATG are then automatically classified as non-TIS. Thus, one and only one ATG is predicted as TIS per cDNA sequence.

We also incorporate this scanning model into our experiment. This time, in a sequence, we test ATGs in turn from left to right, until one of them is classified as TIS. A prediction on a sequence is correct if and only if the TIS itself is predicted as a TIS. Since the scanning model indicates that the first ATG that in an optimal nucleotide context would be TIS, a higher prediction accuracy is expected if only up-stream ATGs and true TIS are used in training. Thus, we ignore all down-stream ATGs in data set I and obtain a new training set containing only true TISs and their up-stream ATGs. Then feature selection and classification model learning are based on this new training data. Table 7.3 shows our results with scanning model being used.

Under this scanning model idea, Artemis reported that 94% of the TIS were correctly predicted on data set II [48]. As mentioned in her paper [48], the data set was split into training and testing parts in some way, the results reported there are not directly comparable with our results.

Testing on genomic sequences

In order to further evaluate the feasibility and robustness of our method, we apply our model built in experiment-b to our own prepared data (data set III), which contain gene sequences of Chromosome X and Chromosome 21. Using the simple linear kernel function, SVM gives 397 correct prediction out of a total of 565 true TISs found in Chromosome X while 132 correct prediction out of a total of 180 true TISs in Chromosome 21. The sensitivities are 70.27% and 73.33%, respectively. To obtain the specificity of our models, we randomly select the same number of sequences containing non-start ATGs (false TIS) from our own extracted negative

Table 7.3: Classification accuracy under scanning model when using data set I (3312 sequences) as training and data set II (188 sequences) as testing (experiment-c). The row of II** is the testing accuracy on data set II before similar sequences being removed (480 sequences). NoCorrectlyPredicted is the number of sequences whose TIS is correctly predicted.

Data	Algorithm	NoCorrectlyPredicted	Accuracy
I	SVM(linear)	3161	95.44%
(train)	SVM(quad)	3156	95.29%
	CS4	3083	93.09%
II	SVM(linear)	174	92.55%
(test)	SVM(quad)	172	91.49%
	CS4	176	93.62%
II**	SVM(linear)	453	94.38%
(test)	SVM(quad)	450	93.75%
	CS4	452	94.17%

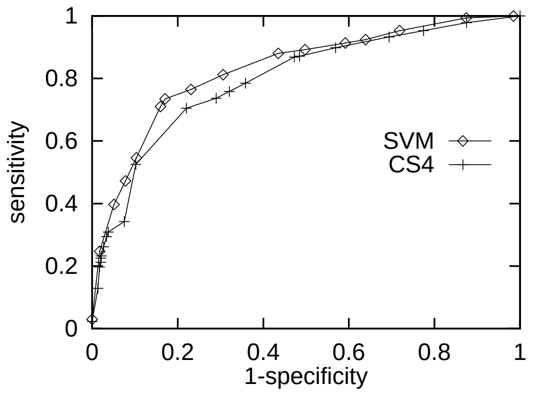


Figure 7.4: ROC curve of SVM and CS4 on prediction TIS in genomic data Chromosome X and Chromosome 21 (experiment-d). The SVM model is built on the linear kernel function. The area under the ROC curve: SVM 0.837, CS4 0.772.

data set. SVM correctly predicts 626 of these 745 non-start ATGs, obtaining a specificity at 84.02%. In the same test, CS4 achieves 52.48% sensitivity and 89.80% specificity. One point needs to be addressed here is that in this validation, we remove the feature built on the ribosome scanning model since that model is not true for genomic data. To illustrate the tradeoff between the prediction sensitivity and specificity, Figure 7.4 gives the ROC curves of SVM and CS4 showing the changes of prediction accuracy on true and false TISs.

7.2.5 Discussion

Significant Features

“What are the key features to predict TIS?” To answer this question, let us have a look of an interesting discovery on the features selected in the 3-fold cross validation on data set I in our experiment-a. Table 7.4 shows the ranking positions of the 10 top-ranked features based on their entropy value for the each fold. Observe that they are the same features though their ordering is slightly different from one fold to another. This suggests that these features, or exactly amino acid patterns, are indeed patterns around true or false TISs. Furthermore, “UP-ATG” can be explained by the ribosome scanning model [27, 4] — seeing such an up-stream ATG makes the candidate ATG less likely to be the TIS. “DOWN-STOP” is the in-frame stop codons down-stream from the target ATG and it is consistent with the biological process of translating in-frame codons into amino acids stops upon encountering an in-frame stop codon — seeing such a down-stream stop codon makes the candidate protein improbably short. “UP3-AorG” is correspondence to the well-known Kozak consensus sequence [56]. Most of the other features were also identified in our previous study [142], in which the feature space is built directly on nucleotides. Remarkably, these amino acid patterns, except “DOWN-L”, all contain “G” residue. Note also that “UP-M” is one of the top features in each fold, but we exclude it as it is redundant given that UP-ATG is true if and only if UP-M > 0. The significance of these features is further verified when we find that both sensitivity and specificity drop down greatly if these features are all excluded from the classification model. However, we do not observe obvious decrease when we remove any one of them from the model. This may suggest that in real biological process of translation there are some factors other than Kozak consensus that may regulate the recognition of TIS.

In addition to the result when only selected features are used, we also obtain cross-validation results on the whole feature space (i.e. without feature selection). We find that using the whole feature space can not let us achieve better results on all of our experiments. For example, SVM with linear kernel function achieves accuracy 90.94% at 79.86% sensitivity and 94.58% specificity for data set I when running 3-fold cross validation on data set I. This result is not as good as that on the selected features.

Table 7.4: Ranking of the top 10 features based on their entropy value as relevant in each of the 3 folds of data set I. Feature "UP-ATG" indicates whether an in-frame up-stream ATG exists (boolean type). Feature "UP3-AorG" tests whether purine A or G tends to be found 3 nucleotides up-stream of a true TIS (boolean type). Feature "UP(DOWN)-X" counts the occurrence that an in-frame (relative to the candidate ATG) triplet coding for the amino acid letter X appears in the up-stream (down-stream) part of a candidate ATG. Feature "DOWN-STOP" is the occurrence of in-frame stop codons down-stream of a candidate ATG.

Fold	UP-ATG	DOWN-STOP	UP3-AorG	DOWN-A	DOWN-V	UP-A	DOWN-L	DOWN-D	DOWN-E	UP-G
1	1	2	4	3	6	5	8	9	7	10
2	1	2	3	4	5	6	7	8	9	10
3	1	2	3	4	5	6	8	9	7	10

Classification algorithms

For the classification methods, overall speaking, SVM performs slightly better than our CS4 method, in terms of prediction accuracy. However, CS4 achieves very good sensitivity when running 3-fold cross validation on data set II where the number of true TISs is much less than the number of false TISs. On the prediction of TIS in genomic sequences, the performance of CS4 is close to that of SVM. This can be illustrated by the ROC curves drawn in the Figure 7.4 — the areas under the curves are SVM 0.837 and CS4 0.772, respectively. Besides, decision trees can output comprehensive rules to disclose the essence of learning and prediction. Some discovered interesting and biologically sensible rules with large coverage are listed below.

1. If *UP-ATG='Y'* and *DOWN-STOP>0*, then prediction is *false TIS*.

2. If *UP3-AorG='N'* and *DOWN-STOP>0*, then prediction is *false TIS*.

3. If *UP-ATG='N'* and *DOWN-STOP=0* and *UP3-AorG='Y'*, then prediction is *true TIS*.

On the other hand, in our series of experiments, SVM built on quadratic polynomial kernels do not show much advantage over those built on simple linear kernel functions. Note that quadratic kernels need much more time on training process.

Comparison with model built on nucleotide acid patterns

In [142], data set I was studied using k-gram nucleotide acid patterns and several classification methods including SVMs, Naive Bayes, Neural Network and decision tree. In that study, feature selection was also conducted, but by CFS (Correlation-based Feature Selection) which is introduced in Section 3.2.6 of Chapter 3. The best accuracy achieved on the 3-fold cross validation was 89.4% at 74.0% sensitivity and 94.4% specificity when some 3-gram nucleotide acid patterns were used. This result is not as good as that presented in this section — 92.45% accuracy at 80.19% sensitivity and 96.48% specificity. However, the good features selected by these two experiments are highly consistent. Besides those 3 features built on bio-knowledge, CFS picked out down-stream TAA (stop codon), TAG (stop codon), TGA (stop codon), CTG (amino acid L), GAC (D), GAG (E) and GCC (A). If we code these 3-gram nucleotide patterns into 1-gram amino acid patterns, we will find they are all among the best features listed in Table 7.4. On the other hand, although there are no 2-gram amino acid patterns among the 10 best features in Table 7.4, some of them are indeed included in the set of selected features that has been used to achieve better results in this study. Note that, our previous study [142] also illustrated that using 4-gram, 5-gram nucleotide acide patterns could not help improve the prediction performance.

Comparison with *ATGpr*

As mentioned earlier, *ATGpr* [103, 83] is a TIS prediction program that makes use of a linear discriminant function, several statistical measures derived from the sequence and the ribosome scanning model. It can be accessed via `http://www.hri.co.jp/atgpr/`. When searching TIS in a given sequence, the system will output several (5 by default) ATGs in the order of decreasing confidence. Let us take the ATG with highest confidence as TIS. Then for the 3312 sequences in our data set I, *ATGpr* can predict correctly true TIS in 2941 (88.80%) of them. This accuracy is 6.64% lower than that we achieved. For our data set II, true TIS in 442 (92.0%) of 480 sequences are properly recognized, which is about 2.38% lower than the accuracy obtained by us. Our results quoted here are based on SVM model using the linear kernel function.

When we feed the genomic data used in our experiment-d to *ATGpr*, the program gives correct TIS predictions on 128 (71.11%) of 180 Chromosome 21 gene sequences and 417 (73.81%) of 565 Chromosome X gene sequences, giving the overall sensitivity as 73.15%. On the other hand, *ATGpr* achieves 70.47% prediction accuracy on the same number of negative sequences that were used in our experiment-d. From the ROC curves shown in Figure 7.4, we can find our prediction specificities are around 80% (SVM) and 73% (CS4) when sensitivity is 73.15% —

9.5% and 2.5% higher than that of *ATGpr* on specificity. This indicates that our program may also outperform *ATGpr* when dealing with genomic data sequences.

7.3 Polyadenylation Signal Prediction

7.3.1 Background

The general polyadenylation machinery of mammalian cells has been well studied for decades. The polyadenylation (poly(A)) reaction of mammalian pre-mRNAs proceeds in two phases: the cleavage of pre-mRNA and the addition of poly(A) tail to the newly formed 3' end. The cleavage reaction requires the cleavage/poly(A) specificity factor (CPSF), the cleavage stimulation factor (CStF), the cleavage factors I and II (CF I and CF II), and poly(A) polymerase (PAP) in most cases. CPSF, PAP and poly(A) binding protein 2 are involved in poly(A) [144]. The assembly of the cleavage/poly(A) complex, which contains most or all of the processing factors and the substrate RNA, occurs cooperatively. CPSF consists of four subunits and binds to the highly conserved AAUAAA hexamer up-stream of the cleavage site. CStF, which is necessary for cleavage but not for the addition of poly(A) tail, interacts with the U/GU rich element located down-stream of the AAUAAA hexamer. Two additional factors, the cleavage factor I and II (CF I and CF II) act only in the cleavage step. CF I has been purified to homogeneity and shown to be an RNA-binding factor. CF II has been only partially purified so far, and its function is not known.

After the formation of the cleavage/polyadenylation complex, the selection of poly(A) site is primarily determined by the distance between a hexameric poly(A) signal (PAS) of sequence AAUAAA (or a one-base variant) and the down-stream element(denoted as DSE). The spacing requirements for the PAS and DSE reflect the spatial requirements for a stable interaction between CPSF and CStF. The DSE is poorly conserved and two main types have been described as a U-rich, or GU-rich element, which locates 20 to 40 bases down-stream of the cleavage site (for reviews, please refer to [28, 144, 141]). DSE is present in a large proportion of genes and can affect the efficiency of cleavage [75, 141]. Although in a few cases, an up-stream element (denoted as USE) is required for the poly(A) signal to be fully activated [5, 18, 79], the position and sequence of the USE are undefined. In summary, the organization of mammalian poly(A)

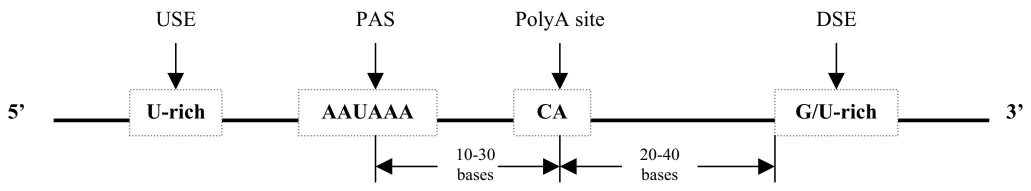


Figure 7.5: Schematic representation of PAS in human mRNA 3'end processing site. Distances are as described in [28].

sites may have an unexpected flexibility and their activity depends on not only the hexameric signal but also the up/down elements. Figure 7.5 is a schematic representation of PAS in human mRNA 3'end processing site [144].

There are several software programs that have been developed to detect PASes in human DNA and mRNA sequences by analysing the characteristics of up-stream and down-stream sequence elements around PASes. In one of early studies, Tabaska and Zhang [119] developed a program named *Polyadq*, which finds PASes using a pair of quadratic discriminant functions. Besides, they also created a database of known active poly(A) sites and trained their program on 280 mRNA sequences and 136 DNA sequences. In their tests of finding PASes, they claimed a correlation coefficient of 0.413 on whole genes and 0.512 in the last two exons of genes. *Polyadq* is available at `http://argon.cshl.org/tabaska/polyadq_form.html`. Recently, Legendre and Gautheret [61] used bioinformatics analysis of EST and genomic sequences to characterize biases in the regions encompassing 600 nucleotides around the cleavage site. The computer program they developed is called *Erpin* which uses 2-gram position-specific nucleotide acid patterns to analyse 300 bases up-stream and down-stream region of a candidate PAS. Being trained by 2327 terminal sequences, *Erpin* was reported to achieve a prediction specificity of 75.5% to 90.4% for a sensitivity of 56% on several sets of validation data. The program can be found at `http://tagc.univ-mrs.fr/pub/erpin/`.

In this study, we will apply our method to characterize the features in the regions encompassing 200 nucleotides around the PAS, i.e. with PAS in the centre and both up-stream window size and down-stream window size as 100 bases.

7.3.2 Data

In a series of experiments, a large number of sequences are used to train and test our classification model. They are from two sources.

(1) Training and testing sequences used by program *Erpin* [61]. The training set contains 2327 terminal sequences including 1632 “unique” and 695 “strong” poly(A) sites. The testing set consists of 982 positive sequences containing annotated PASes from EMBL and four sets of same sized negative sequences: 982 CDS sequences, 982 intronic sequences of the first intron, 982 randomized UTR sequences of same 1^{st} order Markov model as human 3’ UTRs, and 982 randomized UTR sequences of same mono nucleotide composition as human 3’ UTRs. The 2327 training sequences can be downloaded from `http://tagc.univ-mrs.fr/pub/erpin/` and have been trimmed in accordance to our window segments i.e. every sequence contains 206 bases, having a PAS in the center. We obtained testing data sets from Dr Gautheret via personal communication.

(2) Human RefSeq mRNA data set: we obtained 312 human mRNA sequences from RefSeq [94] release 1. Each of these sequences contains a “polyA-signal” feature tag carrying an “evidence=experimental” label. We use these sequences to build model for PAS prediction in mRNA sequences. Besides, we also extracted a set of human mRNA sequences from RefSeq containing a “polyA-site” feature tag carrying an “evidence=experimental” label. In this set, we removed the sequences that have been included in the training set used in building our model. We use these sequences for testing purpose assuming that the annotated PAS positions are correct. Our negative data set was generated by scanning for the occurrences of AATAAA at coding region and those AATAAA sites near the end of sequence were excluded purposely.

7.3.3 Experiments and Results

First, we use simple 1-gram, 2-gram and 3-gram nucleotide acid patterns to construct feature space [69]. Thus, there are 168 ($= (4 + 4^2 + 4^3) \times 2$) features.

Table 7.5: Validation results by different programs on a set of 982 annotated UTR sequences from the EMBL database [61]. TP is the number of true positives. FN is the number of false negatives.

Program	TP	FN	Sensitivity
Erpin	549	433	55.9%
Polyadq	547	435	55.7%
Ours	553	429	56.3%

Preliminary results

In the first experiment, we use the 2327 sequences introduced in [61] (data source (1)) as our true PAS training data. To obtain negative sequences, same sized false PAS data is randomly selected from our own extracted negative data set (data source (2)). Using entropy-based feature selection algorithm and linear kernel SVM classifier, the sensitivity and specificity of 10-fold cross-validation on training data are 89.3% and 80.5%, respectively. In order to compare with other programs, we test our model on the same validation sets whose testing results on programs *Erpin* and *Polyadq* were reported in [61]. As described in data source (1) , these validation sets include true PASes sequences came from 982 annotated UTRs and four same sized control sets known not to contain PASes: coding sequences (CDS), introns and randomized UTRs (simply shuffled UTRs and 1^{st} order Markov model UTRs). For a direct comparison, we also adjust the prediction sensitivity on the 982 true PASes to around 56.0% so that evaluation can be made on the predictions for those four control sets.

Table 7.5 shows the validation results on true PASes and Table 7.6 illustrates the results on four control sets. Figure 7.6 is the ROC curve for this series of tests. All the numbers regarding to the performance of programs *Erpin* and *Polyadq* in Table 7.5 and Table 7.6 are copied or derived from [61]. The results in Table 7.6 demonstrate that our model can give better performance than *Erpin* and *Polyadq* did on false PASes prediction of CDS, intron and simple shuffling sequences, and almost same prediction accuracy on sequences with 1^{st} order Markov randomization.

In this experiment, we select 113 features via entropy measure. These features are then integrated with SVM to form the classification and prediction model. Table 7.7 lists the top 10 of these features ranking by their entropy values (the less the entropy value is, the more important the feature is). Some of these top features can be interpreted by those reported motifs, for example, it clearly visualizes both USE and DSE as characterized by G/U rich segments since

Table 7.6: Validation results by different programs on different sequences not containing PASes: coding sequences (CDS), introns, and two types of randomized UTR sequences (simple shuffling and 1^{st} order Markov simulation) [61]. TN is the number of true negatives. FP is the number of false positives. CC is correlation coefficient, and $CC = \frac{(TP*TN-FP*FN)}{\sqrt{(TP+FP)*(TP+FN)*(TN+FP)*(TN+FN)}}$. Calculations of Precision and CC use TP and FN from Table 7.5.

Data set	Program	TN	FP	Specificity	Precision	CC
CDS	Erpin	880	102	89.6%	84.3%	0.483
	Polyadq	862	120	87.8%	82.0%	0.459
	Ours	887	95	90.3%	85.4%	0.497
Introns	Erpin	741	241	75.5%	69.5%	0.320
	Polyadq	718	264	73.1%	67.5%	0.293
	Ours	775	207	78.9%	72.8%	0.363
Simple shuffling	Erpin	888	94	90.4%	85.4%	0.494
	Polyadq	826	156	84.1%	77.8%	0.415
	Ours	942	40	95.9%	93.3%	0.570
Markov 1^{st} order	Erpin	772	210	78.6%	72.3%	0.354
	Polyadq	733	249	74.6%	68.7%	0.309
	Ours	765	217	77.9%	71.9%	0.351

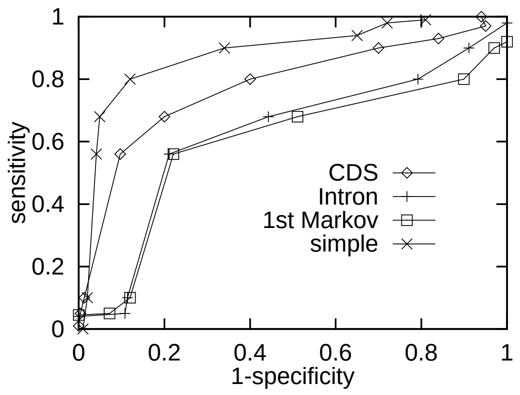


Figure 7.6: ROC curve of our model on some validation sets described in [61] (data source (1)).

UP-TGT, UP-T, DOWN-TGT, DOWN-T, UP-TG and UP-TT are among top features.

Model for prediction PAS in mRNA sequences

When we apply our model to 312 true PASes that were extracted from mRNA sequences by ourselves (data source (2)), the results obtained are not good — only around 20% of them can be predicted correctly. Besides, the program *Erpin* performs even worse on these PASes — with prediction accuracy at only 13%. These poor results may indicate that the good features used in the model for PAS prediction in DNA sequences are not efficient for mRNA. Therefore, we

Table 7.7: The top 10 features selected by entropy-based feature selection method for PAS classification and prediction in human DNA sequences.

Rank	1	2	3	4	5	6	7	8	9	10
Feature	UP -TGT	DOWN -A	UP -T	UP -AG	DOWN -TGT	DOWN -T	UP -TG	UP -TT	DOWN -AA	UP -A

decide to build another model for mRNA sequences without poly(A) tails. This model is also expected to provide a new way for predicting the mRNA cleavage site/poly(A) addition site.

Since the new model is aimed to predict PASes from mRNA sequences, we only consider the up-stream elements around a candidate PAS. Therefore, there are only 84 features (instead of 168 features). To train the model, we use 312 experimentally verified true PASes and same number of false PASes that randomly selected from our prepared negative data set. The validation set comprises 767 annotated PASes and same number of false PASes also from our negative data set but different from those used as training (data source (2)). This time, we achieve reasonably good results. Sensitivity and specificity for 10-fold cross-validation on training data are 79.5% and 81.8%, respectively. Validation result is 79.0% sensitivity at 83.6% specificity. Besides, we observe that the top ranked features are different from those listed in Table 7.7 (detailed features not shown).

Since every 3 nucleotides code for an amino acid when DNA sequences translate to mRNA sequences, it is legitimate to investigate if an alternative approach that generating features based on amino acids can produce more effective PASes prediction for mRNA sequence data. In fact, this idea is also encouraged by the good results we achieved in the TIS prediction described in the previous section.

Similarly as what we did in TIS prediction, we transform the up-stream nucleotides of a sequence window set for each cadidate PAS into an amino acid sequence segment by coding every triplet nucleotides as an amino acid or a stop codon. The feature space is constructed by using 1-gram and 2-gram amino acid patterns. Since only up-stream elements around a candidate PAS are considered, there are 462 ($= 21 + 21^2$) possible amino acid patterns. In addition to these patterns, we also present existing knowledge via an additional feature — denoting number of T residue in up-stream as “UP-T-Number”. Thus, there are 463 candidate features in total.

In the new feature space, we conduct feature selection and train SVM on 312 true PASes and same number of false PASes. The 10-fold cross-validation results on training data are 81.7%

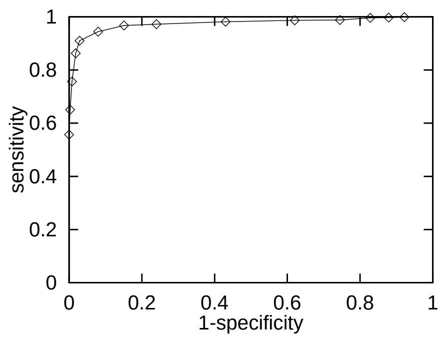


Figure 7.7: ROC curve of our model on PAS prediction in mRNA sequences.

sensitivity with 94.1% specificity. When apply the trained model to our validation set containing 767 true PASes and 767 false PASes, we achieve 94.4% sensitivity with 92.0% specificity (correlation coefficient is as high as 0.865). Figure 7.7 is the ROC curve of this validation. In this experiment, there are only 13 selected features and UP-T-Number is the best feature. This indicates that the up-stream sequence of PAS in mRNA sequence may also contain T-rich segments. However, when we apply this model built for mRNA sequences using amino acid patterns to predict PASes in DNA sequences, we can not get as good results as that achieved in the previous experiment. This indicates that the down-stream elements are indeed important for PAS prediction in DNA sequences.

7.4 Chapter Summary

In this chapter, we proposed a machine learning methodology to identify functional site in biological sequences. Our method comprises three sequential steps: (1) generating candidate features using k-gram nucleotide acid patterns or amino acid patterns and then transforming original sequences respect to the new generated feature space; (2) selecting relevant features using certain feature selection algorithm; and (3) building classification model to recognize the functional site by applying classification techniques to the selected features. Our idea is different from traditional methodologies because it generates new features and also transforms the original nucleotide sequence data to k-gram frequency vectors. The feature selection step does not only greatly shorten the running time of classification program, but also help to obtain explicit important features around the functional site and lead to a more accurate prediction.

We applied our idea to predict translation initiation site (TIS) and polyadenylation signal

(PAS) in DNA and mRNA sequences. For each application, both public data sets and our own extracted sequences were used to test the effectiveness and robustness of the method. The experimental results achieved are better than those reported previously using the same data sets (if available). The important features captured are highly consistent with those reported in the literature. Most importantly, we not only conducted the cross validation within the individual data sets separately, but also established the validation across the different data sets. The success of such a validation indicates that there are predictable patterns around TIS or PAS.

In addition, a web-based toolbox to recognize TIS and PAS from DNA sequences has been implemented based on the techniques presented in this chapter. This toolbox is named as *DNAFSMiner* and more information about it can be found in Appendix B.

Chapter 8

Conclusions

8.1 Summary

This thesis is about how to effectively apply data mining technologies to biological and clinical data. Some problems arising from gene expression profilings and DNA sequence data are studied in depth using data mining techniques of feature generation, feature selection, and feature integration with learning algorithms.

In order to identify genes associated with disease phenotype classification or patient survival prediction from gene expression data, a new feature selection strategy, *ERCOF* (Entropy-based Rank sum test and COrrelation Filtering), is worked out by combining entropy measure, Wilcoxon rank sum test and Pearson correlation coefficient test. ERCOF conducts three-phase feature filtering aiming to find a subset of sharply discriminating genes with little redundancy. In the first phase, it selects genes using an entropy-based method that generally keeps only 10% of the features. In the second phase, a non-parametric statistics called the Wilcoxon rank sum test is applied to the features kept by the first phase to further filter out some genes and divide the remaining ones into two groups — one group consists of genes that are highly expressed in one type of samples (such as *cancer*) while another group consists of genes that are highly expressed in another type of samples (such as *non-cancer*). In the third phase, correlated genes in each group are determined by Pearson correlation coefficient test and only some representatives of them are chosen to form the final set of selected genes.

In Chapter 5, ERCOF is applied to six published gene expression profiling data sets and one proteomic data set to identify genes for phenotype classification. For comparison purpose, several other entropy-based feature selection methods are also run. The classification algorithms used include four ensemble of decision trees approaches, support vector machines (SVM) and k-nearest neighbour (k-NN). The four decision tree methods are the newly implemented CS4 (cascading-and-sharing for decision trees) and state-of-the-art Bagging [19], Boosting, and Random forests. More than one thousand tests are conducted and a variety of comparisons among different feature selection methods and different classification algorithms are addressed. For each data set, some identified discriminating features are also reported and related to the literature and the disease. To demonstrate the advantage of the decision trees over the other classification algorithms, some simple, explicit and comprehensible trees/rules induced from the data sets are also presented and analysed.

In the study of patient survival prediction described in Chapter 6, we present a new idea of selecting informative training samples by defining long-term and short-term survivors. ERCOF is then applied to these samples to identify genes associated with survival status. A regression function built on the selected samples and genes by linear kernel SVM is implemented to assign a risk score to each patient. Kaplan-Meier plots for different risk groups formed on the risk scores are then drawn to show the effectiveness of the model. Two case studies, one on survival prediction for patients after chemotherapy for diffuse large-B-cell lymphoma and one on lung adenocarcinomas, are conducted.

In Chapter 7, data mining methodologies are applied to identify functional sites in DNA sequences. Feature generation is emphasized in this application since sequence data generally contain no explicit features. We first construct feature space using k-gram nucleotide acid or amino acid patterns and then transform original sequences under the new constructed feature space. Feature selection is then conducted to find signal patterns that can distinguish true functional sites from those false ones. In the third step, classification and prediction models are built on the training data sets with the selected features. Our methodology is used to recognize translation initiation sites and polyadenylation signals in DNA and mRNA sequences. For each application, experimental results across different data sets (including both public ones and own extracted ones) are collected to demonstrate the effectiveness and robustness of our method.

8.2 Conclusions

In this thesis, we successfully make use of data mining technologies to solve some problems arising from biological and clinical data. We have articulated explicitly the 3-step frame work of feature generation, feature selection and feature integration with learning algorithms and demonstrated its effectiveness when dealing with phenotype classification and patient survival prediction from gene expression data, and functional sites recognition in DNA sequences.

From large amount of experiments conducted on some high-dimensional gene expression data sets, we clearly observe the improvements on performances of all the classification algorithms under the proposed feature selection scenarios. Among these gene identification methods, we claim ERCOF is an effective approach.

In the aspect of classification algorithms, no single algorithm is absolutely superior to all others, though SVM achieves fairly good results in most of tests. Compared with SVM, decision tree methods can provide simple, comprehensive rules and are not very sensitive to feature selections. Among the decision tree methods, the newly implemented CS4 achieves good prediction performance and provides many interesting rules.

Feature generation is important for some kinds of biological data. We illustrate this point by properly constructing new feature space for functional sites recognition in DNA sequences. Some of the signal patterns identified from the generated feature space are highly consistent with related literature or biological knowledge. The rest might be useful for biologists to conduct further analysis.

8.3 Future Work

There are many ongoing and future explorations regarding to the works presented in this thesis.

Currently, our proposed gene selection method ERCOF is not a non-parametric measure since the expression values are used in the third phase filtering when evaluating the correlations between genes. To avoid this, other metrics and clustering algorithms to measure the relationships of genes are under development.

With more and more high quality gene expression profiles being published, we expect to further test the effectiveness of our proposed frame work and the robustness of various gene selection and classification algorithms on many other data sets. For some particular diseases, we will further extract biological meanings of the genes identified to be most associated with the phenotypes or patient survival status.

Future works in identifying translation initiation sites and poly(A) signals from DNA sequences are planned as follows. (1) We are considering to include patterns containing “dont care” symbols into feature space. Here, a “dont care” symbol (?) stands for any symbol of amino acid or nucleotide acid. Thus, more general signal patterns might be found around functional sites. (2) Some parameters used in constructing feature space and extracting sequences around a candidate functional site will be adjusted so that their impacts on the classification performance will be known. These parameters include the k value of k-gram patterns used as features, the up-stream window size and the down-stream window size of the sequence segment extracted for each candidate, and so on. (3) The classification models built will be tested on more EST (Expressed Sequence Tags) and genomic sequences. (4) Meanwhile, we are expecting the 3-step frame work of feature manipulations will achieve good results on the recognition of other functional sites, such as splice site and etc.

In the aspect of using classification methods to solve biological problems, we will try to provide insight and limitations of different algorithms. In addition to the good performance, how easy it is for users to understand the learning process, to interpret the output classification models, and to incorporate domain knowledge are also important factors in measuring the classification power of an algorithm.

Bibliography

[1] `http://hesweb1.med.virginia.edu/biostat/teaching/shortcourse/km.lam.pdf`.

[2] `http://www.fon.hum.uva.nl/praat/manual/Principal_component_analysis.html`.

[3] `http://www.ncbi.nlm.nih.gov/About/primer/bioinformatics.html`.

[4] P. Agarwal and V. Bafna. The ribosome scanning model for translation initiation: implications for gene prediction and full-length cDNA detection. *Proceedings of 6th International Conference on Intelligent Systems for Molecular Biology*, pages 2–7, June 1998.

[5] Y. Aissouni, C. Perez, B. Calmels, and P.D. Benech. The cleavage/polyadenylation activity triggered by a U-rich motif sequence is differently required depending on the poly(A) site location at either the first or last 3'-terminal exon of the 2'-5' oligo(A) synthetase gene. *Journal of Biological Chemistry*, 277:35808–35814, 2002.

[6] 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.Jr. Hudson, L. Lu, D.B. Lewis, R. Tibshirani, G. Sherlock, W.C. Chan, T.C. Greiner, D.D. Weisenburger, J.O. Armitage, R. Warnke, and L.M. Staudt. Distinct types of diffuse large B-cell lymphoma identified by gene expression profiling. *Nature*, 403:503–511, 2000.

[7] U. Alon, N. Barkai, D. A. Notterman, K. Gish, S. Ybarra, D. Mack, and A. J. Levine. Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *Proceedings of National Academy of Sciences of the United States of American*, 96:6745–6750, 1999.

[8] D.B. Altman. *Practical Statistics for Medical Research*. Chapman and Hall, London, 1991.

[9] T. Ando and M. Katayama. Selection of causal gene sets from transcriptional profiling by FNN modeling and prediction of lymphoma outcome. *Proceedings of 13th International Conference on Genome Informatics*, pages 278–279, Tokyo, Japan, December 2002.

[10] 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. *Nature Genetics*, 30(1):41–47, 2002.

[11] E. Bair and R. Tibshirani. Semi-supervised methods to predict patient survival from gene expression data. *PLoS Biology*, 2:0511–0522, April 2004.

[12] P. Baldi and A.D. Long. A Bayesian framework for the analysis of microarray expression data: regularized t-test and statistical inferences of gene changes. *Bioinformatics*, 17(6):509–519, 2001.

[13] E. Bauer and R. Kohavi. An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. *Machine Learning*, 36(1/2):105–139, 1999.

[14] D.G. Beer, S.L. Kardia, C.C. Huang, T.J. Giordano, A.M. Levin, D.E. Misek, L. Lin, G. Chen, T.G. Gharib, D.G. Thomas, M.L. Lizyness, R. Kuick, S. Hayasaka, J.M. Taylor, M.D. Iannettoni, M.B. Orringer, and S. Hanash. Gene-expression profiles predict survival of patients with lung adenocarcinoma. *Nature Medicine*, 8(8):816–823, 2002.

[15] C.L. Blake and P.M. Murphy. The UCI machine learning repository. *http://www.cs.uci.edu/˜mlearn/MLRepository.html*, 1998.

[16] K.M. Bloch and G.R. Arce. Nonlinear correlation for the analysis of gene expression data. In *Proceedings of the 2002 Workshop on Genomic Signal Processing and Statistics*. Raleigh, North Carolina, October, 2002.

[17] A.C. Borczuk, L. Gorenstein, K.L. Walter, A.A. Assaad, L. Wang, and C.A. Powell. Non-small-cell lung cancer molecular signatures recapitulate lung developmental pathway. *American Journal of Pathology*, 163(5):1949–1960, 2003.

[18] S. Brackenridge and N.J. Proudfoot. Recruitment of a basal polyadenylation factor by the upstream sequence element of the human lamin b2 polyadenylation signal. *Molecular Cell Biology*, 20:2660–2669, 2000.

[19] L. Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

[20] L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[21] L. Breiman. Manual on setting up, using, and understanding random forests v3.1. *http://oz.berkeley.edu/users/breiman/Using_random_forests_V3.1.pdf*, 2002.

[22] L. Breiman, J.H. Friedman, R.A. Olshen, and C.J. Stone. *Classification and regression trees*. Wadsworth International Group, Belmont, CA., 1984.

[23] M.P. Brown, W.N. Grundy, D. Lin, N. Cristianini, C.W. Sugnet, T.S. Furey, M. Ares, Jr., and Haussler D. Knowledge-based analysis of microarray gene expression data using support vector machines. *Proceedings of the National Academy of Science*, 97(1):262–267, 2000.

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

[25] J. Cheng, C. Hatzis, H. Hayashi, M.A. Krogel, S. Morishita, D. Page, and J. Sese. KDD Cup Report. *ACM SIGKDD Explorations*, 3(2):47–64, 2001.

[26] G. Chu, B. Narasimhan, R. Tibshirani, and V.G. Tusher. *SAM user guide and technical document*. Stanford University, 2004.

[27] A. Cigan, L. Feng, and T. Donahue. tRNA functions in directing the scanning ribosome to the start site of translation. *Science*, 242:93–97, 1988.

[28] D.F. Colgan and Manley J.L. Mechanism and regulation of mRNA polyadenylation. *Genes & Development*, 11:2755–2766, 1997.

[29] T.P. Conrads, M. Zhou, E.F. 3rd Petricoin, L. Liotta, and Veenstra T.D. Cancer diagnosis using proteomic patterns. *Expert Review of Molecular Diagnostics*, 3(4):411–420, 2003.

[30] D.R. Cox. Regression models and life-tables (with discussion). *Journal of the Royal Statistical Society*, B34:187–220, 1972.

[31] A.C. Culhane, G. Perriere, Considine E.C., T.G. Cotter, and D.G. Higgins. Between-group analysis of microarray data. *Bioinformatics*, 18(12):1600–1608, 2002.

[32] J.L. DeRisi, V.R. Iyer, and P.O. Brown. Exploring the metabolic and genetic control of gene expression on a genomic scale. *Science*, 278:680–685, 1997.

[33] T.G. Dietterich. Ensemble methods in machine learning. In Kittler J. and F. Roli, editors, *Multiple Classifier Systems*, pages 1–15, 2000.

[34] T.G. Dietterich. An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. *Machine Learning*, 40(2):139–157, 2000.

[35] 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*, 97(457):77–87, 2002.

[36] U. Fayyad and K. Irani. Multi-interval discretization of continuous-valued attributes for classification learning. *Proceedings of the 13th International Joint Conference on Artificial Intelligence*, pages 1022–1029, 1993.

[37] E. Fix and Hodges J.L. Discriminatory analysis: non-parametric discrimination: consistency properties. *Technical Report 21-49-004(4)*, 1951.

[38] Y. Freund and R. E. Schapire. Experiments with a new boosting algorithm. In *Proceedings of 13th International Conference on Machine Learning*, pages 148–156. Morgan Kaufmann, 1996.

[39] T. S. Furey, N. Cristianini, N. Duffy, D. W. Bednarski, M. Schummer, and D. Haussler. Support vector machine classification and validation of cancer tissue samples using microarray expression data. *Bioinformatics*, 16(10):906–914, 2000.

[40] O.K. Glebov, L.M. Rodriguez, K. Nakahara, J. Jenkins, J. Cliatt, C.J. Humbyrd, J. DeNobile, P. Soballe, R. Simon, G. Wright, P. Lynch, S. Patterson, H. Lynch, S. Gallinger, A. Buchbinder, G. Gordon, E. Hawk, and I.R. Kirsch. Distinguishing right from left colon by the pattern of gene expression. *Cancer Epidemiology Biomarkers & Prevention*, 12(8):755–762, August 2003.

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

[42] G.J. Gordon, R.V. Jensen, L.L. Hsiao, S.R. Gullans, J.E. Blumenstock, S. Ramaswamy, W.G. Richards, D.J. Sugarbaker, and R. Bueno. Translation of microarray data into clinically relevant cancer diagnostic tests using gene expression ratios in lung cancer and mesothelioma. *Cancer Research*, 62(17):4963–4967, 2002.

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

[44] M.A. Hall. *Correlation-based feature selection for machine learning*. PhD thesis, Department of Computer Science, University of Waikato, Hamilto, New Zealand, 1998.

[45] M.A. Hall. Correlation-based feature selection for discrete and numeric class machine learning. *Proceedings of the Seventeenth International Conference on Machine Learning*, 2000.

[46] M.A. Hall and G. Holmes. Benchmarking attribute selection techniques for discrete class data mining. *IEEE Transaction on Knowledge and Data Engineering*, 15(3):in press, May/June 2003.

[47] C.A. Harrington, C. Rosenow, and J. Retief. Monitoring gene expression using DNA microarrays. *Current Opinion in Microbiology*, 3(3):285–291, 2000.

[48] A. G. Hatzigeorgiou. Translation initiation start prediction in human cDNAs with high accuracy. *Bioinformatics*, 18(2):343–350, 2002.

[49] J. Herrero, R. Diaz-Uriarte, and J. Dopazo. Gene expression data preprocessing. *Bioinformatics*, 19(5):655–656, 2003.

[50] I. Inza, B. Sierra, Blanco R., and P. Larraaga. Gene selection by sequential search wrapper approaches in microarray cancer class prediction. *Journal of Intelligent and Fuzzy Systems*, 2002.

[51] J. Jaeger, R. Sengupta, and W.L. Ruzzo. Improved gene selection for classification of microarrays. *Pacific Symposium on Biocomputing*, 8:53–64, 2003.

[52] M.V. Johns. An empirical Bayes approach to non-parametric two-way classification. In H. Solomon, editor, *Studies in item analysis and prediction*, Palo Alto, CA, 1961. Stanford Univeristy Press.

[53] I.T. Jolliffe. *Principal Component Analysis*. Springer-Verlag, 1986.

[54] 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. *Natural Medicine*, 7(6):673–679, June 2001.

[55] R. Kohavi and G.H. John. Wrappers for feature subset selection. *Artificial Intelligence*, 97:273–324, 1997.

[56] M. Kozak. An analysis of 5'-noncoding sequences from 699 vertebrate messenger RNAs. *Nucleic Acids Research*, 15:8125–8148, 1987.

[57] M. Kozak. The scanning model for translation: an update. *Journal of Cell Biology*, 108(2):229–241, 1989.

[58] M. Kozak. Interpreting cDNA sequences: some insights from studies on translation. *Mammalian Genome*, 7:563–574, 1996.

[59] M. Kozak. Initiation of translation in prokaryotes and eukaryotes. *Gene*, 234:187–208, 1999.

[60] M. LeBlanc, C. Kooperberg, T.M. Grogan, and T.P. Miller. Directed indices for exploring gene expression data. *Bioinformatics*, 19(6):686–693, 2003.

[61] M. Legendre and D. Gautheret. Sequence determinants in human polyadenylation site selection. *BMC Genomics*, 4(1):7, 2003.

[62] J. Li and H. Liu. Ensembles of cascading trees. In *Proceedings of 3rd IEEE International Conference on Data Mining*, pages 585–588, Melbourne, Florida, USA, November 19 - 22, 2003.

[63] J. Li, H. Liu, S.-K. Ng, and L. Wong. Discovery of significant rules for classifying cancer diagnosis data. *Bioinformatics*, 19(Suppl. 2):ii93–ii102, 2003.

[64] J. Li, H. Liu, and L. Wong. Mean-entropy discretized features are effective for classifying high-dimensional biomedical data. *The 3rd ACM SIGKDD Workshop on Data Mining*, pages 17–24, August 2003.

[65] J. Li, H. Liu, and L. Wong. Use of built-in features in the interpretation of high-dimensional cancer diagnosis data. In *Proceedings of the Second Asia Pacific Bioinformatics Conference (APBC2004)*, pages 67–74, Dunedin, New Zealand, January 18-22 2004.

[66] J. Li and L. Wong. Solving the fragmentation problem of decision trees by discovering boundary emerging patterns. *Proceedings of IEEE International Conference on Data Mining*, pages 653–656, Maebashi City, Japan, December 2002.

[67] L. Li, T.A. Darden, C.R. Weinberg, and L.G. Pedersen. Gene assessment and sample classification for gene expression using a genetic algorithm/k-nearest neighbor method. *Combinatorial Chemistry & High Throughput Screening*, 4(8):727–739, 2001.

[68] Y. Li, C. Campbell, and M. Tipping. Bayesian automatic relevance determination algorithms for classifying gene expression data. *Bioinformatics*, 18(10):1332–1339, 2002.

[69] H. Liu, H. Han, J. Li, and L. Wong. An in-silico method for prediction of polyadenylation signals in human sequences. *Proceedings of 14th International Conference on Genome Informatics*, pages 84–93, December 2003.

[70] H. Liu, H. Han, J. Li, and L. Wong. Using amino acid patterns to accurately predict translation initiation sites. *In-Silico Biology*, 4, 0022, 2004.

[71] H. Liu and R. Setiono. Chi2: Feature selection and discretization of numeric attributes. *Proceedings of the IEEE 7th International Conference on Tools with Artificial Intelligence*, pages 388–391, November 1995.

[72] H. Liu and L. Wong. Data mining tools for biological sequences. *Journal of Bioinformatics and Computational Biology*, 1(1):139–168, April 2003.

[73] D.J. Lockhart, H. Dong, M.C. Byrne, M.T. Follettie, M.V. Gallo, M.S. Chee, M. Mittmann, C. Wang, M. Kobayashi, H. Horton, and E.L. Brown. Expression monitoring by hybridization to high-density oligonucleotide arrays. *Nature Biotechnology*, 14:1675–1680, 1996.

[74] M. Lunn and D.R. McNeil. Applying Cox regression to competing risks. *Biometrics*, 51:1524–532, 1995.

[75] M.A. McDevitt, R.P. Hart, W.W. Wong, and J.R. Nevins. Sequence capable of restoring poly(A) site function define two distinct downstream elements. *EMBO Journal*, 1(5):2907–2931, 1986.

[76] D. Meyer, F. Leisch, and K. Hornik. Benchmarking support vector machines. *Report Series, Vienna University of Economics and Business Administration*, Report No. 78, November 2002.

[77] L.D. Miller, P.M. Long, L. Wong, S. Mukherjee, L.M. McShane, and E.T. Liu. Optimal gene expression analysis by microarrays. *Cancer Cell*, 2:353–361, November 2002.

[78] T.M. Mitchell. *Machine Learning*. McGrawHill, USA, 1997.

[79] A. Moreira, Y. Takagaki, S. Brackenridge, M. Wollerton, J.L. Manley, and N.J. Proudfoot. The upstream sequence element of the C2 complement poly(A) signal activates mRNA 3' end formation by two distinct mechanisms. *Genes Development*, 12(16):2522–2534, 1998.

[80] S. Mukherjee, P. Tamayo, J. Mesirow, D. Slonim, A. Verri, and T. Poggio. Support vector machine classification of microarray data. Technical report, CBCL Paper 182/AL Memo 1676 MIT, 1999.

[81] S.K. Murthy, S. Kasif, and S. Salzberg. A system for induction of oblique decision trees. *Journal of Artificial Intelligence Research*, 2:1–32, 1994.

[82] D.V. Nguyen and D.M. Rocke. Tumor classification by partial least squares using microarray gene expression data. *Bioinformatics*, 18(1):39–50, 2002.

[83] T. Nishikawa, T. Ota, and T. Isogai. Prediction whether a human cDNA sequence contains initiation codon by combining statistical information and similarity with protein sequences. *Bioinformatics*, 16(11):960–967, 2000.

[84] 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 Research*, 61:3124–3130, 2001.

[85] A.B. Olshen and A.N. Jain. Deriving quantitative conclusions from microarray expression data. *Bioinformatics*, 18(7):961–970, 2002.

[86] W. Pan. A comparative review of statistical methods for discovering differentially expressed genes in replicated microarray experiments. *Bioinformatics*, 18(4):546–554, 2002.

[87] P.J. Park, M. Pagano, and M. Bonetti. A non-parametric scoring algorithm for identifying informative genes from microarray data. *Pacific Symposium on Biocomputing*, pages 52–63, 2001.

[88] P.J. Park, L. Tian, and S. Kohane. Linking gene expression data with patient survival times using partial least squares. *Bioinformatics*, 18(Suppl 1):S120–S127, 2002.

[89] A.G. Pedersen and H. Nielsen. Neural network prediction of translation initiation sites in eukaryotes: perspectives for est and genome analysis. *Proceedings of the 5th International Conference on Intelligent Systems for Molecular Biology*, pages 226–233, 1997.

[90] S. Peri and A. Pandey. A reassessment of the translation initiation codon in vertebrates. *TRENDS in Genetics*, 17(12):685–687, December 2001.

[91] T.V. Perneger. What is wrong with bonferroni adjustments. *British Medical Journal*, 136:1236–1238, 1998.

[92] E. F. Petricoin, A. M. Ardekani, B. A. Hitt, P. J. Levine, V. A. Fusaro, S. M. Steinberg, G. B. Mills, C. Simone, D. A. Fishman, E. C. Kohn, and L. A. Liotta. Use of proteomic patterns in serum to identify ovarian cancer. *The Lancet*, 359:572–577, February 2002.

[93] J. Platt. Fast training of support vector machines using sequential minimal optimization. In B. Schlkopf, C. Burges, and Smola A., editors, *Advances in Kernel Methods - Support Vector Learning*, pages 185–208. MIT Press, 1998.

[94] K.D. Pruitt, K.S. Katz, H. Sicotte, and Maglott D.R. Introducing RefSeq and LocusLink: curated human genome resources at the NCBI. *Trends Genet*, 1(16):44–47, 2000.

[95] C.H. Pui and W.E. Evans. Acute lymphoblastic leukemia. *The New England Journal of Medicine*, 339:605–615, 1998.

[96] J.R. Quinlan. Induction of decision trees. *Machine Learning*, 1:81–106, 1986.

[97] J.R. Quinlan. *C4.5: program for machine learning*. Morgan Kaufmann, San Mateo, CA., USA, 1993.

[98] J.R. Quinlan. Bagging, boosting, and C4.5. In *Proceedings of 13th National Conference on Artificial Intelligence*, pages 725–730. AAAI Press/MIT Press, 1996.

[99] T.H. Rainer, P.K. Lam, E.M. Wong, and R.A. Cocks. Derivation of a prediction rule for post-traumatic acute lung injury. *Resuscitation*, 42(3):187–196, 1999.

[100] J. Rice. *Mathematical Statistics and Data Analysis*. Wadsworth, Pacific Grove, CA, 1988.

[101] J. Rissanen. Modeling by shortest data description. *Automatica*, 14:465–471, 1978.

[102] A. Rosenwald, G. Wright, WC. Chan, JM. Connors, E. Campo, RI. Fisher, RD. Gascoyne, HK. Muller-Hermelink, Smel, EB. , JM. Giltnane, EM. Hurt, H. Zhao, L. Averett, L. Yang, WH. Wilson, ES. Jaffe, R. Simon, RD. Klausner, J. Powell, PL. Duffey, DL. Longo, TC. Greiner, DD. Weisenburger, WG. Sanger, BJ. Dave, JC. Lynch, J. Vose, JO. Armitage, E. Montserrat, A. Lopez-Guillermo, TM. Grogan, TP. Miller, M. LeBlanc, G. Ott, S. Kvaloy, J. Delabie, H. Holte, P. Krajci, T. Stokke, and LM. Staudt. The use of molecular profiling to predict survival after chemotherapy for diffuse large-B-cell lymphoma. *The New England Journal of Medicine*, 346(25):1937–1947, 2002.

[103] A.A. Salamov, T. Nishikawa, and M.A. Swindells. Assessing protein coding region integrity in cDNA sequencing projects. *Bioinformatics*, 14(5):384–390, 1998.

[104] S. Salzberg. Locating protein coding regions in human DNA using a decision tree algorithm. *Journal of Computational Biology*, 2(3):473–485, 1995.

[105] R. Sandy. *Statistics for Business and Economics*. McGrawHill, USA, 1989.

[106] E.M. Santos and H.M. Gomes. A comparative study of polynomial kernel SVM applied to appearance-based object recognition. *International Workshop on Pattern Recognition with Support Vector Machines*, August, 2002.

[107] R. E. Schapire. The boosting approach to machine learning: An overview. In *Proceedings of MSRI Workshop on Nonlinear Estimation and Classification*, 2002.

[108] B. Scholkop and A.J. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA., 2002.

[109] M. Schummer, W. Ng, R. Bumgarner, P. Nelson, B. Schummer, D. Bednarski, L. Hassell, R. Baldwin, B. Karlan, and L. Hood. Comparative hybridization of an array of 21,500 ovarian cDNAs for the discovery of genes overexpressed in ovarian carcinomas. *Gene*, 238:375–385, 1999.

[110] H.P. Selker, J.L. Griffith, S. Patil, W.J. Long, and R.B. D'Agostino. A comparison of performance of mathematical predictive methods for medical diagnosis: identifying acute cardiac ischemia among emergency department patients. *Journal of Investigative Medicine*, 43(5):468–476, 1995.

[111] R. Shamir and R. Sharan. CLICK: A clustering algorithm for gene expression analysis. *Proceedings of the Eighteenth International Conference on Intelligent Systems for Molecular Biology*, 2000.

[112] C.E. Shannon. *The Mathematical Theory of Communication*. University of Illinois Press, Urbana, USA, 1949.

[113] M.A. Shipp and D.P. Harrington. A predictive model for aggressive non-Hodgkin's lymphoma. *The New England Journal Of Medicine*, 329:987–994, 1993.

[114] 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. *Nature Medicine*, 8(1):68–74, 2002.

[115] D. Singh, P.G. Febbo, K. Ross, D.G. Jackson, J. Manola, C. Ladd, P. Tamayo, A.A. Renshaw, A.V. D'Amico, J.P. Richie, E.S. Lander, M. Loda, P.W. Kantoff, T.R. Golub, and W.R. Sellers. Gene expression correlates of clinical prostate cancer behavior. *Cancer Cell*, 1:203–209, 2002.

[116] D. K. Slonim, P. Tamayo, T. R. Golub, J. P. Mesirov, and E. S Lander. Class prediction and discovery using gene expression data. In *Proceedings of 4th Annual International Conference on Computational Molecular Biology*, pages 263–272, Tokyo, Japan, 2000.

[117] L.A. Soinov, M.A. Krestyaninova, and A. Brazma. Towards reconstruction of gene networks from expression data by supervised learning. *Genome Biology*, 4(1):R6.1–9, 2003.

[118] J.M. Sorace and M. Zhan. A data review and re-assessment of ovarian cancer serum proteomic profiling. *BMC Bioinformatics*, 4:24, 2003.

[119] J.E. Tabaska and M.Q. Zhang. Detection of polyadenylation signals in human DNA sequences. *Gene*, 231:77–86, 1999.

[120] N.R. Temkin, R. Holubkov, J.E. Machamer, H.R. Winn, and S.S. Dikmen. Classification and regression trees (CART) for prediction of function at 1 year following head trauma. *Journal of Neurosurgery*, 82(5):764–771, 1995.

[121] J.G. Thomas, J.M. Olson, S.J. Tapscott, and L.P. Zhao. An efficient and robust statistical modeling approach to discover differentially expressed genes using genomic expression profiles. *Genome Research*, 11:1227–1236, 2001.

[122] C.J. Thornton. *Techniques in Computational Learning*. Chapman and Hall, London, 1992.

[123] R. Tibshirani, T. Hastie, B. Narasimhan, and G. Chu. Diagnosis of multiple cancer types by shrunken centroids of gene expression. *Proceedings of the National Academy of Science (PNAS)*, 99:6567–6572, 2002.

[124] R. Tibshirani, T. Hastie, B. Narasimhan, and G. Chu. Class prediction by nearest shrunken centroids, with applications to DNA microarrays. *Statistical Science*, 18(1):104–117, 2003.

[125] O.G. Troyanskaya, M. Cantor, G. Sherlock, T. Brown, P.O. Hastie, R. Tibshirani, D. Botstein, and R.B. Altman. Missing value estimation methods for DNA microarrays. *Bioinformatics*, 17(6):520–525, 2001.

[126] O.G. Troyanskaya, M.E. Garber, P.O. Brown, D. Botstein, and R.B. Altman. Nonparametric methods for identifying differentially expressed genes in microarray data. *Bioinformatics*, 18(11):1454–1461, 2002.

[127] V.G. Tusher, R. Tibshirani, and G. Chu. Significance analysis of microarrays applied to the ionizing radiation response. *Proceedings of the National Academy of Science (PNAS)*, 98:5116–5121, 2001.

[128] P.E. Utgoff and C.E. Brodley. Linear machine decision trees. Technical report, Department of Computer Science, University of Massachusetts, Amherst, MA., 1991.

[129] M.J. van de Vijver, Y.D. He, L.J. van't Veer, H. Dai, A.A. Hart, D.W. Voskuil, G.J. Schreiber, J.L. Peterse, C. Roberts, M.J Marton, M. Parrish, D. Atsma, A. Witteveen, A. Glas, L. Delahaye, T. van der Velde, H. Bartelink, S. Rodenhuis, E.T. Rutgers, S.H. Friend, and R. Bernards. A gene-expression signature as a predictor of survival in breast cancer. *Journal of New England Medicine*, 347(25):1999–2009, 2002.

[130] V.N. Vapnik. *The Natural of Statistical Learning Theory*. Springer, 1995.

[131] J. Weston, S. Mukherjee, O. Chapelle, M. Pontil, T. Poggio, and V. Vapnik. Feature selection for SVMs. *Advances in Neural Information Processing Systems*, 13:668–674, 2001.

[132] J. Weston, F. Perez-Cruz, O. Bousquet, O. Chapelle, S. Elisseeff, and B. Scholkopf. Feature selection and transduction for prediction of molecular bioactivity for drug design. *Bioinformatics*, 19(6):764–771, 2003.

[133] F. Wilcoxon. Individual comparisons by ranking methods. *Biometrics*, 1:80–83, 1945.

[134] H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementation*. Morgan Kaufmann, San Mateo, CA, 2000.

[135] E.P. Xing, M.I. Jordan, and R.M. Karp. Feature selection for high-dimensional genomic microarray data. *Proceedings of the Eighteenth International Conference on Machine Learning*, 2001.

[136] E.P. Xing and R.M. Karp. Cliff: Clustering of high-dimensional microarray data via iterative feature filtering using normalized cuts. *Proceedings of The Ninth International Conference on Intelligence Systems for Molecular Biology, published on Bioinformatics*, 17(suppl):S306–S315, 2001.

[137] Y. Xu, V. Olman, and D. Xu. Clustering gene expression data using a graph-theoretic approach: an application of minimum spanning tree. *Bioinformatics*, 18(4):536–545, 2002.

[138] Y. Yang, L.J. Mou, N. Liu, and M.S. Tsao. Autotaxin in expression in non-small-cell lung cancer. *American Journal of Respiratory Cell and Molecular Biology*, 21(2):216–222, 1999.

[139] Y.H. Yang, S.D. Dudoit, P. Luu, D.M. Li, V. Peng, and J. Ngai. Normalization for cDNA microarray data: a robust composite method addressing single and multiple slide systematic variation. *Nucleic Acids Research*, 30(4):e15, 2002.

[140] E.-J. Yeoh, M. E. Ross, S. A. Shurtleff, W. K. Williams, D. Patel, R. Mahfouz, F. G. Behm, S. C. Raimondi, M. V. Relling, A. Patel, C. Cheng, D. Campana, D. Wilkins, X. Zhou, J. Li, H. Liu, C.-H. Pui, W. E. Evans, C. Naeve, L. Wong, and J. R. Downing. Classification and subtype discovery and prediction of outcome in pediatric acute

lymphoblastic leukemia by gene expression profiling. *Cancer Cell*, 1:133–143, March 2002.

[141] M.I. Zarudnaya, I.M. Kolomiets, A.L. Potyahaylo, and D.M. Hovorun. Downstream elements of mammalian pre-mRNA polyadenylation signals: primary, secondary and higher-order structures. *Nucleic Acids Research*, 1(31):1375–1386, 2003.

[142] F. Zeng, H.C. Yap, and L. Wong. Using feature generation and feature selection for accurate prediction of translation initiation sites. *Proceedings of 13th International Conference on Genome Informatics*, pages 192–200, Tokyo, Japan, December 2002.

[143] M.Q. Zhang. Identification of human gene core promoter in silico. *Genome Research*, 8:319–326, 1998.

[144] J. Zhao, L. Hyman, and C. Moore. Formation of mRNA 3' ends in eukaryotes: mechanism, regulation, and interrelationships with other steps in mRNA synthesis. *Microbiology and Molecular Biology Reviews*, 63(2):405–445, 1999.

[145] A. Zien, G. Raetsch, S. Mika, B. Schoelkopf, C. Lemmen, A. Smola, T. Lengauer, and K.-R. Mueller. Engineering support vector machine kernels that recognize translation initiation sites. *Bioinformatics*, 16(9):799–807, September 2000.

Appendix A

Lists of Genes Identified in Chapter 5

Table A.1: 54 common genes selected by each fold of ERCOF in 10-fold cross validation test for prostate cancer data set.

Probe name	Accession number	Gene name
40435_at	J03592	Human ADP/ATP translocase mRNA, 3'end, clone pHAT8
40419_at	X85116	H.sapiens epb72 gene exon 1
31444_s_at	M62895	Human lipocortin (LIP) 2 pseudogene mRNA, complete cds-like region
37720_at	M22382	Human mitochondrial matrix protein P1 (nuclear encoded) mRNA, complete cds
32634_s_at	U38260	Human islet cell autoantigen ICAp69 mRNA, complete cds
34608_at	M24194	Human MHC protein homologous to chicken B complex protein mRNA, complete cds
33137_at	Y13622	Homo sapiens mRNA for latent transforming growth factor-beta binding protein-4
40436_g_at	J03592	Human ADP/ATP translocase mRNA, 3'end, clone pHAT8
34784_at	Z83844	Human DNA sequence from clone 37E16 on chromosome 22 Contains a novel gene, a gene similar to SH3-binding protein, LGALS1 (14 kDa beta-galactoside-binding lectin) gene, part of a gene similar to mouse p116Rip, ESTs, STSs, GSSs and two CpG islands
1676_s_at	M55409	Homo sapiens pancreatic tumor-related protein mRNA, partial cds
36587_at	Z11692	H.sapiens mRNA for elongation factor 2
33614_at	X80822	H.sapiens mRNA for ORF
38814_at	AF038954	Homo sapiens vacuolar H(+)-ATPase subunit mRNA, complete cds
33668_at	AF037643	Homo sapiens 60S ribosomal protein L12 (RPL12) pseudogene, partial sequence
40024_at	D86640	Homo sapiens mRNA for stac, complete cds
39756_g_at	Z93930	Human DNA sequence from clone 292E10 on chromosome 22q11-12. Contains the XBP1 gene for X-box binding protein 1 (TREB5), ESTs, STSs, GSSs and a putative CpG island
34853_at	AB007865	Homo sapiens KIAA0405 mRNA, complete cds
33820_g_at	X13794	H.sapiens lactate dehydrogenase B gene exon 1 and 2
40856_at	U29953	Human pigment epithelium-derived factor gene, complete cds
31538_at	M17885	Human acidic ribosomal phosphoprotein P0 mRNA, complete cds
36601_at	M33308	Human vinculin mRNA, complete cds
33134_at	AB011083	Homo sapiens mRNA for KIAA0511 protein, partial cds
32076_at	D83407	ZAKI-4 mRNA in human skin fibroblast, complete cds
31545_at	AL031228	dJ1033B10.4 (40S ribosomal protein S18 (RPS18, KE-3))
33328_at	W28612	49b3 Homo sapiens cDNA
39416_at	U90913	Human clone 23665 mRNA sequence
40607_at	U97105	Homo sapiens N2A3 mRNA, complete cds
769_s_at	D00017	Homo sapiens mRNA for lipocortin II, complete cds
32412_at	M13934	Human ribosomal protein S14 gene, complete cds
37819_at	AF007130	Homo sapiens clone 23750 unknown mRNA, partial cds
1521_at	X17620	Human mRNA for Nm23 protein, involved in developmental regulation (homolog. to Drosophila Awd protein)
1513_at		Antigen, Prostate Specific, Alt. Splice Form 3
39939_at	D21337	Human mRNA for collagen
35776_at	AF064243	Homo sapiens intersectin short form mRNA, complete cds
31527_at	X17206	Human mRNA for LLRep3
33408_at	AB023151	Homo sapiens mRNA for KIAA0934 protein, partial cds
34840_at	AI700633	we38g03.x1 Homo sapiens cDNA, 3'end
39315_at	D13628	Human mRNA for KIAA0003 gene, complete cds
35119_at	X56932	H.sapiens mRNA for 23 kD highly basic protein
575_s_at	M93036	Human (clone 21726) carcinoma-associated antigen GA733-2 (GA733-2) mRNA, exon 9 and complete cds
262_at	M21154	Human S-adenosylmethionine decarboxylase mRNA, complete cds
37639_at	X07732	Human hepatoma mRNA for serine protease hepsin
32243_g_at	AL038340	DKFZp566K192_s1 Homo sapiens cDNA, 3'end
36864_at	AJ001625	Homo sapiens mRNA for Pex3 protein
38044_at	AF035283	Homo sapiens clone 23916 mRNA sequence
38098_at	D80010	Human mRNA for KIAA0188 gene, partial cds
39366_at	N36638	yx88f05.r1 Homo sapiens cDNA, 5'end

Table A.2: 54 common genes selected by each fold of ERCOF in 10-fold cross validation test for prostate cancer data set (continued 1).

Probe name	Accession number	Gene name
32206_at	AB007920	Homo sapiens mRNA for KIAA0451 protein, complete
39550_at	AB011156	Homo sapiens mRNA for KIAA0584 protein, partial
34304_s_at	AL050290	Homo sapiens mRNA; cDNA DKFZp586G1923 (from clone DKFZp586G1923)
37730_at	U22055	Human 100 kDa coactivator mRNA, complete cds
41288_at	AL036744	DKFZp564I1663_r1 Homo sapiens cDNA, 5'end
31583_at	X67247	H.sapiens rpS8 gene for ribosomal protein S8
172_at	U57650	Human SH2-containing inositol 5-phosphatase (hSHIP) mRNA, complete cds

Table A.3: 39 common m/z identities among top 50 entropy measure selected features in 10-fold cross validation on ovarian cancer proteomic profiling. Their corresponding Wilcoxon test p-values are derived from paper [118].

M/Z identity	Wicoxon p-value	Entropy measure
244.95245	1.16115E-30	0.13998
245.8296	7.59262E-30	0.16299
245.24466	1.59454E-30	0.17846
244.66041	1.30324E-30	0.18037
245.53704	2.25194E-30	0.18209
435.46452	5.16697E-30	0.23104
246.41524	3.70287E-29	0.23574
246.12233	1.70497E-29	0.23743
247.00158	1.00124E-28	0.23968
417.73207	1.03527E-27	0.25183
434.68588	1.7291E-29	0.25791
435.07512	3.1774E-30	0.25839
435.85411	1.65702E-29	0.26475
246.70832	6.49125E-29	0.27451
261.88643	6.58307E-29	0.28096
418.11364	6.48304E-27	0.28419
247.295	1.45824E-28	0.30174
247.88239	1.30577E-27	0.31365
434.29682	9.27807E-28	0.31648
262.18857	2.34772E-27	0.32680
261.58446	1.5817E-27	0.33865
247.58861	2.33737E-28	0.34268
244.36855	2.11132E-26	0.34343
436.24386	5.43042E-28	0.34656
464.76404	5.64673E-26	0.35072
464.36174	2.34956E-26	0.36228
222.69673	7.50798E-26	0.37045
417.35068	1.30456E-26	0.37647
463.95962	1.13655E-25	0.38043
465.16651	5.44957E-25	0.38914
222.41828	4.20921E-27	0.39731
222.14001	3.27501E-25	0.40447
418.49538	9.26396E-25	0.40599
262.49088	2.6516E-23	0.41769
436.63379	2.16083E-25	0.42559
25.589892	1.80877E-24	0.43315
463.55767	4.42152E-23	0.44623
4003.6449	5.09873E-22	0.45153
220.75125	3.25692E-24	0.46876

Table A.4: 280 genes identified by ERCOF from training samples on ALL-AML leukaemia data set. Probes with bold font were also reported in [41].

Probe	Gene name
X95735_at	Zyxin
M55150_at	FAH Fumarylacetoacetate
M31166_at	PTX3 Pentaxin-related gene, rapidly induced by IL-1 beta
M27891_at	CST3 Cystatin C (amyloid angiopathy and cerebral hemorrhage)
U46499_at	GLUTATHIONE S-TRANSFERASE, MICROSOMAL
L09209_s_at	APLP2 Amyloid beta (A4) precursor-like protein 2
X70297_at	CHRNA7 Cholinergic receptor, nicotinic, alpha polypeptide 7
M77142_at	NUCLEOLYSIN TIA-1
J03930_at	ALKALINE PHOSPHATASE, INTESTINAL PRECURSOR
M92287_at	CCND3 Cyclin D3
U22376_cds2_s_at	C-myb gene extracted from Human (c-myb) gene, complete primary cds, and five complete alternatively spliced cds
M27783_s_at	ELA2 Elastatse 2, neutrophil
D14874_at	ADM Adrenomedullin
M16038_at	LYN V-yes-1 Yamaguchi sarcoma viral related oncogene homolog
U50136_rna1_at	Leukotriene C4 synthase (LTC4S) gene
M98399_s_at	CD36 CD36 antigen (collagen type I receptor, thrombospondin receptor)
M21551_rna1_at	Neuromedin B mRNA
Y12670_at	LEPR Leptin receptor
M83652_s_at	PFC Properdin P factor, complement
M23197_at	CD33 CD33 antigen (differentiation antigen)
U46751_at	Phosphotyrosine independent ligand p62 for the Lck SH2 domain mRNA
D88422_at	CYSTATIN A
M54995_at	PPBP Connective tissue activation peptide III
U02020_at	Pre-B cell enhancing factor (PBEF) mRNA
M31523_at	TCF3 Transcription factor 3 (E2A immunoglobulin enhancer binding factors E12/E47)
X04085_rna1_at	Catalase (EC 1.11.1.6) 5'flank and exon 1 mapping to chromosome 11, band p13 (and joined CDS)
M81933_at	CDC25A Cell division cycle 25A
U12471_cds1_at	Thrombospondin-p50 gene extracted from Human thrombospondin-1 gene, partial cds
M91432_at	ACADM Acyl-Coenzyme A dehydrogenase, C-4 to C-12 straight chain
X59417_at	PROTEASOME IOTA CHAIN
M12959_s_at	TCRA T cell receptor alpha-chain
X74262_at	RETINOBLASTOMA BINDING PROTEIN P48
L27584_s_at	CAB3b mRNA for calcium channel beta3 subunit
HG4316-HT4586_at	Transketolase-Like Protein
J05243_at	SPTAN1 Spectrin, alpha, non-erythrocytic 1 (alpha-fodrin)
M31303_rna1_at	Oncoprotein 18 (Op18) gene
X62654_rna1_at	ME491 gene extracted from H.sapiens gene for Me491/CD63 antigen
X90858_at	Uridine phosphorylase
M84526_at	DF D component of complement (adipsin)
J04615_at	SNRPN Small nuclear ribonucleoprotein polypeptide N
D26308_at	NADPH-flavin reductase
L08177_at	CMKBR7 Chemokine (C-C) receptor 7
X14008_rna1_f_at	Lysozyme gene (EC 3.2.1.17)
X87613_at	Skeletal muscle abundant protein
M80254_at	PEPTIDYL-PROLYL CIS-TRANS ISOMERASE, MITOCHONDRIAL PRECURSOR
M96326_rna1_at	Azurocidin gene
J04990_at	CATHEPSIN G PRECURSOR
U62136_at	Putative enterocyte differentiation promoting factor mRNA, partial cds
D10495_at	PRKCD Protein kinase C, delta
X52142_at	CTPS CTP synthetase
U73737_at	GTBP DNA GT mismatch-binding protein
X74801_at	T-COMPLEX PROTEIN 1, GAMMA SUBUNIT
U32944_at	Cytoplasmic dynein light chain 1 (hdlc1) mRNA
X15949_at	IRF2 Interferon regulatory factor 2

Table A.5: 280 genes identified by ERCOF from training samples on ALL-AML leukaemia data set. Probes with bold font were also reported in [41] (continued 1).

Probe	Gene name
M31158_at	PRKAR2B Protein kinase, cAMP-dependent, regulatory, type II, beta
M15780_at	GB DEF = DNA/endogenous human papillomavirus type 16 (HPV) DNA, right flank and viral host junction
X62320_at	GRN Granulin
D49950_at	Liver mRNA for interferon-gamma inducing factor(IGIF)
U37055_rna1_s_at	Hepatocyte growth factor-like protein gene
D88378_at	Proteasome inhibitor hPI31 subunit
X61587_at	ARHG Ras homolog gene family, member G (rho G)
X07743_at	PLECKSTRIN
AFFX-HUMTFRR/M11507_3_at	AFFX-HUMTFRR/M11507_3_at (endogenous control)
L42572_at	Motor protein
Z69881_at	Adenosine triphosphatase, calcium
M63138_at	CTSD Cathepsin D (lysosomal aspartyl protease)
M28170_at	CD19 CD19 antigen
L41870_at	RB1 Retinoblastoma 1 (including osteosarcoma)
D26156_s_at	Transcriptional activator hSNF2b
M11722_at	Terminal transferase mRNA
U09087_s_at	Thymopoietin beta mRNA
M29540_at	CARCINOEMBRYONIC ANTIGEN PRECURSOR
L47738_at	Inducible protein mRNA
D38073_at	MCM3 Minichromosome maintenance deficient (S. cerevisiae) 3
HG4321-HT4591_at	Ahnak-Related Sequence
U41813_at	HOXA9 Homeo box A9
X85116_rna1_s_at	Epb72 gene exon 1
X58431_rna2_s_at	HOX 2.2 gene extracted from Human Hox2.2 gene for a homeobox protein
M28130_rna1_s_at	Interleukin 8 (IL8) gene
Y00787_s_at	INTERLEUKIN-8 PRECURSOR
U82759_at	GB DEF = Homeodomain protein HoxA9 mRNA
U16954_at	(AF1q) mRNA
Z48501_s_at	GB DEF = Polyadenylate binding protein II
M62762_at	ATP6C Vacuolar H+ ATPase proton channel subunit
M22960_at	PPGB Protective protein for beta-galactosidase (galactosialidosis)
M28209_at	RAS-RELATED PROTEIN RAB-1A
U85767_at	Myeloid progenitor inhibitory factor-1 MPIF-1 mRNA
M13792_at	ADA Adenosine deaminase
L05148_at	Protein tyrosine kinase related mRNA sequence
L08246_at	INDUCED MYELOID LEUKEMIA CELL DIFFERENTIATION PROTEIN MCL1
M19045_f_at	LYZ Lysozyme
M20203_s_at	GB DEF = Neutrophil elastase gene, exon 5
U67963_at	Lysophospholipase homolog (HU-K5) mRNA
J03801_f_at	LYZ Lysozyme
X51521_at	VIL2 Villin 2 (ezrin)
M13452_s_at	LMNA Lamin A
D87076_at	KIAA0239 gene, partial cds
L07648_at	MXI1 mRNA
HG2810-HT2921_at	Homeotic Protein Pl2
L38608_at	ALCAM Activated leucocyte cell adhesion molecule
L28821_at	MANA2 Alpha mannosidase II isozyme
U73960_at	ADP-ribosylation factor-like protein 4 mRNA
M94633_at	GB DEF = Recombination acitivating protein (RAG2) gene, last exon
S50223_at	HKR-T1
Z15115_at	TOP2B Topoisomerase (DNA) II beta (180kD)
U84487_at	CX3C chemokine precursor, mRNA, alternatively spliced
U65928_at	JUN V-jun avian sarcoma virus 17 oncogene homolog
U53468_at	NADH:ubiquinone oxidoreductase subunit B13 (B13) mRNA
U72936_s_at	X-LINKED HELICASE II

Table A.6: 280 genes identified by ERCOF from training samples on ALL-AML leukaemia data set. Probes with bold font were also reported in [41] (continued 2).

Probe	Gene name
X66401_cds1_at	LMP2 gene extracted from H.sapiens genes TAP1, TAP2, LMP2, LMP7 and DOB
X66533_at	GUANYLATE CYCLASE SOLUBLE, BETA-1 CHAIN
AF009426_at	Clone 22 mRNA, alternative splice variant alpha-1
U90546_at	Butyrophilin (BTF4) mRNA
U28833_at	Down syndrome critical region protein (DSCR1) mRNA
M63488_at	RPA1 Replication protein A1 (70kD)
U02493_at	54 kDa protein mRNA
D86479_at	Non-lens beta gamma-crystallin like protein (AIM1) mRNA, partial cds
M31211_s_at	MYL1 Myosin light chain (alkali)
U26266_s_at	DHPS Deoxyhypusine synthase
U05259_rna1_at	MB-1 gene
M58297_at	ZNF42 Zinc finger protein 42 (myeloid-specific retinoic acid-responsive)
D63880_at	KIAA0159 gene
U38846_at	Stimulator of TAR RNA binding (SRB) mRNA
M81695_s_at	ITGAX Integrin, alpha X (antigen CD11C (p150), alpha polypeptide)
D14664_at	KIAA0022 gene
X16546_at	RNS2 Ribonuclease 2 (eosinophil-derived neurotoxin; EDN)
HG627-HT5097_s_at	Rhesus (Rh) Blood Group System Ce-Antigen, Alt. Splice 2, Rhvi
M22324_at	ANPEP Alanyl (membrane) aminopeptidase (aminopeptidase N, aminopeptidase M, microsomal aminopeptidase, CD13)
HG2981-HT3127_s_at	Epican, Alt. Splice 11
Z49194_at	OBF-1 mRNA for octamer binding factor 1
HG1612-HT1612_at	Macmarcks
X77533_at	Activin type II receptor
U20998_at	SRP9 Signal recognition particle 9 kD protein
X17042_at	PRG1 Proteoglycan 1, secretory granule
HG2788-HT2896_at	Calcyclin
HG2855-HT2995_at	Heat Shock Protein, 70 Kda (Gb:Y00371)
U29175_at	Transcriptional activator hSNF2b
J03589_at	UBIQUITIN-LIKE PROTEIN GDX
U41767_s_at	Metargidin precursor mRNA
X06182_s_at	KIT V-kit Hardy-Zuckerman 4 feline sarcoma viral oncogene homolog
M57731_s_at	GRO2 GRO2 oncogene
M24400_at	CTRB1 Chymotrypsinogen B1
M69043_at	MAJOR HISTOCOMPATIBILITY COMPLEX ENHANCER-BINDING PROTEIN MAD3
D43950_at	T-COMPLEX PROTEIN 1, EPSILON SUBUNIT
M19507_at	MPO Myeloperoxidase
M59820_at	CSF3R Colony stimulating factor 3 receptor (granulocyte)
D83785_at	KIAA0200 gene
U50733_at	Dynamitin mRNA
D80001_at	KIAA0179 gene, partial cds
M29696_at	IL7R Interleukin 7 receptor
U72621_at	LOT1 mRNA
M63438_s_at	GLUL Glutamate-ammonia ligase (glutamine synthase)
X62535_at	DAGK1 Diacylglycerol kinase, alpha (80kD)
M84371_rna1_s_at	CD19 gene
L13278_at	CRYZ Crystallin zeta (quinone reductase)
X14850_at	HISTONE H2A.X
J03473_at	ADPRT ADP-ribosyltransferase (NAD+; poly (ADP-ribose) polymerase)
U79274_at	Clone 23733 mRNA
D86983_at	KIAA0230 gene, partial cds
X63469_at	GTF2E2 General transcription factor TFIIE beta subunit, 34 kD
D88270_at	GB DEF = (lambda) DNA for immunoglobin light chain
X59350_at	CD22 CD22 antigen
U35451_at	Heterochromatin protein p25 mRNA
X61970_at	PROTEASOME ZETA CHAIN

Table A.7: 280 genes identified by ERCOF from training samples on ALL-AML leukaemia data set. Probes with bold font were also reported in [41] (continued 3).

Probe	Gene name
U66838_at	Cyclin A1 mRNA
U94836_at	ERPROT 213-21 mRNA
X54326_at	MULTIFUNCTIONAL AMINOACYL-TRNA SYNTHETASE
D55654_at	MDH1 Malate dehydrogenase 1, NAD (soluble)
U31556_at	E2F5 E2F transcription factor 5, p130-binding
X83490_s_at	GB DEF = Fas/Apo-1 (clone pCRTM11-Fasdelta(3,4))
M83667_rna1_s_at	NF-IL6-beta protein mRNA
D38522_at	KIAA0080 gene, partial cds
Z68747_at	GB DEF = Imogen 38
X64072_s_at	SELL Leukocyte adhesion protein beta subunit
M65214_s_at	TCF3 Transcription factor 3 (E2A immunoglobulin enhancer binding factors E12/E47)
M29194_at	LIPC Lipase, hepatic
M86406_at	ACTN2 Actinin alpha 2
U16307_at	Glioma pathogenesis-related protein (GliPR) mRNA
U26173_s_at	BZIP protein NF-IL3A (IL3BP1) mRNA
L11669_at	Tetracycline transporter-like protein mRNA
X15573_at	PFKL Phosphofructokinase (liver type)
X56411_rna1_at	ADH4 gene for class II alcohol dehydrogenase (pi subunit), exon 1
X96752_at	L-3-hydroxyacyl-CoA dehydrogenase
U90552_at	Butyrophilin (BTF5) mRNA
HG4582-HT4987_at	Glucocorticoid Receptor, Beta
AF005043_at	Poly(ADP-ribose) glycohydrolase (hPARG) mRNA
U47077_at	DNA-dependent protein kinase catalytic subunit (DNA-PKcs) mRNA
M83233_at	TCF12 Transcription factor 12 (HTF4, helix-loop-helix transcription factors 4)
X16832_at	CTSH Cathepsin H
D00763_at	GAPD Glyceraldehyde-3-phosphate dehydrogenase
U27460_at	Uridine diphosphoglucose pyrophosphorylase mRNA
X63753_at	SON SON DNA binding protein
Z21507_at	EEF1D Eukaryotic translation elongation factor 1 delta (guanine nucleotide exchange protein)
U57721_at	L-kynurenine hydrolase mRNA
S68134_s_at	GB DEF = CREM=cyclic AMP-responsive element modulator beta isoform [human, mRNA, 1030 nt]
U81556_at	Hypothetical protein A4 mRNA
X97335_at	Kinase A anchor protein
D86967_at	KIAA0212 gene
X66899_at	EWSR1 Ewing sarcoma breakpoint region 1
M37435_at	CSF1 Colony-stimulating factor 1 (M-CSF)
J03798_at	SMALL NUCLEAR RIBONUCLEOPROTEIN SM D1
U30521_at	FRAP FK506 binding protein 12-rapamycin associated protein
U50939_at	Amyloid precursor protein-binding protein 1 mRNA
U83410_at	CUL-2 (cul-2) mRNA
X59543_at	RIBONUCLEOSIDE-DIPHOSPHATE REDUCTASE M1 CHAIN
S71043_rna1_s_at	Ig alpha 2=immunoglobulin A heavy chain allotype 2 constant region, germ line [human, peripheral blood neutrophils, Genomic, 1799 nt]
L49229_f_at	GB DEF = Retinoblastoma susceptibility protein (RB1) gene, with a 3 bp deletion in exon 22 (L11910 bases 161855-162161)
M95678_at	PLCB2 Phospholipase C, beta 2
U49020_cds2_s_at	MEF2A gene (myocyte-specific enhancer factor 2A, C9 form) extracted from Human myocyte-specific enhancer factor 2A (MEF2A) gene, first coding
U00802_s_at	Drebrin E
M93056_at	LEUKOCYTE ELASTASE INHIBITOR
M95178_at	ALPHA-ACTININ 1, CYTOSKELETAL ISOFORM
L25931_s_at	LBR Lamin B receptor
M32304_s_at	TIMP2 Tissue inhibitor of metalloproteinase 2
D38128_at	PTGIR Prostaglandin I2 (prostacyclin) receptor (IP)

Table A.8: 280 genes identified by ERCOF from training samples on ALL-AML leukaemia data set. Probes with bold font were also reported in [41] (continued 4).

Probe	Gene name
D87742_at	KIAA0268 gene, partial cds
M63379_at	CLU Clusterin (complement lysis inhibitor; testosterone-repressed prostate message 2; apolipoprotein J)
X80907_at	GB DEF = P85 beta subunit of phosphatidyl-inositol-3-kinase
AF012024_s_at	Integrin cytoplasmic domain associated protein (Icap-1a) mRNA
J04621_at	SDC2 Syndecan 2 (heparan sulfate proteoglycan 1, cell surface-associated, fibroglycan)
M80899_at	AHNAK AHNAK nucleoprotein (desmoyokin)
U97105_at	Dihydropyrimidinase related protein-2
M30703_s_at	Amphiregulin (AR) gene
U43292_at	MDS1B (MDS1) mRNA
U05572_s_at	MANB Mannosidase alpha-B (lysosomal)
D31887_at	KIAA0062 gene, partial cds
X97748_s_at	GB DEF = PTX3 gene promotor region
Y00339_s_at	CA2 Carbonic anhydrase II
X52056_at	SPI1 Spleen focus forming virus (SFFV) proviral integration oncogene spi1
M92357_at	B94 PROTEIN
AFFX-HUMTFRR/M11507_M_at	AFFX-HUMTFRR/M11507_M_at (endogenous control)
X66610_at	ALPHA ENOLASE, LUNG SPECIFIC
U07139_at	CAB3b mRNA for calcium channel beta3 subunit
HG4535-HT4940_s_at	Dematin
X64364_at	BSG Basigin
HG3162-HT3339_at	Transcription Factor Iia
X51420_at	TYRP1 Tyrosinase-related protein 1
D50918_at	KIAA0128 gene, partial cds
AJ000480_at	GB DEF = C8FW phosphoprotein
J04027_at	Adenosine triphosphatase mRNA
S76638_at	NFKB2 Nuclear factor of kappa light polypeptide gene enhancer in B-cells 2 (p49/p100)
U28042_at	DEAD box RNA helicase-like protein mRNA
M11147_at	FTL Ferritin, light polypeptide
HG4755-HT5203_s_at	Spinal Muscular Atrophy 4
X65644_at	IMMUNODEFICIENCY VIRUS TYPE I ENHANCER-BINDING PROTEIN 2
D26579_at	Transmembrane protein
U88964_at	HEM45 mRNA
U07132_at	Orphan receptor mRNA, partial cds
L20941_at	FTH1 Ferritin heavy chain
M83221_at	TRANSCRIPTION FACTOR RELB
L09235_at	ATP6A1 ATPase, H+ transporting, lysosomal (vacuolar proton pump), alpha polypeptide, 70kD, isoform 1
Z32765_at	GB DEF = CD36 gene exon 15
M57710_at	LGALS3 Lectin, galactoside-binding, soluble, 3 (galectin 3)
L22075_at	Guanine nucleotide regulatory protein (G13) mRNA
K03195_at	(HepG2) glucose transporter gene mRNA
M21119_s_at	LYZ Lysozyme
U61836_at	Putative cyclin G1 interacting protein mRNA, partial sequence
U77396_at	No cluster in current Unigene and no Genbank entry for U77396 (qualifier U77396_at)
L41067_at	Transcription factor NFATx mRNA
L33930_s_at	CD24 signal transducer mRNA and 3' region
M22898_at	TP53 Tumor protein p53 (Li-Fraumeni syndrome)
M92439_at	130 KD LEUCINE-RICH PROTEIN
M61853_at	CYP2C18 Cytochrome P450, subfamily IIC (mephenytoin 4-hydroxylase), polypeptide 18
X66171_at	CMRF35 mRNA
AF015913_at	GB DEF = SKB1Hs mRNA

Table A.9: 280 genes identified by ERCOF from training samples on ALL-AML leukaemia data set. Probes with bold font were also reported in [41] (continued 5).

Probe	Gene name
U50928_at	PKD2 Autosomal dominant polycystic kidney disease type II
D63874_at	HMG1 High-mobility group (nonhistone chromosomal) protein 1
X82240_rna1_at	TCL1 gene (T cell leukemia) extracted from H.sapiens mRNA for Tcell leukemia/lymphoma 1
U79285_at	GLYCYLPEPTIDE N-TETRADECANOYLTRANSFERASE
U21858_at	HISTONE H3.3
L76702_at	Protein phosphatase 2A 74 kDa regulatory subunit (delta or B”” subunit)
M19888_at	SPRR1B Small proline-rich protein 1B (cornifin)
U31814_at	Transcriptional regulator homolog RPD3 mRNA
X77307_at	5-HYDROXYTRYPTAMINE 2B RECEPTOR
U49844_at	Protein kinase ATR mRNA
U65410_at	Mitotic feedback control protein Madp2 homolog mRNA
D14658_at	KIAA0102 gene
Y07604_at	Nucleoside-diphosphate kinase
M60527_at	DCK Deoxycytidine kinase
X58072_at	GATA3 GATA-binding protein 3

Table A.10: Thirty-seven genes selected by ERCOF on training samples and reported in [140] to separate TEL-AML1 from other subtypes of ALL cases in pediatric ALL study. All these genes are relatively highly expressed (above the mean value aross all the samples) in TEL-AML1 samples.

Probe	Accession No.	Description
34481_at	AF030227	vav proto-oncogene, exon 27
36239_at	Z49194	H.sapiens mRNA for oct-binding factor
37470_at	AF013249	Homo sapiens leukocyte-associated Ig-like receptor-1 (LAIR-1) mRNA
38203_at	U69883	Human calcium-activated potassium channel hSK1 (SK) mRNA
38570_at	X03066	Human mRNA for HLA-D class II antigen DO beta chain
38578_at	M63928	Homo sapiens T cell activation antigen (CD27) mRNA
38906_at	M61877	Human erythroid alpha-spectrin (SPTA1) mRNA
40745_at	L13939	Homo sapiens beta adaptin (BAM22) mRNA
41381_at	AB002306	Human mRNA for KIAA0308 gene
41442_at	AB010419	Homo sapiens mRNA for MTG8-related protein MTG16a
31898_at	D86967	Human mRNA for KIAA0212 gene
32660_at	AB002340	Human mRNA for KIAA0342 gene
34194_at	AL049313	Homo sapiens mRNA; cDNA DKFZp564B076 (from clone DKFZp564B076)
35614_at	AB012124	Homo sapiens TCFL5 mRNA for transcription factor-like 5
35665_at	Z46973	H.sapiens mRNA for phosphatidylinositol 3-kinase
36524_at	AB029035	Homo sapiens mRNA for KIAA1112 protein
36537_at	AB011093	Homo sapiens mRNA for KIAA0521 protein
37280_at	U59912	Human chromosome 4 Mad homolog Smad1 mRNA
41200_at	Z22555	H.sapiens encoding CLA-1 mRNA
32224_at	AB018312	Homo sapiens mRNA for KIAA0769 protein
36985_at	X17025	Human homolog of yeast IPP isomerase
38124_at	X55110	Human mRNA for neurite outgrowth-promoting protein
40570_at	AF032885	Homo sapiens forkhead protein (FKHR) mRNA
41498_at	AB020718	Homo sapiens mRNA for KIAA0911 protein
41814_at	M29877	Human alpha-L-fucosidase
32579_at	U29175	Human transcriptional activator (BRG1) mRNA
33162_at	X02160	Human mRNA for insulin receptor precursor
1779_s_at	M16750	Human pim-1 oncogene mRNA
1488_at	L77886	Human protein tyrosine phosphatase mRNA
1336_s_at	X06318	Human mRNA for protein kinase C (PKC) type beta I
1299_at	X93512	H.sapiens mRNA for telomeric DNA binding protein (orf2)
1217_g_at	X07109	Human mRNA for protein kinase C (PKC) type beta II
932_i_at	L11672	Human Kruppel related zinc finger protein (HTF10) mRNA
880_at	M34539	Human FK506-binding protein (FKBP) mRNA
755_at	D26070	Human mRNA for type 1 inositol 1,4,5-trisphosphate receptor
577_at	M94250	Human retinoic acid inducible factor (MK) gene exons 1-5
160029_at	X07109	protein kinase C beta 1

Table A.11: Top 20 genes selected by entropy measure on training samples to separate MLL from other subtypes of ALL cases in pediatric ALL study. The last column indicates the sample class in which the gene is highly expressed (above the mean value aross all the samples).

Probe	Accession No.	Description	HighlyExp
34306_at	AB007888	Homo sapiens KIAA0428 mRNA	MLL
36777_at	AJ001687	Homo sapiens NKG2D gene, exons 2-5 and joined mRNA and CDS	MLL
33412_at	AI535946	vicpro2.D07.r Homo sapiens cDNA, 5' end	MLL
657_at	L11373	Human protocadherin 43 mRNA, complete cds for abbreviated PC43	MLL
32207_at	M64925	Human palmitoylated erythrocyte membrane protein (MPP1) mRNA	OTHERS
33847_s_at	AI304854	Homo sapiens cDNA, 3' end	MLL
34337_s_at	AJ010014	Homo sapiens mRNA for M96A protein	OTHERS
1389_at	J03779	Human common acute lymphoblastic leukemia antigen (CALLA) mRNA	OTHERS
34861_at	D63997	Homo sapiens mRNA for GCP170	OTHERS
40518_at	Y00062	Human mRNA for T200 leukocyte common antigen (CD45, LC-A)	MLL
40913_at	W28589	Homo sapiens cDNA	OTHERS
31898_at	D86967	Human mRNA for KIAA0212 gene	OTHERS
38413_at	D15057	Human mRNA for DAD-1	MLL
2062_at	L19182	Human MAC25 mRNA	MLL
794_at	X62055	H.sapiens PTP1C mRNA for protein-tyrosine phosphatase 1C	MLL
40519_at	Y00638	Human mRNA for leukocyte common antigen (T200)	MLL
41747_s_at	U49020	Human myocyte-specific enhancer factor 2A (MEF2A) gene	MLL
38160_at	AF011333	Homo sapiens DEC-205 mRNA	MLL
37692_at	AI557240	Homo sapiens cDNA, 3' end	MLL
40797_at	AF009615	Homo sapiens ADAM10 (ADAM10) mRNA	MLL

Table A.12: Twenty-four genes selected by ERCOF on training samples and reported in [140] to separate MLL from other subtypes of ALL cases in pediatric ALL study. All these genes are relatively highly expressed (above the mean value aross all the samples) in MLL samples except U70321 (accession number). Genes with bold font are among top 20 features selected by entropy measure and can be found in Table A.11 as well.

Probe	Accession No.	Description
36777_at	AJ001687	Homo sapiens NKG2D gene, exons 2-5 and joined mRNA and CDS
39424_at	U70321	Human herpesvirus entry mediator mRNA
40076_at	AF004430	Homo sapiens hD54+ins2 isoform (hD54) mRNA
40493_at	L05424	Human hyaluronate receptor (CD44) gene
40506_s_at	U75686	Homo sapiens polyadenylate binding protein mRNA
40763_at	U85707	Human leukemogenic homolog protein (MEIS1) mRNA
40797_at	AF009615	Homo sapiens ADAM10 (ADAM10) mRNA
40798_s_at	Z48579	H.sapiens mRNA for disintegrin-metalloprotease (partial)
41747_s_at	U49020	Human myocyte-specific enhancer factor 2A (MEF2A) gene, first coding
32193_at	AF030339	Homo sapiens receptor for viral semaphorin protein (VESPR) mRNA
32215_i_at	AB020685	Homo sapiens mRNA for KIAA0878 protein
33412_at	AI535946	Homo sapiens cDNA, 5' end
34306_at	AB007888	Homo sapiens KIAA0428 mRNA
34785_at	AB028948	Homo sapiens mRNA for KIAA1025 protein
35298_at	U54558	Homo sapiens translation initiation factor eIF3 p66 subunit mRNA
37675_at	X60036	H.sapiens mRNA for mitochondrial phosphate carrier protein
38391_at	M94345	Homo sapiens macrophage capping protein mRNA
38413_at	D15057	Human mRNA for DAD-1
2062_at	L19182	Human MAC25 mRNA
2036_s_at	M59040	Human cell adhesion molecule (CD44) mRNA
1914_at	U66838	Human cyclin A1 mRNA
1126_s_at	L05424	Human cell surface glycoprotein CD44 (CD44) gene, 3' end of long tailed isoform
1102_s_at	M10901	Human glucocorticoid receptor alpha mRNA
657_at	L11373	Human protocadherin 43 mRNA, complete cds for abbreviated PC43

Table A.13: Nineteen genes selected by ERCOF on training samples and reported in [140] to separate Hyperdip>50 from other subtypes of ALL cases in pediatric ALL study. All these genes are relatively highly expressed (above the mean value aross all the samples) in Hyperdip>50 samples.

Probe	Accession No.	Description
38518_at	Y18004	Homo sapiens mRNA for SCML2 protein
39628_at	AI671547	Homo sapiens cDNA, 3' end
31863_at	D80001	Human mRNA for KIAA0179 gene
37543_at	D25304	Human mRNA for KIAA0006 gene
38968_at	AB005047	Homo sapiens mRNA for SH3 binding protein
39039_s_at	AI557497	Homo sapiens cDNA, 3' end
39329_at	X15804	Human mRNA for alpha-actinin
39389_at	M38690	Human CD9 antigen mRNA
32207_at	M64925	Human palmitoylated erythrocyte membrane protein (MPP1) mRNA
32236_at	AF032456	Homo sapiens ubiquitin conjugating enzyme G2 (UBE2G2) mRNA
32251_at	AA149307	Homo sapiens cDNA, 3' end
36620_at	X02317	Human mRNA for Cu/Zn superoxide dismutase (SOD)
36937_s_at	U90878	Homo sapiens carboxyl terminal LIM domain protein (CLIM1) mRNA
37350_at	AL031177	26S Proteasome subunit p28 (Ankyrin repeat protein)) (isoform 1)
38738_at	X99584	H.sapiens mRNA for SMT3A protein
39168_at	AB018328	Homo sapiens mRNA for KIAA0785 protein
40903_at	AL049929	Homo sapiens mRNA; cDNA DKFZp547O0510 (from clone DKFZp547O0510)
32572_at	X98296	H.sapiens mRNA for ubiquitin hydrolase
306_s_at	J02621	Human non-histone chromosomal protein HMG-14 mRNA

Appendix B

Some Resources

B.1 Kent Ridge Biomedical Data Set Repository

All the gene expression profiles and proteomic data described in Chapter 5, and some DNA sequences used in Chapter 7 can be found in the *Kent Ridge Biomedical Data Set Repository* at `http://sdmc.i2r.a-star.edu.sg/rp/`. In this data repository, we have collected gene expression data, protein profiling data and genomic sequence data that are related to classification and are published recently in *Science*, *Nature* and other prestigious journals. As the file formats of these original raw data are different from common ones used in most of machine learning softwares, we have transformed these data sets into the standard *.data* and *.names* format and stored them in this repository. Besides, we also provide data in *.arff* format which is used by *Weka*, a machine learning software package developed at University of Waikato in New Zealand. Detailed information of Weka can be found at `http://www.cs.waikato.ac.nz/~ml/weka/`.

B.2 DNAFSMiner

The *DNAFSMiner* (DNA Functional Site Miner) is a web-based toolbox for recognition of functional sites in DNA sequences. It was built on the technologies presented in Chapter 7 and written in Java and Perl languages. It can be accessed via `http://sdmc.i2r.a-star.edu.sg/DNAFSMiner/`. Currently, it can be used to identify translation initiation site (*TISMiner*) in vertebrate mRNA, cDNA, and DNA sequences and polyadenylation signal (*Poly(A) Signal Miner*) in human sequences.