University of Alberta

RANK-1 BICLUSTER CLASSIFIER

by

Nasimeh Asgarian

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the degree of **Master of Science**.

Department of Computing Science

Edmonton, Alberta

Spring 2007

Rank-1 Bicluster Classifier

Nasimeh Asgarian

University of Alberta

Library Release Form

Name of Author: Nasimeh Asgarian

Title of Thesis: Rank-1 Bicluster Classifier

Degree: Master of Science

Year this Degree Granted: 2007

____

1209 Latta Crescent
Edmonton, Alberta
Canada, T6R 3N2

Date: _______________

University of Alberta

RANK-1 BICLUSTER CLASSIFIER

by

Nasimeh Asgarian

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the degree of **Master of Science**.

Department of Computing Science

Edmonton, Alberta

Spring 2007

University of Alberta

Faculty of Graduate Studies and Research

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research for acceptance, a thesis entitled **Rank-1 Bicluster Classifier** submitted by **Nasimeh Asgarian** in partial fulfillment of the requirements for the degree of **Master of Science**.

____

Russell Greiner

____

David Wishart

____

Guohui Lin

Date: ____________________

Abstract

A DNA-microarray measures the gene expression levels of thousands of genes under different experimental conditions (samples). These values describe the unique properties to each cell type. The samples may have come from different time points, diseased or healthy tissues, or different individuals. This technology benefits biological research greatly in understanding of biological processes.

Our goal is to learn a microarray classifier to distinguish members of various classes, based on their expression levels.

In this thesis we propose a method for sample classification by reducing the dimensionality of the data, using *bi-clusters*. A *bi-cluster* is a subset of genes and a subset of samples that have similar patterns based on the gene expression values. We also propose a novel algorithm for finding bi-clusters from the microarray data, using the best rank-1 matrix approximation. We demonstrate that our method works effectively by comparing its prediction accuracy with other classifiers, including another bi-clustering algorithm.

Acknowledgements

First, I would like to thank my supervisor, Dr. Russell Greiner, for being an enthusiastic and genius supervisor. He introduced me to the microarray data and biclustering methods, and provided the idea underlying the main result of this thesis. This research would have been impossible without his helps and encouragements.

Many thanks to David Wishart and Guohui Lin for a very careful reading of the thesis, and the helpful comments that improved the thesis. Also, I would like to thank the staff and faculty of the Department of Computing Science at the University of Alberta for providing such a nice academic environment.

My friends at the University of Alberta played a big role in making my studies more enjoyable. Especially, I am grateful to Leila Homaeian, Reza Sherkat, Parastoo Orouji, Alireza Sameny, and Davood Rafiei. Also, I would like to thank Kathryn Graham, Ali Ghodsi, Dale Schuurmans, Mohammad Ghavamzadeh, and Volodymyr Mnih for the helpful discussions that we had. I thank the staff at the Alberta Ingenuity Center for Machine Learning, especially David Woloschuk.

Finally, I would like to thank my family. I owe so much to my dear parents, Javad Asgarian and Zinat Hosseini. They encouraged me to learn and supported me throughout my life. I always feel the warmth of their love, even now that we are thousands of kilometers apart. Also, many thanks to Naghmeh for being such a wonderful sister. Last but not least, I wish to thank my husband, Mohammadreza, whose patience, love, and support made my studies both possible and enjoyable. This thesis is dedicated to him.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

Biologists are using DNA-microarrays to measure the gene expression level of biological samples. Tens of thousands of genes are measured on a very small set, usually tens to a few hundred, of samples. The *samples* may correspond to different time points, different experimental conditions, different organs, diseased or healthy tissues, or different individuals. This data is typically stored in a data matrix of real numbers. Biologists analyze this data seeking patterns in the gene expression levels in order to understand the expression level of the genes under different conditions. This can help them to determine the genes involved in a disease, suggest biomarkers of a specific disease, propose targets for drug intervention, and use microarray as a screening tool.

Working with microarray data has several challenges. The most important one is the size of the matrix. There are very few samples with too many features, typically less than a hundred samples versus more than $50,000$ genes. Dealing with outliers and missing values are other challenges.

Biologists are mainly interested in the following tasks [3]:

- Finding patterns in data (clustering) such as:

    - Finding clusters of genes according to their expression values over different samples, and
    - Finding clusters of samples based on the expression values of genes.

- Classification

    - Based on the expression values of other genes with a known class (based on the functionality of genes), classify a new gene based on its expression value, and
    - Classify a new sample with given expression value of the genes for that sample, where the sample class depends on the study. The class can be different time points, cancer versus normal, clinical outcome, etc.

Applying known clustering algorithms such as k-means clustering and hierarchical clustering to gene expression data can run into a significant difficulty. These algorithms require the genes in the same cluster to behave similarly over *all* experimental conditions. But many gene activation patterns are common to a group of genes only under *specific experimental conditions*. In fact, based on the general understanding of cellular processes, we expect subsets of genes to be co-regulated and co-expressed only under certain experimental conditions, but to behave almost independently under other conditions [3]. ”Discovering such local expression patterns may be the key to uncovering many genetic pathways that are not apparent otherwise. It is therefore highly desirable to move beyond the clustering diagram, and to develop algorithmic approaches capable of discovering local patterns in the microarray data.” [3]

Clustering methods can be applied to either the genes or the samples in the data matrix, separately. Instead, *bi-clustering* algorithms perform clustering of genes and samples at the same time. They identify subsets of genes that have similar activity patterns under a specific subset of samples (finding a local model in the data). Therefore, bi-clustering approaches are an ideal technique to use when the following situation applies [3]:

1. Only a small set of genes participates in a cellular process of interest.

2. An interesting cellular process is active only in a subset of the conditions.

3. A single gene may participate in multiple pathways that may or may not be co-active under all conditions.

In this thesis we propose:

- An algorithm for finding bi-clusters from microarray data, based on the best rank-1 matrix approximation, and

- A method for sample classification based on these bi-clusters. We call our method RoBicC (Rank-one Bi-cluster Classifier).

We also compare the performance RoBicC with other classifiers, including ones that use other bi-clustering algorithm. We do this by replacing the bi-clusters found by our method with those found by other methods. Then using our sample classification method, we classify the samples and compare the prediction accuracies.

Thesis statement: In this thesis, we demonstrate that our RoBicC system (classifier based on rank-1 bi-clusters) can effectively classify sample in microarray data.

1.2 Overview

The rest of Chapter 1 provides background information on the technology used in microarray gene expression studies, as well as bi-clustering methods and classification tasks. Chapter 2 summarizes previous work that have been described for bi-clustering and classifying microarrays. Chapter 3 describes the bi-clustering problem and the structure of our algorithm for finding bi-clusters. Chapter 4 explains how we use bi-clusters to classify microarray samples. Chapter 5 describes some of the publicly available microarray data, evaluates our bi-clustering and classification method, and compares our results with the known results for the datasets. Finally, Chapter 6 draws conclusions and describes some future work.

1.3 Introduction to Microarrays

In the past few years, there has been an explosion in the development of high-throughput methods for molecular biology experiments. The appearance of the DNA microarray technology is an example of such a high-throughput methods. Microarray technology allows the study of tens of thousands of genes at the same time, under different experimental conditions. This technique is proving to be a great help for biological research and in the understanding of gene regulation and gene interactions. These results also have very important applications in pharmaceutical and clinical research [8].

However, it is not easy to handle the huge amount of data generated by microarray experiments. We need to use advanced data processing techniques to handle these types of data. There are now several machine learning methods to deal with microarray data, depending on the task [18], [19], [21], [22], [23].

One of the tasks is finding useful patterns from microarray data, e.g. find a group of samples that has similar correlation with respect to a set of genes (biclustering). We are also interested in classification tasks. For instance, based on the microarray gene expression data, predict if a patient has cancer or not, or if a patient will respond to a certain type of treatment or not?

1.3.1 Biology Background

A *cell* is the fundamental unit of life as it contains all the structures and molecular constituents needed for life. Almost every cell of the body contains a full set of genes. *DNA* carries the genetic information of a cell. A DNA molecule -called a chromosome- consists of thousands of genes. Each *gene* contains structural information about protein sequence and regulatory information about protein expression,as shown in Figure 1.1. Each gene codes for a protein, but in any cell only some of these genes are expressed, and it is this expressed subset that confers unique properties to each cell type [1].

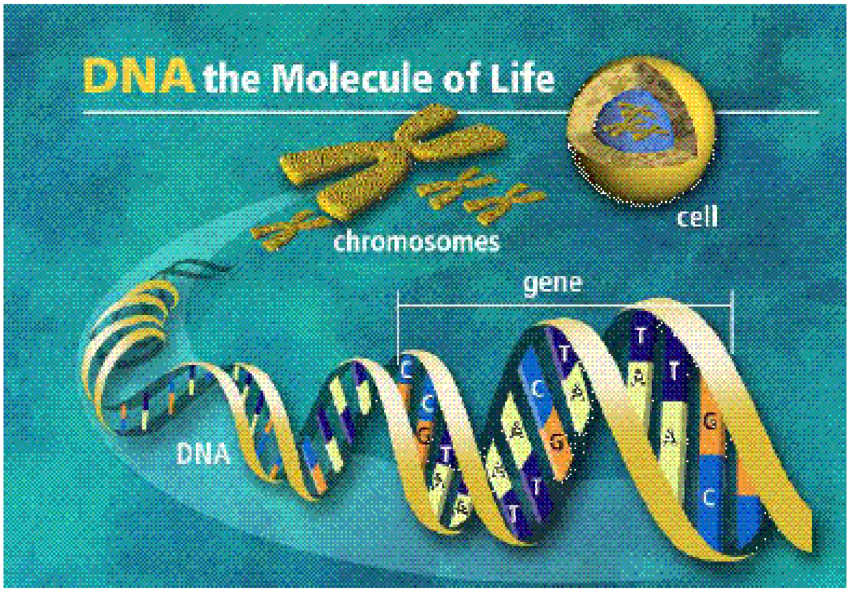


Figure 1.1: Cell, Chromosome, and DNA [29].

Proteins are sequences of amino-acids. Each protein has a unique sequence of amino-acids, and this sequence specifies the shape and function of the protein. Proteins perform most of the critical functions of cells.

Intermediate between a gene and its corresponding protein is messenger RNA (mRNA). When a gene is activated, cellular machinery begins to copy certain segments of that gene. The resulting product is known as *messenger RNA (mRNA)*, which is the body's template for creating proteins. To produce a protein from DNA, as shown in Figure 1.2, DNA is first transcribed to mRNA, which is then translated to protein.

"Gene expression" is the term used to describe the transcription of the information contained within the DNA, into messenger RNA (mRNA) molecules that are then translated into the proteins [1]. Gene expression is the process by which a gene's DNA sequence is converted into the structures and functions of a cell. Transcription is the process through which the genetic information from DNA is converted into RNA. Translation is the process through which messenger RNA (mRNA) molecules are converted into the proteins. Microarrays are designed to measure the concentration of mRNA.

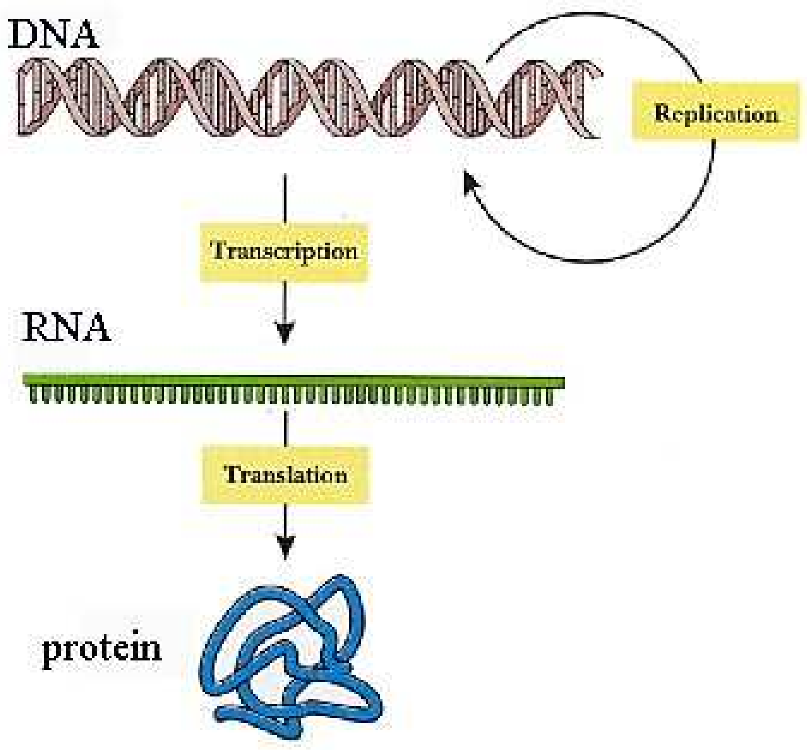


Figure 1.2: The route from the DNA code to the protein. “The Fundamental Paradigm” [30].

1.3.2 Microarray Technology

A DNA-microarray is a small, solid support (usually a glass microscope slide) onto which the sequences from thousands of different genes are immobilized, or attached, at fixed locations. Figure 1.3 shows a microarray slide. The DNA is printed, spotted, or actually synthesized directly onto the support [1].

There are different ways that microarrays can be used to measure gene expression levels. One of the most popular microarray applications is comparing gene expression levels under two different conditions, e.g., the same cell type in a healthy versus a diseased state. In this type of study, the total mRNA from the cells in two different conditions is extracted and labeled with two different fluorescent labels (for example, green dye for healthy and red dye for diseased state). Both extracts are washed over the microarray. This is called a 2-color microarray.

The dyes enable the amount of sample bound to a spot to be measured by the level of fluorescence emitted when it is excited by a laser. Thus, from the fluores-

Figure 1.3: Microarray Slide (1-color) [31].

cence intensities and colors for each spot, the relative expression levels of the genes in both samples can be estimated. Here is the list of possible colors in a 2-color microarray [1]:

- *Green:* represents Control DNA, where DNA derived from normal tissue is hybridized to the target DNA.
- *Red:* represents Sample DNA, where DNA is derived from diseased tissue hybridized to the target DNA.
- *Yellow:* represents a combination of Control and Sample DNA, where both hybridized equally to the target DNA.
- *Black:* represents areas where neither the Control nor Sample DNA hybridized to the target DNA.

After the hybridization step is complete, the microarray will be placed on a scanner that uses a specific frequency of light from a laser. This will produce an image from the scanned array. To get information about gene expression levels, this image is analyzed using image analysis software. Each spot on the array is identified, its intensity is measured and compared to the background. To obtain the final gene expression matrix from spot quantizations, all the quantities related to some gene have to be combined and the entire matrix has to be scaled to make different arrays comparable. This process is called "normalization". Figure 1.4 shows these steps.

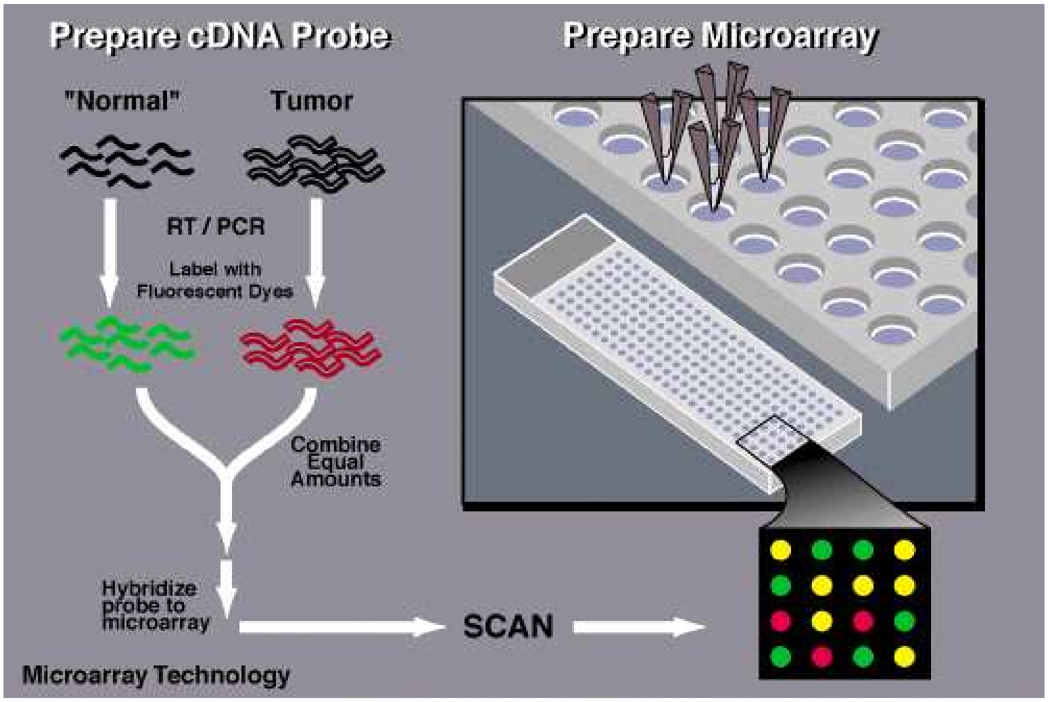


Figure 1.4: Microarray Technology [2].

The data from one sample corresponds to a $n \times 1$ vector of real numbers, where n can be tens of thousands. The data generated by a set of microarray experiments can be viewed as a matrix of expression levels, organized by genes versus samples. If we have p samples, this produces a $n \times p$ matrix of real numbers. Each column of the matrix contains the expression levels of the n genes monitored in the microarray, and each row contains the the expression levels of a gene as it varies over the p samples. The next task is analyzing the data generated by a microarray.

1.4 Bi-clustering

One of the objectives in analyzing microarray data is finding patterns in the data. These patterns can be found with respect to the genes in the matrix or with respect to the samples (or different conditions) in the matrix. Clustering methods can be used to find the clusters of genes or clusters of samples in the data, separately.

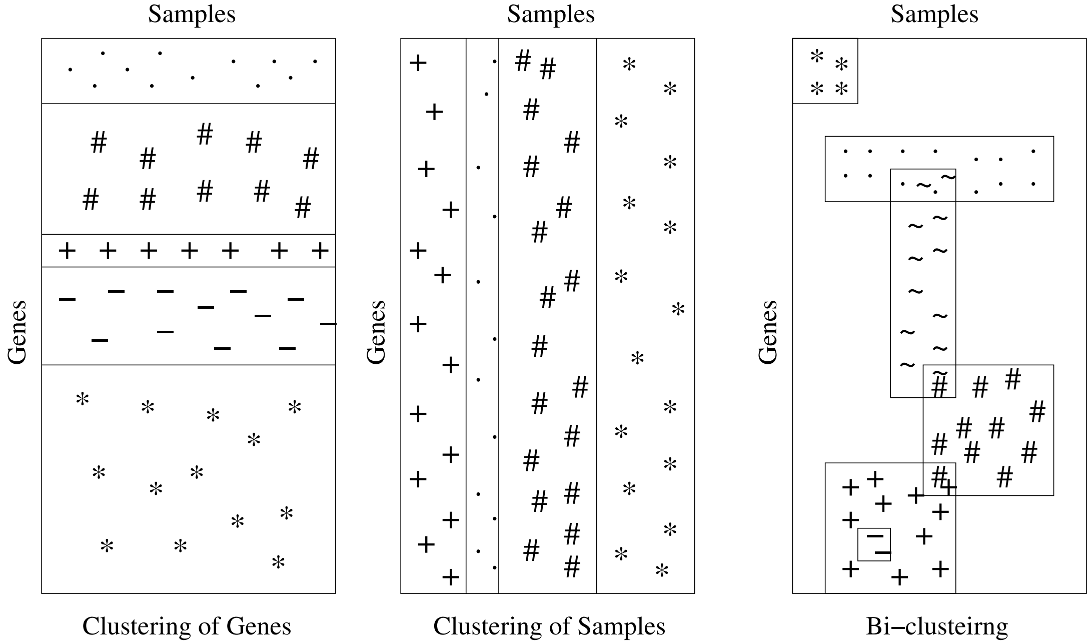


Figure 1.5: Clustering versus Bi-clustering.

Traditional clustering methods, such as hierarchical and K-means approaches, consider the similarity of genes over all samples. They either find the clusters of genes with similar patterns over *all* samples, or they find the clusters of samples with similar pattern over *all* genes. But many activation patterns are common to a group of genes only under specific experimental conditions. In fact, based on the general understanding of cellular processes, it is expected that subsets of genes will be co-regulated and co-expressed only under certain experimental conditions, but to behave almost independently under other conditions [3].

Bi-clustering methods perform the clustering on genes and samples at the same time. They find a *subset* of genes that have similar pattern under a specific *subset* of samples.Therefore, clustering algorithms find a *global model* while bi-clustering algorithms find a *local model* in the data [3]. Discovering such local expression patterns may be the key to uncovering many genetic pathways that are not visible otherwise. It is therefore highly desirable to move beyond the clustering methods, and to develop algorithmic approaches capable of discovering local patterns in the microarray data; see Figure 1.5.

Bi-clustering algorithms can identify subsets of samples or experimental conditions that have similar pattern over a subsets of genes. It might be the case that some genes show different patterns for a group of samples (some genes may have two or more functions). It means that bi-clusters may not be disjoint, i.e. they might overlap. Clustering, in its simple form, partitions the genes into disjoint gene sets. On the other hand, there might be some genes or samples that do not show any pattern for any samples or genes. Therefore, each gene and each sample should be able to be in more than one cluster or no cluster at all.

Therefore, bi-clustering approaches should find groups of genes and samples according to the following restrictions [3]:

1. A cluster of genes should be defined with respect to only a subset of samples.
2. A cluster of samples should be defined with respect to only a subset of genes.
3. A gene or sample should be able to belong to more than one cluster or no cluster at all.

Table 1.1 summarizes the differences between the classical clustering algorithms and bi-clustering algorithms.

Clustering Algorithms	Bi-clustering Algorithms
Find clusters of genes (or clusters of samples).	Find clusters of genes and clusters of samples *simultaneously*.
Find the clusters of genes with similar patterns over *all* samples.	Find the clusters of genes with similar patterns over a *subset* of samples.
Find the clusters of samples with similar patterns over *all* genes.	Find the clusters of samples with similar patterns over a *subset* of genes.
Find a *global model*.	Find a *local model*.

Table 1.1: Comparing clustering algorithms and bi-clustering algorithms.

Another characteristic that distinguishes different approaches to cluster analysis is flat like K-means approaches versus hierarchical approaches [5].

The K-means algorithm process is as follows:

- The dataset is partitioned into K clusters and the data points are randomly assigned to the clusters resulting in clusters that have roughly the same number of data points.
- For each data point, calculate the distance from the data point to each cluster, based on a pre-defined distance function (e.g., euclidean distance). If the data point is closest to its own cluster, leave it where it is. If the data point is not closest to its own cluster, move it into the closest cluster.
- Repeat the above step until a complete pass through all the data points results in no data point moving from one cluster to another. At this point the clusters are stable and the clustering process ends.

K-Means clustering generates a specific number of disjoint, flat (non-hierarchical) clusters. In this type of clusters, there are always K clusters, and there is always at least one item in each cluster. The clusters are non-hierarchical and they do not overlap.

A bottom-up hierarchical algorithm process is as follows (a top-down clustering method works in a similar way but in the opposite direction):

- Assign each object to a separate cluster.
- Evaluate all pair-wise distances between clusters, where distance is a predefined function.
- Construct a distance matrix using the distance values between clusters.
- Look for the pair of clusters with the shortest distance.
- Remove the pair from the matrix and merge them.
- Evaluate all distances from this new cluster to all other clusters, and update the matrix.

- Repeat until the distance matrix is reduced to a single element.

Hierarchical clustering (bottom-up method) generates clusters that have sub-clusters, which in turn have sub-clusters, and so on. Gene expression data might also have this hierarchical quality. Bottom-up hierarchical clustering starts with every single object (gene or sample) in a single cluster. Then, in each successive iteration, it merges the closest pair of clusters by satisfying some similarity criteria, until all of the data is in one cluster.

With a large number of variables, K-means may be computationally faster than hierarchical clustering (if K is small). But the number of clusters, K, is fixed in this approach and it is difficult to predict what K should be. In addition, different numbers of clusters and initial partitions can result in different final clusters.

The advantage of using hierarchical clustering is that it can produce an ordering of the objects, which may be informative for data display. They also generate smaller clusters that may be helpful for discovery. The disadvantage is that it is not possible to relocate the objects that may have been *incorrectly* grouped at an early stage. Also, the use of different distance metrics for measuring distances between clusters may generate different results.

1.5 Classification

Bi-clustering algorithms provide a method for finding subgroups of genes and samples that have similar patterns under a certain experimental conditions. After clustering, the next step is to find the relationship within genes and samples or between them (supervised learning task). Classification deals with the problem of predicting class membership of unlabeled test data points after learning from a training set of data points with known class memberships. There are several classification tasks involving microarray data that biologists are interested in, depending on the studies. Here is a summary of the major classification tasks:

- **Gene Classification**: Using the expression values of a set of genes belonging to known classes (based on the functionality of genes), predict the class of a new gene, with known expression value.

- **Gene Interactions**: It is known that the expression level of gene g is regulated by a set of other genes, G. Using the expression values of the genes in the set G, predict the expression value of g.

- **Sample Classification**: Using the expression values of a set of samples with known classes, predict the class of a new sample based on its expression values.

- **Time Series:** Gene expression levels change over time, as proteins regulate gene transcription, due to a change in the state of a disease, or due to the effect of a certain treatment. Given expression levels of genes from different time points $\{t_1, \ldots, t_n\}$, predict the time point for a new gene given its expression value.

This thesis focuses on the third task: sample (patient) classification. Sample classification requires learning a classifier from the data for different samples (with known class) and their gene expression levels. The learned model is then used to predict the class of a new sample. Predicting the class of a patient includes:

1. **Cancer versus Normal** (Diagnosis): Having expression values of a set of patients, some of whom have cancer and some of whom do not have cancer, build a classifier based on this information in order to predict if a new patient has cancer or not.

2. **Type of cancer** (Diagnosis): Having expression values of a set of patients, all of whom have the same cancer (for example, human acute leukemia) with different types (acute myeloid leukemia, AML, versus acute lymphoblastic leukemia, ALL), predict the type of cancer for a new case.

3. **Clinical outcome** (Prognosis): Like the previous studies, in this task expression values of a set of patients, all of whom have the same cancer, is given. They will all go under the same treatment and after a certain amount of time they will be checked to determine if they have responded to the treatment or

not. The class label here is whether their cancer recurred after a certain treatment or not. Therefore, the task here is to build a classifier in order to predict if a patient responds to a certain treatment or not.

For this task, a set of microarray experiments, each corresponding to mRNA from a different patient, is given. The mRNA is taken from the same cell type from each patient. Since this is supervised learning, the class labels for these patients are known as well. We want to use this set of patients (their expression values and their classes) to produce a classifier that can be used to accurately predict the class of a new patient, given his or her gene expression values. Using all the genes in order to build the classifier and predict the patient's class often leads to over-fitting the data. Here *Over-fitting* means that the prediction accuracy on training data increases as the number of features increases while the prediction accuracy on test data decreases (Figure 1.6). This often occurs when the number of features (i.e. genes) is large with respect to the number of training cases (samples), especially when the features are highly correlated with each other, or if many features are irrelevant.

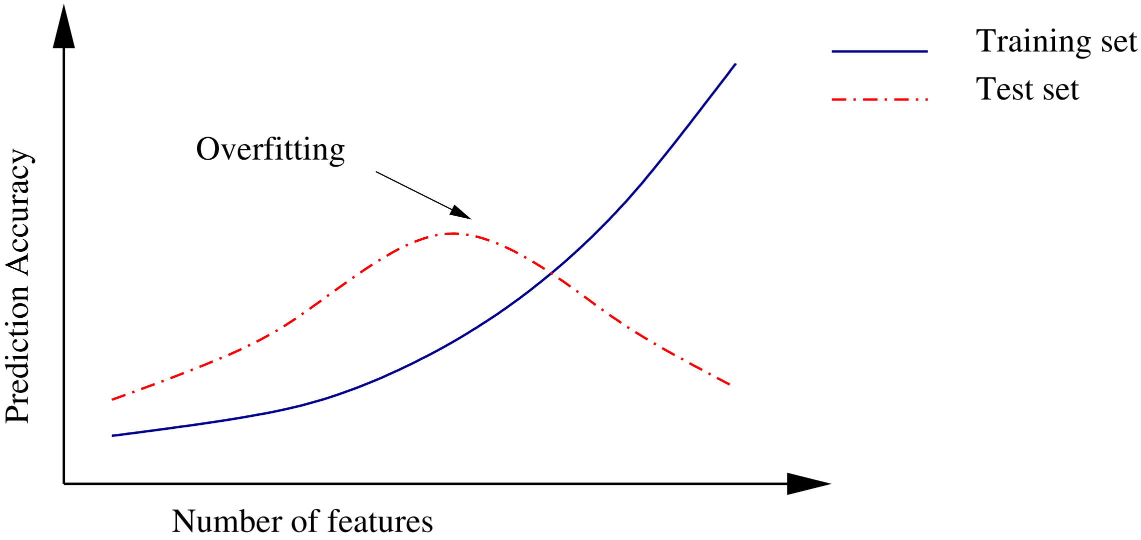


Figure 1.6: Over-fitting.

The best way to avoid over-fitting is to use lots of training instances. Unfortunately, in microarray data, there are very few. Usually only tens to a few hundred of samples are available with each sample having lots of features (tens of thousands of genes). Therefore, there are several algorithms for finding a smaller set of genes in order to avoid over-fitting the data.

The first task (cancer versus normal) is relatively easy to predict accurately, because many genes change in cancer cells. Since the expression levels for thousands of genes are known, it is possible to find a subset of genes that changes in their expression levels lead to accurately predicting the class of patients. The challenges here include the quality of mRNA, and the noise in the data. For these studies, support vector machines usually give high prediction accuracies. Similarly, there are several algorithms that can be applied for the second task, and they can predict with high accuracy.

The third task is more challenging. As all the patients have the same cancer and the treatment is the same for all of them, the gene expression levels are very similar for all the samples, and finding patterns in them is harder than the previous tasks. Another challenge with these studies is that the number of samples is even smaller than the other cases. Finding the patients with exactly the same disease and having them come back to determine the state of their disease after a certain time is difficult. Often the classifiers that work well for the first two tasks, do not predict the class labels accurately for this type of study. This type of study is very important because, if it is clear that a certain treatment is not going to work for a patient, then the doctors can use some other treatments.

We would like to classify the samples, but since the first two tasks are easier, we focus on predicting the clinical outcome based on gene expression data. Because it is very important and there are not many “good” studies that can predict the clinical outcome accurately. The results for the other tasks will be available at [6].

Chapter 2

Related Work

2.1 Bi-clustering Algorithms

2.1.1 The Problem

The microarray data is stored in an $n \times p$ matrix M, which is defined by a set of rows (genes), $G = \{g_1, g_2, \ldots, g_n\}$, and a set of columns (samples), $S = \{s_1, s_2, \ldots, s_p\}$, where $|G| \gg |S|$. Each element m_{ij} is given as a real value, which is usually the logarithm of the relative abundance of the mRNA of the gene under a specific condition [3]. Here m_{ij} represents (the logarithm of) the expression level of gene i for sample j (or under condition j). Table 2.1 shows an example of the gene expression matrix.

	Sample 1	Sample 2	...	Sample j	...	Sample p
Gene 1	m_{11}	m_{12}	...	m_{1j}	...	m_{1p}
Gene 2	m_{21}	m_{22}	...	m_{2j}	...	m_{2p}
...	...	...	...	...	...	...
Gene i	m_{i1}	m_{i2}	...	m_{ij}	...	m_{ip}
...	...	...	...	...	...	...
Gene n	m_{n1}	m_{n2}	...	m_{nj}	...	m_{np}

Table 2.1: Data matrix for gene expression values.

M_{IJ} is a sub-matrix of M, where $I \subseteq G$ is a subset of genes (rows), and $J \subseteq S$ is a subset of samples (columns).

A *bi-cluster* is a subset of genes that have a similar pattern over a subset of samples, or vice versa. Bi-cluster M_{IJ} is a subset of genes and a subset of samples, where (after re-ordering) $I = \{x_1, \ldots, x_h\}$ such that $I \subseteq G$ and $h \leq n$, and $J = \{y_1, \ldots, y_l\}$ such that $J \subseteq S$ and $l \leq p$. Therefore, a bi-cluster is defined as an $|I| \times |J|$ sub-matrix of M.

Hence, the problem is as follows: Given a data matrix, M, find a set of bi-clusters, B_k where $k = 1, \ldots, K$, such that the set of genes, I_k, in B_k have similar pattern under the set of samples, J_k, in B_k.

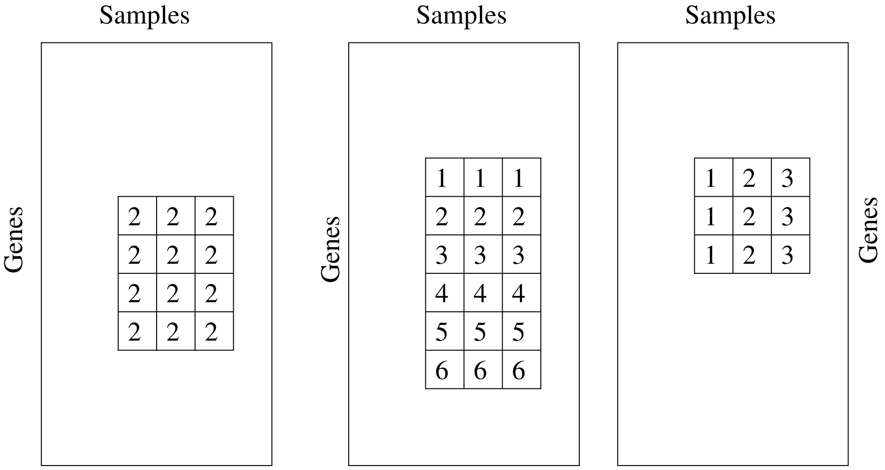
In [3], the authors established a relation between data matrices and graph theory. Using this connection, then they discussed the complexity of this problem. They represented the data matrix as a *weighed bipartite graph.* A graph $G = (V, E)$, where V is the set of vertices and E is the set of edges, is bipartite if its vertices can be partitioned into two sets L and R ($V = L \cup R$, $L \cap R = \{\}$), such that every edge in E has exactly one end in L and the other in R. The data matrix M can be viewed as a weighted bipartite graph where each node $n_i \in L$ corresponds to a gene and each node $n_j \in R$ corresponds to a sample. The edge between nodes n_i and n_j has weight m_{ij} from the data matrix.

The complexity of the problem depends on the exact problem formulation and evaluation of the quality of the given bi-cluster. Almost all variations of this problem are NP-complete. Because of the complexity of the problem, many bi-clustering algorithms use heuristic approaches to find the bi-clusters. The ones that avoid heuristics can have exponential worst-case running time [3].

2.1.2 Previous Work

Madeira et al. [3] provides a comprehensive survey that analyzes a large number of existing approaches to bi-clustering. They classify them according to the type of bi-clusters they can find, the patterns of bi-clusters that are discovered, the methods used to perform the search, and the target applications. They identify different types of bi-clusters that the bi-clustering algorithms can find:

1. Bi-clusters with constant values; $\forall m_1, m_2 \in M_{IJ}\ m_1 = m_2 = \mu$, where μ is the baseline value for the bi-cluster (Figure 2.1(a)).



(a) Constant Values (b) Constant Rows (c) Constant Columns

Figure 2.1: Examples of type 1 and type 2 bi-clusters.

2. Bi-clusters with constant values on rows or column (Figures 2.1(b), 2.1(c)):
   - Constant rows: $\forall i \in I, j_1, j_2 \in J\ m_{ij_1} = m_{ij_2}$.
   - Constant columns: $\forall i_1, i_2 \in I, j \in J\ m_{i_1 j} = m_{i_2 j}$.

3. Bi-clusters with coherent values; There are two models for bi-clusters with coherent values; Figure 2.2 shows an example for each of these models:
   - *The additive model:* In this model each element of the bi-cluster is represented as $m_{ij} = \mu + \alpha_i + \beta_j$, where $\mu, \alpha_i, \beta_j \in \mathbb{R}$, μ is the baseline value for the bi-cluster, α_i is the adjustment for row i, and β_j is the adjustment for column j.
   - *The multiplicative model:* In this model, each element of the bi-cluster is represented as $m_{ij} = \mu \times \alpha_i \times \beta_j$, where $\mu, \alpha_i, \beta_j \in \mathbb{R}$, μ is the baseline value for the bi-cluster, α_i is the adjustment for row i , and β_j is the adjustment for column j. Our RoBic fits into this category.

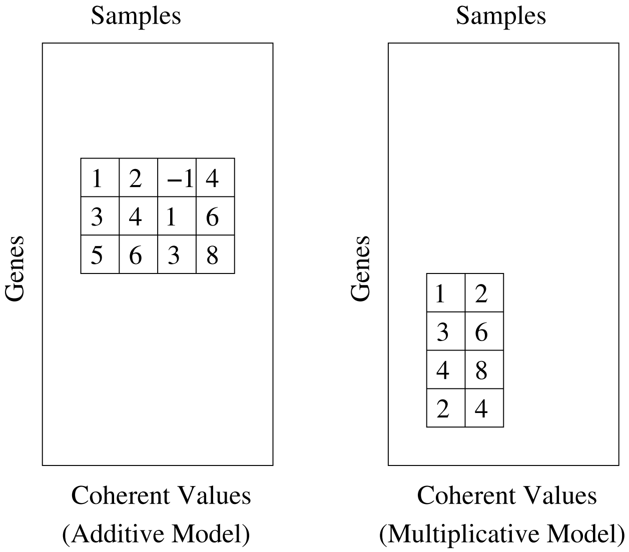


Figure 2.2: Examples of type 3 bi-clusters. For the additive model, $\mu = 0$, $\alpha = [1, 3, 5]$, and $\beta = [0, 1, -2, 3]$; for multiplicative model, $\mu = 1$, $\alpha = [1, 3, 4, 2]$, and $\beta = [1, 2]$.

4. Bi-clusters with coherent evolutions: these are the subsets of rows and subsets of columns with coherent values regardless of the exact numeric values in the data matrix. This property can be observed on the entire bi-cluster (on both rows and columns of the sub-matrix), on the rows of the bi-cluster, or on the columns of the bi-clusters. For example, in gene expression data, we might be interested in finding a subset of genes that are up-regulated or down-regulated across a subset of samples without taking into account their actual expression values in the data matrix. See Figure 2.3 for examples.

To find type 1 bi-clusters (with constant values), the rows and columns of the data matrix have to be re-arranged somehow in order to have best presentation of the similar rows and columns together. Then find subsets of rows and subsets of columns with constant values. In other words, for gene expression data, constant bi-clusters find the subset of genes with constant expression values within a subset of samples. A perfect constant bi-cluster is a sub-matrix M_{IJ}, where all $m_{ij} = \mu$, for all $i \in I$, and $j \in J$. But this approach finds bi-clusters that have a subset of genes with constant expression values for a subset of samples when it is performed on data without noise, which is not true for a lot of microarray data. This means

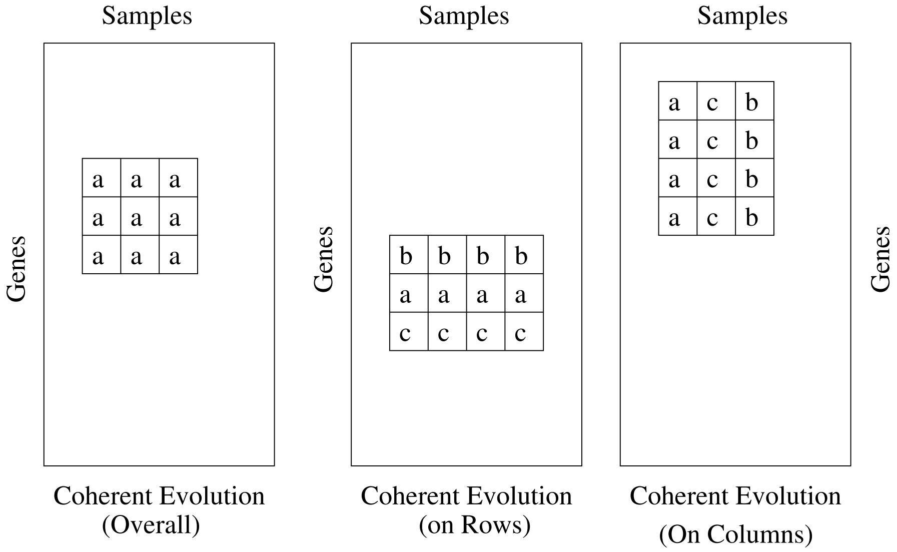


Figure 2.3: Examples of type 4 bi-clusters. For example, “a” represents the genes that are up-regulated across a subset of samples, or “b” represents the genes that are down-regulated across a subset of samples.

that the values m_{ij} for a bi-cluster are generally presented as $\eta_{ij} + \mu$, where η_{ij} is the noise associated with position (i, j).

A bi-cluster with constant values in columns identifies a set of samples with constant expression values across a subset of genes, allowing the expression levels to differ from sample to sample. The same explanation can be applied for a bi-cluster with constant values in rows.

Sheng et al. [9] tackle the problem in the Bayesian framework, by presenting a bi-clustering strategy based on a frequency model for expression pattern of a bi-cluster and Gibbs sampling for parameter estimation. Their goal is to find the genes and samples of a bi-cluster, by representing the pattern of a bi-cluster as a probabilistic model described by the posterior frequency of every discretized expression level discovered under each sample of the bi-cluster.

There are several problems with this approach:

1. They work with *discretized* microarray data. The problem with discretization is that the noise blurs the differences between meaningful values and background. Having noise in the data (which is true for most microarray datasets) results in small changes in the expression values. As a simple example, sup-

pose that $m_{ij} \in \mathbb{R}$, and $0 \leq m_{ij} \leq 1$, $\forall m_{ij} \in M$ and the gene expression value for the k^{th} gene $(1 \leq k \leq n)$ and the l^{th} sample $(1 \leq l \leq p)$ is $m_{kl} = 0.5 - \epsilon_{kl}$, where ϵ_{kl} is the amount of noise associated with the kl^{th} element. Suppose the data is discretized to matrix D such that

- $d_{ij} = 0$ if $0 \leq m_{ij} < 0.5$, and
- $d_{ij} = 1$ if $0.5 \leq m_{ij} \leq 1$.

The noise level in the data can cause an element with value of $M_{kl} = 0.5 - \epsilon_{kl}$ to be in the wrong category of discrete data. This can cause problem, because it is not possible to distinguish M_{kl} from another m_{ij} whose expression value is very small (i.e. 0.0001), and they are both in the same category after discritization.

2. They use *multinomial distributions* to model the data in every column (or row) in a bi-cluster, which might not be a correct model for the data. They refer to this model of the bi-cluster as the pattern Θ, a matrix that each column $\Theta_{.j}$ contains the parameters for the j^{th} independent multinominal distribution.

$$\Theta = \begin{bmatrix} \theta_{11} & \dots & \theta_{1j} & \dots & \theta_{1w} \\ \theta_{21} & \dots & \theta_{2j} & \dots & \theta_{2w} \\ \vdots & & \vdots & & \vdots \\ \theta_{l1} & \dots & \theta_{lj} & \dots & \theta_{lw} \end{bmatrix}$$

where $0 \leq \theta_{ij} \leq 1$, $\sum_i \theta_{ij} = 1$, and l is the number of genes, and w denotes the total number of conditions in the bi-cluster.

3. They assume that these multinomial distributions for different columns in a bi-cluster are *mutually exclusive*. They work with a row-column orientation of the data matrix and want the values within bi-clusters to be consistent across the rows of bi-clusters for the selected columns, but these values may differ for each column.

4. Their probabilistic model considers only the presence of a *single* bi-cluster the data set.

Sheng et al. applied their algorithm on a leukemia microarray data set, which grouped samples based on their expression behavior over a subset of genes. The samples collected in every bi-cluster came from the same category.

Kluger et al. [10] look for checker-board structures (bi-clusters with coherent values) in the data matrix by integrating bi-clusters of rows and columns with normalization of the data matrix. They assumed that after a particular normalization, which was designed to find bi-clusters, if they exist, the contribution of a bi-cluster is given by a multiplicative model: $m_{ij} = \mu \times \alpha_i \times \beta_j$, where $\mu, \alpha_i, \beta_j \in \mathbb{R}$, μ is the value within the bi-cluster, α_i is the adjustment for row i, and β_j is the adjustment for column j. They also use gene expression data and see each value m_{ij} in the data matrix as the product of the tendency of gene i to be expressed in all samples and the tendency of all genes to be expressed in sample j. In order to evaluate the quality of a bi-cluster, they tested the results against a null hypothesis of no structure in the data matrix. They presented their results by figures of clusters found from the datasets.

But all of these bi-clustering approaches evaluate separately the contributions of bi-clusters, without considering the interactions between bi-clusters.

In 2000, Lazzeroni and Owen [7] introduced the *Plaid models* for gene expression data. These models are of a form of bi-clustering, in which the value of an element in the data matrix is viewed as a sum of terms called “layers” (which is the equivalent of bi-clusters). To find the bi-clusters, they first form a color image of the data on an n by p grid, with each cell colored according to the value of m_{ij}. Then they use some ways of reordering the rows and column in order to group the similar rows and similar columns together, to form an image with blocks of similar colors (values). Each block is nearly uniformly colored and the image outside the blocks has natural background color. Figure 2.4 shows an example.

In other words:

$$m_{ij} = \mu_0 + \sum_{k=1}^{K} \theta_{ijk} \rho_{ik} \kappa_{jk}$$

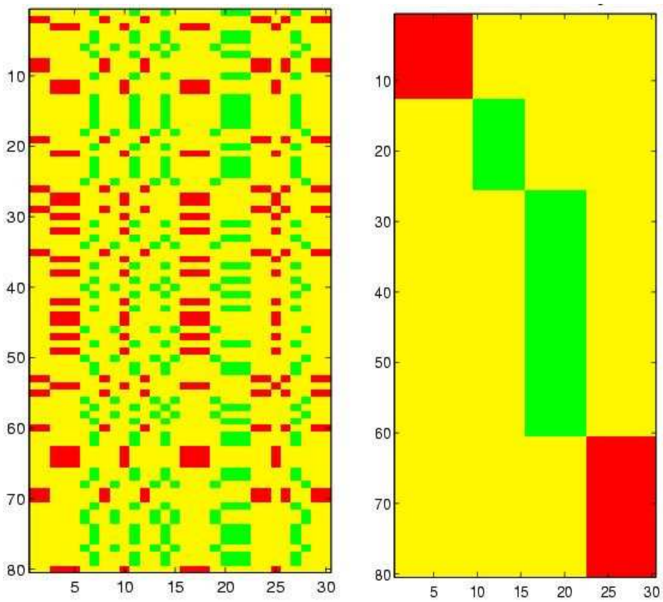


Figure 2.4: Color image of data for Plaid models before and after reordering disjoint rows and columns [7].

where μ_0 is the background color, K is number of bi-clusters, and θ_{ijk} is the color in bi-cluster k specified by row i and column j, ρ_{ik} and κ_{jk} are binary values that represent the membership of row i and column j in bi-cluster k, respectively. That is,

- $\rho_{ik} = 1$ if gene i is in bi-cluster k, otherwise it is 0.
- $\kappa_{jk} = 1$ if sample i is in bi-cluster k, otherwise it is 0.

If every gene and every sample is in exactly *one* bi-cluster, we would have $\sum_k \rho_{ik} = 1$ for all i, and $\sum_k \kappa_{jk} = 1$ for all j. However, Plaid considers a more general model, where for overlapping bi-clusters $\sum_k \rho_{ik} \geq 2$ for some i, and $\sum_k \kappa_{jk} \geq 2$ for some j. For the genes or samples that do not fit into any bi-clusters, $\sum_k \rho_{ik} = 0$ for some i, and $\sum_k \kappa_{jk} = 0$ for some j.

The Plaid algorithm finds the bi-clusters one at a time. It searches for bi-cluster K in the residual

$$R_{ij} = m_{ij} - \theta_{ij0} - \sum_{k=1}^{K-1} \theta_{ijk}\rho_{ik}\kappa_{jk},$$

,where $\theta_{ij0} = \mu_0$, to minimize the sum of squared error:

$$\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{p}(R_{ij} - \theta_{ijK}\rho_{iK}\kappa_{jK})^2.$$

They use the EM algorithm to find the θ, ρ, and κ values. Each bi-cluster that the Plaid algorithm finds may represent the presence of a particular set of biological process or conditions. θ_{ijk} values provide the effects of bi-cluster k upon the genes and samples.

In [3], the Plaid algorithm is called the *general additive model* (Figure 2.5), and after comparing an extensive list of available algorithms, they conclude that this model is “rich enough to appropriately model very complex interactive processes”. Therefore, we will compare our bi-clusters with Plaid’s.

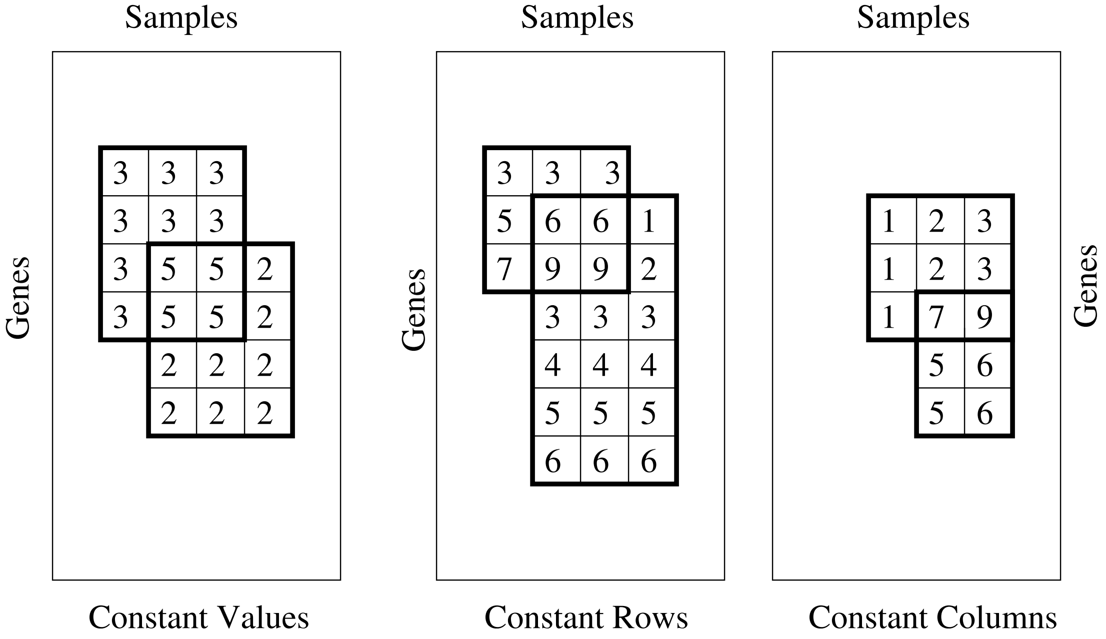


Figure 2.5: Overlapping bi-clusters with general additive model.

Preli et al. [4] provides a systematic comparison and evaluation of prominent bi-clustering methods in the light of gene classification. In particular, the authors focus on the following questions:

- What comparison / validation methodology is adequate for the bi-clustering context?

- How meaningful are the bi-clusters selected by the existing methods?
- How do different methods compare to each other?

They introduced a new algorithm (Bimax) and compared it with five different algorithms, and hierarchical clustering.

Bimax works with *discretized* microarray data, where the data is converted into a binary matrix. The idea is illustrated in Figure 2.6. First, it tries to identify the areas of M that has 0's and can be excluded from next inspection. It partitions the data matrix M into three sub-matrices, one of which contains 0's only. The algorithm is then recursively applied to the remaining sub-matrices, called U and V below. The recursion ends if the current matrix represents a bicluster (it contains only 1's). The algorithm stops if either q biclusters have been selected (for real datasets, q is set to 100) or none of the remaining ones fulfills the selection criterion. The selection procedure they adopted follows a greedy approach: in each step, the largest of the remaining biclusters is chosen that has less than $o\%$ of its cells in common with any previously selected bicluster; for real datasets a they set maximum overlap of $o = 25\%$.

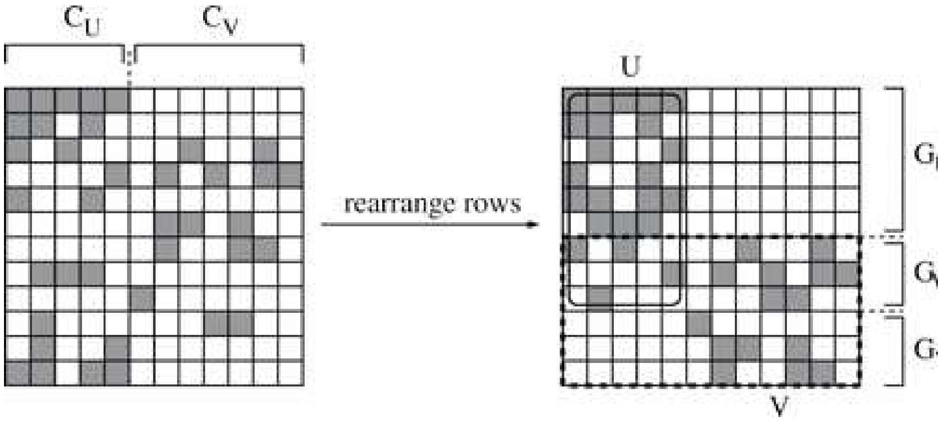


Figure 2.6: Illustration of the Bimax algorithm. "To divide the input matrix into two smaller, possibly overlapping sub-matrices U and V, first the set of columns is divided into two subsets C_U and C_V, here by taking the first row as a template. Afterwards, the rows of E are resorted: first come all genes that respond only to conditions given by C_U, then those genes that respond to conditions in C_U and in C_V and finally the genes that respond to conditions in C_V only. The corresponding sets of genes G_U, G_W and G_V then define in combination with C_U and C_V the resulting sub-matrices U and V which are decomposed recursively" [4].

At the end they compared the algorithms based on GO annotations and metabolic pathway information. They conclude that most bi-clustering algorithms, in this study, are capable of dealing with overlapping transcription modules and generate functionality enriched clusters. They have implemented these algorithms [23], but we could not use this as it is not easy to store the found bi-clusters for further analysis and comparisons.

2.2 Classification Algorithms

We are interested in the following classification problem: Given DNA-microarray data for a set of samples characterized by a given class, identify “patterns” of gene expression that can be used to predict the sample class.

Supervised learning methods have been established (e.g. [21]) to predict if a sample belongs to a known class or not depending on the sample’s gene expression profile. Van’t Veer et al. [11] applies supervised classification to identify a gene expression signature that can predict the clinical outcome of breast cancer, that is predicting if a patient remains disease-free after a certain period of time or not. They selected approximately $5,000$ genes (the ones that were significantly regulated across a group of samples) from the $25,000$ genes on the microarray. The correlation coefficient of the expression for each gene with disease outcome was calculated and 231 genes were found to be significantly associated with the disease outcome based on all samples. Then they rank-order these genes based on the magnitude of the correlation coefficient. Then, the number of genes in the ’prognosis classifier’ was optimized by sequentially adding subsets of 5 genes from the top of this ranked-ordered list and evaluating its power for correct classification using *cross-validation*. They gain the best accuracy, 83%, using 70 of the marker genes. In cross-validation, a portion of the data is set aside as training data leaving the remainder as testing data. In *K-fold cross-validation*, the original sample is partitioned into K subsamples. From the K subsamples, a single subsample is retained as the test data for testing the model, and the remaining $K-1$ subsamples are used as training data. The cross-validation process is then repeated K times (the folds), with each of the K subsamples used exactly once as the test data. The K results from the folds then can be averaged to produce a single estimation.

The problem with their approach was that they selected the features (genes) by looking at the *entire dataset*. Instead, they should have done the feature selection in each cross-validation fold. By doing this, they may select a different set of attributes for each cross-validation fold because each fold uses a different training set to decide which attributes to keep. When they corrected their approach, the prediction accuracy went down by 10%, to 73% [24].

Golub et al. [12] developed another method for class prediction. First they find the genes whose expression values were strongly correlated with the class distinction to be predicted. Then they developed a method called "neighborhood analysis" to define an "idealized expression pattern" corresponding to a gene that is uniformly high in one class and uniformly low in the other. Then they use a fixed subset of "informative genes", chosen based on their correlation with the class distinction. Each informative gene casts a "weighted vote", and the votes were summed to determine the winning class as well as a "prediction strength". They applied their method to a human leukemia database. The predictor made "strong prediction" for AML versus ALL class prediction. In total, the predictor made strong prediction for 29 of the 34 samples, and the accuracy was 100% for those 29 samples. But the neighborhood analysis found no striking excess of genes correlated with response to chemotherapy, comparing to the AML-ALL distinction. They found no evidence of a strong multi-gene expression signature correlated with clinical outcome. We report our result on this data set in Chapter 5.

To determine if different types of tumors could be molecularly distinguished, Pomeroy et al. [13] used principal component analysis, in which the high dimensionality of data was reduced to three viewable dimensions representing linear combination of features (genes) that account for most of the variance in the original dataset. The normal samples were easily separable from tumors and the different tumor types were similarly separable, using hierarchical clustering. For tumor classification using these genes, they used a weighted-voting algorithm and evaluated by cross-validation testing. Their system achieved 83% classification accuracy of tumors. Next, they checked if their clusters were correlated with patient survival. Their method could not find any statistically significance in the proportion of survivors compared with treatment failure in each cluster. They then tried another technique, a k-nearest neighbor algorithm (k-NN), to compute the distance of a test sample to each of the training samples, with known classes. Then they predicted the class of the sample to be that of the majority of the k-closest sample. The outcome predictions based on gene expression were statistically significant for k-NN models ranging from 2 to 21 genes. The optimal predictions made by an 8-gene model with 78% accuracy.

Singh et al. [14] also use a k-NN approach with gene selection for predicting the class label of unknown prostate cancer samples. They use the same approach for predicting the clinical outcome and obtained 90% accuracy for a 5-gene model with 2 nearest neighbors. They performed a 1000 permutation test [1] on class labels (recurrence versus non-recurrence) and found 37 of their models whose accuracy matched or exceeded theirs.

Gordon et al. [16] described an alternative approach using gene expression measurements to predict clinical parameters in cancer. They select 2 genes randomly, then use the ratios of their expression levels and choose thresholds to accurately distinguish between different classes of samples. This approach works fine for predicting different types of tumors in lung cancer in mesothelioma, but when applied on clinical outcome dataset [13] , the prediction accuracy dropped by 10%, to 68%.

A widely used classification approach for microarrays is the "shrunken centroid method" [18], which finds subsets of genes that best characterize each class. It shrinks the class centroids (by a shrinkage parameter δ) toward the overall centroids after standardizing by the within-class standard deviation for each gene. This standardization has the effect of giving higher weight to genes whose expression is stable within samples of the same class. The shrinkage it uses is called *soft thresholding*. The threshold, δ, can be determined by a cross-validation process. This method has the property that many of the genes are eliminated from the class

[1] see Chapter 5

prediction as δ increases.

Somorjai et al. [19] used exhaustive search as feature selection method (with least trimmed squares, LTS, or linear discriminant analysis, LDA, and leave-one-out cross-validation), requesting only two genes. LDA finds the linear combination of features that best separate two or more classes of samples. Then, they use the selected genes for class prediction. The prediction accuracy is between 92.9% and 97.1% (using 3-7 genes) for a prostate cancer versus healthy dataset.

Pranckeviciene et al. [20] uses the optimal features found by the LDA EX-FS classifier. They find the optimal features by using exhaustive-search-based feature selection (see their paper for more detail). They order the genes according to the classifier accuracy they produce. Then, they compare the classification results obtained with four different classifiers, using this optimal number of features.

For microarray classification, in general, the found best classifiers are SVM and the shrunken centroids methods. In addition, most findings confirm that feature selection leads to better classification performances, and that it is generally advisable to perform some dimensionality reduction, or at least exploratory analysis prior to classification.

The usual method is to screen the genes by employing univariate statistical tests for class discrimination, to select a subset of genes. Then the quality of this subset is evaluated by its classification performance on an independent test set. We introduce an algorithm to reduce the dimension of the dataset by finding the bi-clusters (Chapter 3). Then we use these bi-clusters to predict the class of samples based on their membership in the bi-clusters (Chapter 4).

Chapter 3

Finding Bi-clusters

Recall from Section 2.1.1 that the microarray data is stored in an $n \times p$ matrix M. Each element m_{ij} is given as a real value, which represents the expression level of gene i for sample j (or under condition j). A *bi-cluster* is a subset of genes that have similar pattern over a subset of samples. Bi-cluster M_{IJ} is a subset of genes and a subset of samples, where $I \subseteq \{1, \ldots, n\}$ and $J \subseteq \{1, \ldots, p\}$. Therefore, a bi-cluster is defined as a sub-matrix of M; see Section 2.1.1 for more detail. This chapter describes the structure of our algorithm for finding the bi-clusters.

3.1 The Algorithm

We have a large matrix M, where $|M|$ is $n \times p$ (e.g. $50,000 \times 80$); see Figure 3.1. Our first sub-goal is to find two vectors, α and β, of size $|\alpha| = n$ (e.g. $50,000$) and $|\beta| = p$ (e.g. 80) that minimize

$$\|M - \alpha.\beta^T\| = \sum_{ij} (m_{ij} - \alpha_i.\beta_j)^2.$$

The algorithm is based on the least squares rank-1 matrix approximation to M. Computing the singular value decomposition (SVD) of M gives three matrices, U,

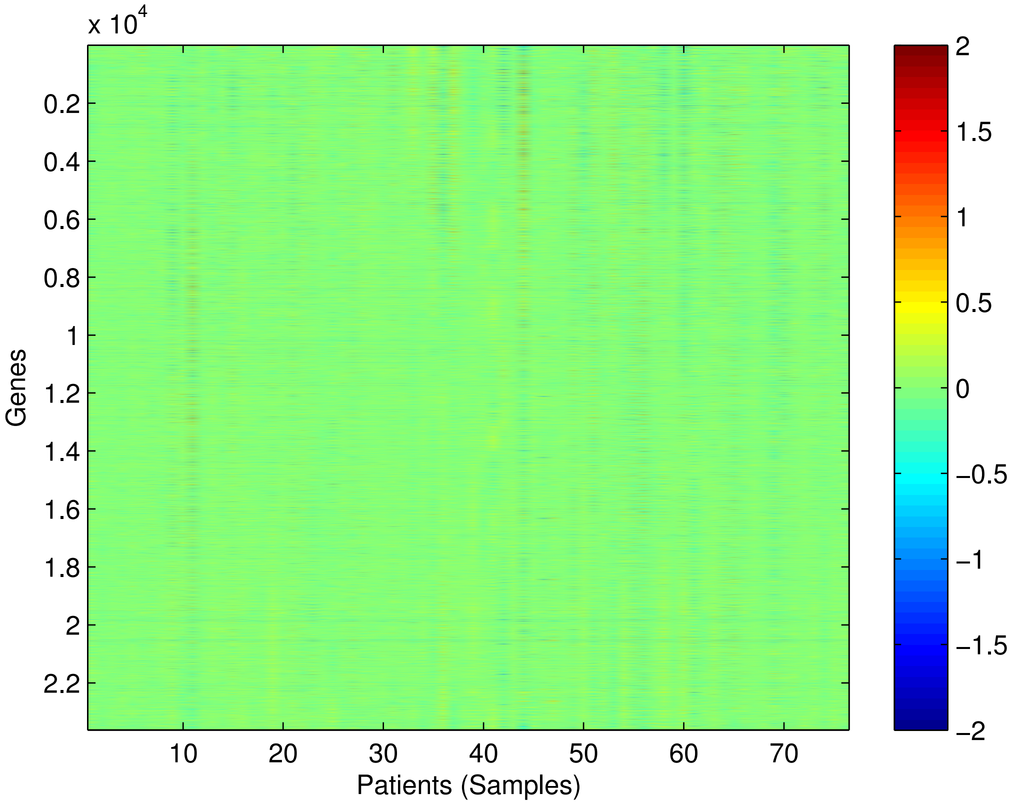


Figure 3.1: Plot of gene expression values (data matrix M) for breast cancer [11] data.

S, and V as shown in Figure **??** such that:

$$[U, S, V] = SVD(M, d)$$

- U is $n \times d$ with orthonormal columns (d is given or $d = min\{n, p\}$),
- S is a diagonal matrix of d eigenvalues in decreasing order (of size $d \times d$), and
- V is $p \times d$ with orthonormal columns.

Then, $U.S.V^T$ is the closest rank-d approximation to M. For proof and more details see [25, page 72], [26, page 322], and [27, page 288]. The columns of U are called the *left singular vectors* of M and the columns of V are called the *right*

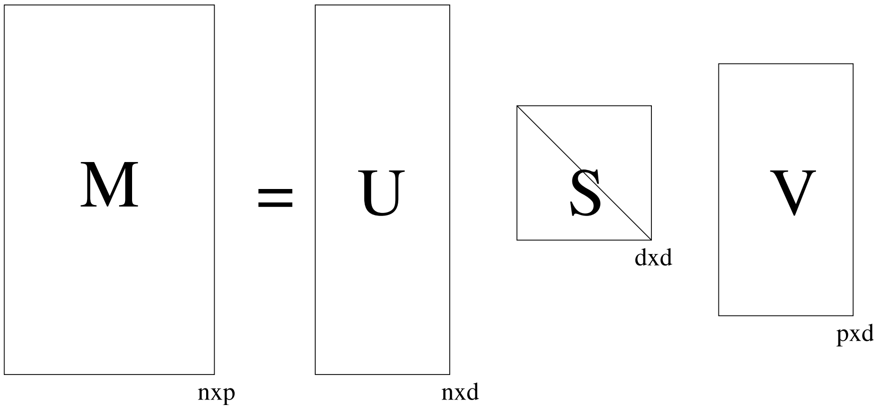


Figure 3.2: Singular value decomposition of the data matrix M.

singular vectors of M. If $d = 1$ (rank-1 approximation) then S is a scalar and we set:

$$\alpha = U$$

$$\beta = V.S$$

Then, $\alpha.\beta^T = U.S.V^T$ is the least square rank-1 approximation of M. Note that since S is a scalar, we can set $\alpha = U.S$ and $\beta = V$, too. But, for the rest of our computations, it does not make a difference. We let α represent the genes' vector and β represent the samples' vector.

In the next step, we sort the values in α and β in descending order producing $\alpha^{(s)}$ and $\beta^{(s)}$, and re-arrange the rows and column in M such that the order of genes and samples in M is the same as the $\alpha^{(s)}$, and $\beta^{(s)}$, vectors. Let's call the re-arranged data matrix M_{sorted}; see Figure 3.3. Figure 3.4 shows the plot of $|\alpha^{(s)}|$ values (absolute values) and Figure 3.5 shows the plot of $|\beta^{(s)}|$ values for breast cancer data as an example.

Now that we have two vectors for genes and samples with sorted values, we need to define a "hinge function" that gives a subset of genes and subset of samples that define a bi-cluster.

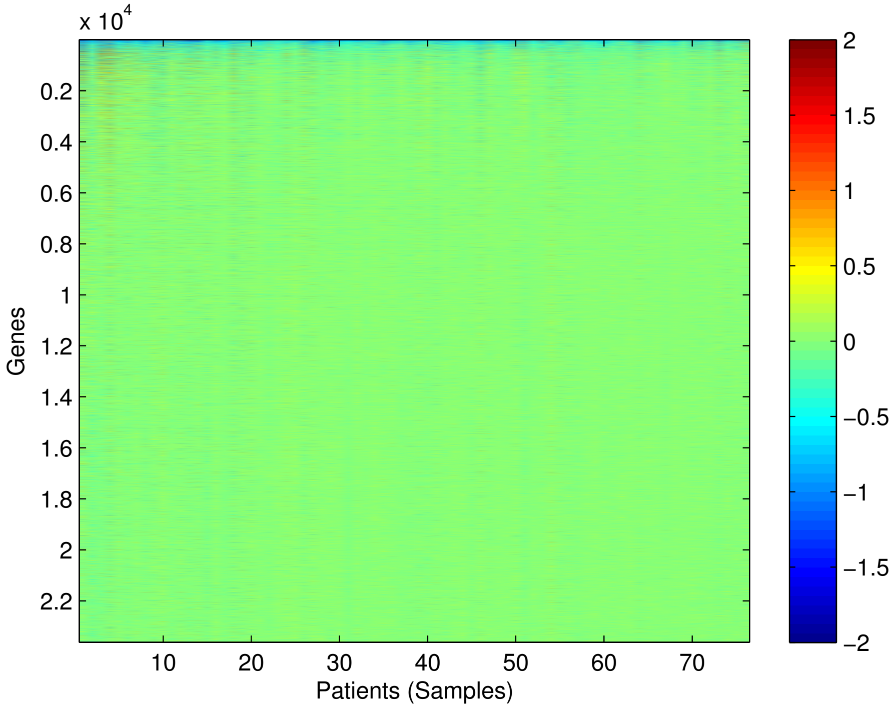


Figure 3.3: Plot of re-arranged gene expression values (data matrix M_{sorted}) for breast cancer [11] data.

We find the subsets by approximating the values in the vectors by two best lines. That is, for each $i = 1, \ldots, n$, find the best fitting line for values $\alpha_1^{(s)}$ to $\alpha_i^{(s)}$, which has error $e_1^{(i)}$, and the best fitting line for values $\alpha_{i+1}^{(s)}$ to $\alpha_n^{(s)}$, with error $e_2^{(i)}$. Then choose i, the separation point for genes, such that $e_1^{(i)} + e_2^{(i)}$ is minimum; $i = argmin\{e_1^{(i)} + e_2^{(i)}\}$. Therefore, the genes from index 1 to i are in the current bicluster. We can find the samples' separation point, j, (the samples in the bi-cluster with similar pattern) using the same procedure. Because of the large number of genes, we use only the first half of the genes in the sorted vector ($\alpha_1^{(s)}, ..\alpha_{\frac{n}{2}}^{(s)}$) to find the best fitting lines for gene values. See lines shown in Figure 3.4 and 3.5.

We have tried other approaches for finding the subsets of genes and samples from the α and β vectors. Appendix A describes other hinge functions that we have considered, which did not work as well as the "two line approach" shown above. It also describes other approaches for finding the bi-clusters that we have tested.

After finding the genes and samples in the first bi-cluster, we subtract their values off from the data matrix, and repeat the same process on the remainder of the

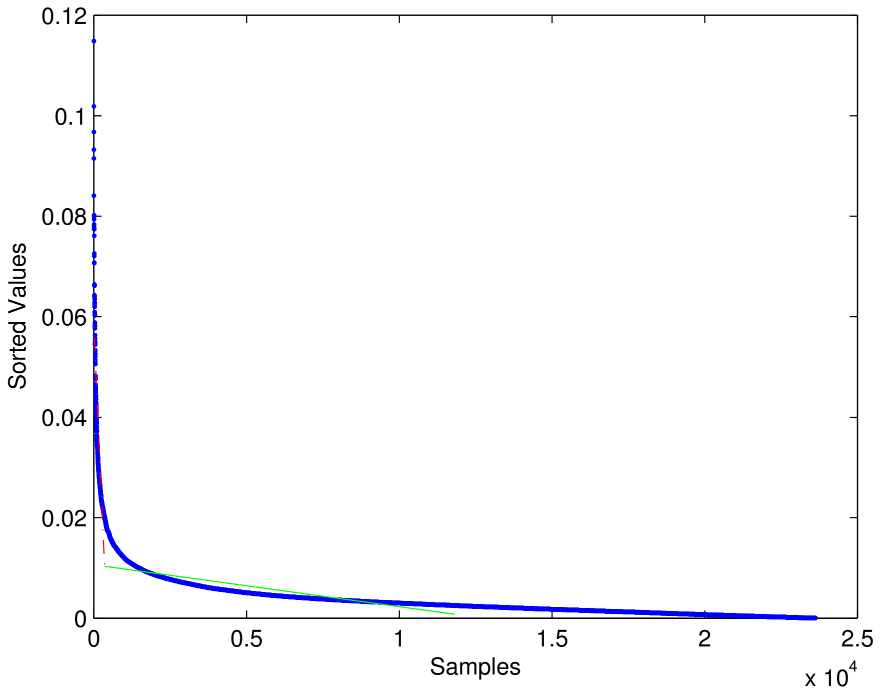


Figure 3.4: Plot of sorted α (absolute) values for breast cancer [11] data.

data matrix to find the next bi-clusters. The details of this process are as follows:

Let α_τ be a vector whose first i^{th} elements are equal to the first i^{th} elements of the sorted α and the rest of its elements are 0. That is,

$$\alpha_\tau = \{\alpha_1^{(s)}, \alpha_2^{(s)}, ..., \alpha_i^{(s)}, 0, 0, .., 0\}$$

and $|\alpha_\tau| = |\alpha|$ (size of α_τ is the same as α). Similarly,

$$\beta_\xi = \{\beta_1^{(s)}, \beta_2^{(s)}, ..., \beta_j^{(s)}, 0, 0, .., 0\}$$

and $|\beta_\xi| = |\beta|$. Then,

$$M^{(2)} = M_{sorted} - \alpha_\tau . {\beta_\xi}^T.$$

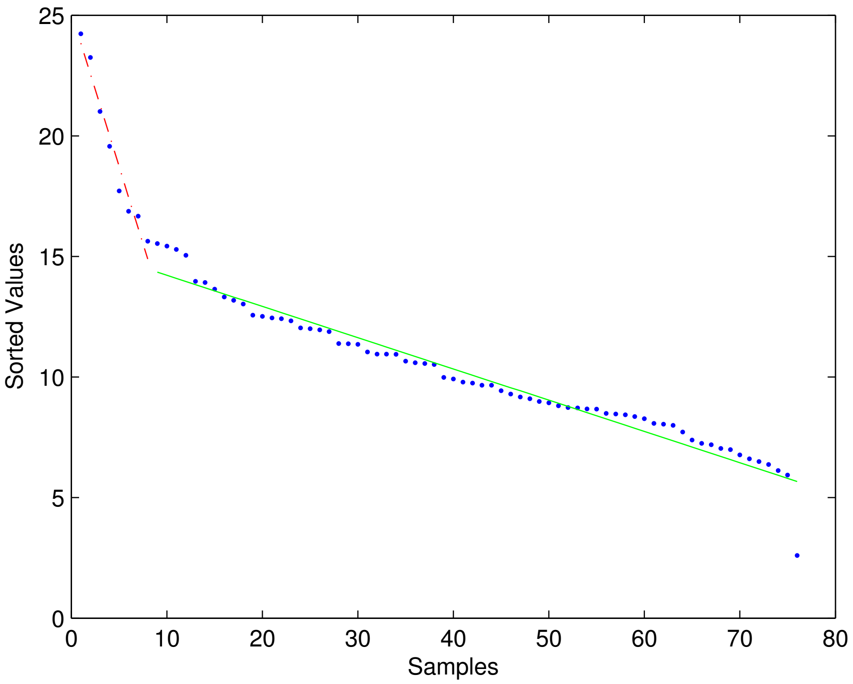


Figure 3.5: Plot of sorted β (absolute) values for breast cancer [11] data.

$M^{(2)}$ is the new data matrix formed after removing the values for the genes and samples in the first bi-cluster. We repeat the same process on the $M^{(2)}$ to find the next bi-clusters. That is, find the new $\alpha^{(2)}$ and new $\beta^{(2)}$ vectors that is the best rank-1 approximation to the $M^{(2)}$. Find the subsets of genes and samples from the new vectors. Subtract their values from $M^{(2)}$ to produce $M^{(3)}$, and repeat the process on the remainder matrix, $M^{(3)}$. Figure 3.6 summarizes our algorithm.

Since we are using the bi-clusters to reduce the dimensionality of the data matrix, we can repeat this process until we find the number of the bi-clusters that gives us the best prediction accuracy for the sample classification. Chapter 4 explains how the bi-clusters can be used to predict a particular label over two classes of samples.

$M^{(1)} = M$.

for $i = 1..30$ do

1. Compute the *largest* singular value decomposition of $M^{(i)}$,

$$[U, S, V] = SVD(M^{(i)}, 1).$$

2. Set $\alpha = U$ and $\beta = V.S$,
   ($\alpha.\beta^T$ is the best rank-1 approximation to $M^{(i)}$)
   - α represents the genes' vector, and
   - β represents the patients' vector.
3. Sort the values in α and β vectors, to get $\alpha^{(s)}$ and $\beta^{(s)}$.
4. Re-arrange the rows and columns in $M^{(i)}$ such that the order of genes and patients is the same as $\alpha^{(s)}$ and $\beta^{(s)}$. This gives $M^{(i)}_{Sorted}$.
5. Find the separation point for genes, i, and for patients, j by approximating the values in α and β vectors using a hinge function. Here the hinge function is fitting the values in the vectors to two best lines. The genes from index 1 to i and the patients from index 1 to j are in the current bi-cluster.
6. Set
   - $\alpha^{(s,i)}_{\tau} = \{\alpha^{(s)}_1, \alpha^{(s)}_2, ..., \alpha^{(s)}_i, 0, 0, ..., 0\}$,
   - $\beta^{(s,j)}_{\xi} = \{\beta^{(s)}_1, \beta^{(s)}_2, ..., \beta^{(s)}_j, 0, 0, ..., 0\}$.
7. $M^{(i+1)} = M^{(i)}_{Sorted} - \alpha^{(s,i)}_{\tau}.\beta^{(s,j)T}_{\xi}$.

Figure 3.6: Summary of rank-1 bi-cluster algorithm (RoBic).

Chapter 4

Bi-cluster Classifier

Chapter 3 explains how to find the bi-clusters, subsets of genes and subsets of samples with similar patterns. This chapter explains how we can use the bi-clusters to reduce the dimensionality of the data matrix, and predict the class of a novel sample.

4.1 The Problem

We have p samples, and for each of these samples we have n different features, which is their gene expression values, such that $n \gg p$. We also know the class labels for q samples ($q < p$), which form the training set. The goal is to predict the class labels for the rest of the samples, the test set (which includes $p - q$ samples), based on the gene expression values for both training set and the test set.

Instead of directly using all n features, or selecting a subset of the n features, to predict the samples' class label, we reduce the dimension of the data matrix M, which is $n \times p$, to $p \times K$ by finding K bi-clusters from M, where $K \ll n$.

Hence, the problem is as follows: Given a data matrix, M, a set of bi-clusters $\{B_1, B_2, \ldots, B_K\}$ from M, and a subset of samples with known classes, predict the class label for the samples in the test set.

4.2 The Algorithm

Suppose we have K bi-clusters, $\{B_1, B_2, \ldots, B_K\}$, each of which has an associated subset of genes and a subset of samples. The class label is known for a group of samples in M, the training set, but is unknown for the rest, the test set. We reduce the dimensionality of the data matrix M ($n \times p$), which is originally very large, based on the bi-clusters to get matrix R, which is $p \times K$. We do this by creating a bit vector for each sample, where the element r_{jk} of R is 1 if the j^{th} sample is in the k^{th} bi-cluster, otherwise it is 0. Table 4.1 shows an example of the reduced dimension of the data matrix, the training set, and the test set.

	B_1	B_2	$\ldots$	B_k	$\ldots$	B_K	Sample Class
Sample 1	1	1	$\ldots$	0	$\ldots$	1	$+$
Sample 2	1	0	$\ldots$	0	$\ldots$	0	$-$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
Sample j	1	1	$\ldots$	1	$\ldots$	1	$+$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
Sample q	1	1	$\ldots$	0	$\ldots$	1	$+$
Sample $q+1$	0	0	$\ldots$	0	$\ldots$	1	?
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	?
Sample p	1	1	$\ldots$	0	$\ldots$	0	?

Table 4.1: The reduced-dimension of the data matrix, R. The element r_{jk} of R is 1 if the j^{th} sample is in the k^{th} bi-cluster, otherwise it is 0.

Now that we have the reduced-dimension version of the data matrix, we can apply any classification algorithm to predict the class labels for the test set. We use two main classification algorithms: "Support Vector Machines" (SVM), and "Naive Bayes" (N.B.) [28]. Figure 4.1 summarizes the algorithm for bi-cluster classifier,RoBicC.

Since finding the bi-clusters and classification based on the bi-clusters are two independent steps, one can replace the rank-1 bi-clustering algorithm, RoBic, with other bi-clustering algorithm to find a (perhaps different) set of bi-clusters, and classify the samples based on those found bi-clusters.

The next chapter presents some prediction accuracies based on the rank-1 bi-

RoBicC(L, Training set, Test set)

1. Find the bi-clusters from the data matrix, M, using both the unlabeled part of the training set and the test set.
2. Build the binary matrix, R, based on the membership of the samples in the bi-clusters.
3. Learn a classifier, C, using R, the training set labels, and learning algorithm L.
4. Run the classification algorithm C, to predict the class label of samples in the test set.

Figure 4.1: Summary of bi-cluster classifier algorithm, (RoBicC[L], where L is the classification algorithm, e.g. SVM or Naive Bayes).

clustering algorithm, and compares them with the results from the Plaid [7] biclusters, and other approaches for classifying samples.

Chapter 5

Datasets and Results

We proposed an algorithm for finding bi-clusters from the microarray data, which reduces the dimension of the data matrix, M. In order to evaluate the quality of the bi-clusters, we proposed a method for classifying samples based on these found bi-clusters. In this chapter we apply our approach on some publicly available datasets and compare the results with other approaches including some based on other types of bi-clusters.

This project was started with a microarray data from the Cross Cancer Institute; see Appendix B for more details. The number of samples in this dataset was too small to verify our approach, and so we sought a dataset with more samples in order to verify both the bi-clustering algorithm and the bi-cluster classifier. Therefore, we used some publicly available microarray datasets [11], [12], [13], [14], [15]. We will mainly use the datasets for predicting the clinical outcome of a certain treatment. By using the publicly available data, we can compare our results with other known results for the same dataset. Table 5.1 describes the dataset characteristics.

Before analyzing the data, we had to deal with the missing values in the data. First, we remove a gene if its expression value was missing for at least one patient (if any such gene existed). Next, we found the bi-clusters using the complete data set. Note that in this process we do not use the any prior knowledge about the class of the samples. We extract 30 bi-clusters from datasets (we can extract more bi-

Info/Dataset	Breast Cancer [11]	AML [12] (Outcome)	Brain [13] (Outcome)	Prostate[14] (Outcome)	Colon [15]
Number of Samples	76	15	60	21	62
Number of Genes	23625	7129	7129	12600	2000
Number of classes	2	2	2	2	2
Class 1 (count)	relapse (32)	fail (8)	fail (39)	recurrent (8)	negative (40)
Class 2 (count)	non-relapse (44)	success (7)	survive (21)	non-recurrent (13)	positive (22)

Table 5.1: Characteristics of the datasets.

clusters if needed). Then, we build the binary matrix, R, of the transformed data, based on the bi-cluster membership; for more details see Chapter 4.

The code for finding the bi-clusters from M is written in MATLAB. A script is written in JAVA to transform M to R, based on the bi-clusters. The transformed matrix is written in *Weka* [33] format (ARFF). Weka is a collection of machine learning algorithms for data mining tasks. It contains tools for data pre-processing, classification, regression, clustering, association rules, and visualization. We use Weka for building the classifier on the transformed data and evaluating the results.

For finding the optimal number of bi-clusters, B_{opt}, we used the following methods:

1. "*opt*" is the optimal number of bi-clusters, in that the prediction accuracy on test set based on bi-clusters $\{B_1, B_2, \ldots, B_{opt}\}$ is maximum.

2. Use the built-in in-fold feature selection algorithms in Weka to find the bi-clusters that give maximum prediction accuracy on test data.

Table 5.2 shows the prediction accuracies for the data sets, based on 5-fold cross-validation. The first row shows the "base-line" percentage based on the number of samples in each class. Base-line percentage is the accuracy of using the majority class. The second row shows the best known published results for each of the datasets. The missing values in this row is because of either the study for that particular dataset was not for classification, or the authors in original paper did not provide the accuracy results. The third and fourth rows show the result for applying Naive Bayes (N.B.) and Support Vector Machines (SVM) classifiers on the original data matrix, M.

	Info/Dataset	Breast Cancer	AML	Brain	Prostate	Colon
1	Baseline%	57.89%	53.33%	65%	61.90%	64.52%
2	Best published result	73%		78%	90%	
3	Naive Bayes	63.18%	46.67%	63.33%	47.62%	58.06%
4	SVM	67.11%	53.33%	65%	47.62%	80.65%
5	Plaid+N.B. {Bi-clusters}	73.68% {7}	73.33% {6},{7},{20}	66.67% {12}	66.66% {9}	90.32% {17}
6	Plaid+SVM {Bi-clusters}	67.11% {7}	66.66% {2},{19}	65% {10}	61.90% {9}	85.48% {15}
7	RoBiC+N.B {Bi-clusters}	86.84% {21}	80% {19-30}	91.67% {10}	90.48% {13,15,16,17}	82% {5,7,10}
8	RoBiC+SVM {Bi-clusters}	88.16% {25}	80% {18-21,27}	91.67% {16-18,20,23}	95.24% {11}	88% {5,10,28}
9	Permut. Test					
	Average	54.08%	51.66%	60.50%	56.77%	62.51%
	Maximum	77.63%	100%	81.67%	90.47%	82.26%
	# above our value	0	27	0	0	0

Table 5.2: Comparing the prediction accuracies for the datasets, based on 30 bi-clusters and 5-fold cross validation.

Row number 5 contains the results for finding the bi-clusters using Plaid [7], where the sample classification is based on Naive Bayes classifier. Each number in curly brackets shows the number of bi-clusters for the specified accuracies, $\{7\}$ means $\{B_1, B_2, \ldots, B_7\}$ were used for building the classifier. Row 6 is similar to row 5, except that SVM classifier is used for sample classification. Similarly, rows 7 and 8 show the prediction accuracies based on the bi-clusters found using RoBiC. The majority of the bi-clusters that we find have the property that most (or all) of the samples in a bi-cluster belongs to a particular class of the samples. Figure 5.1 shows the samples in the first 25 bi-clusters for breast cancer data and their class.

Finally the last row shows the results (the average prediction accuracy, the maximum prediction accuracy, and the number of models with prediction accuracies

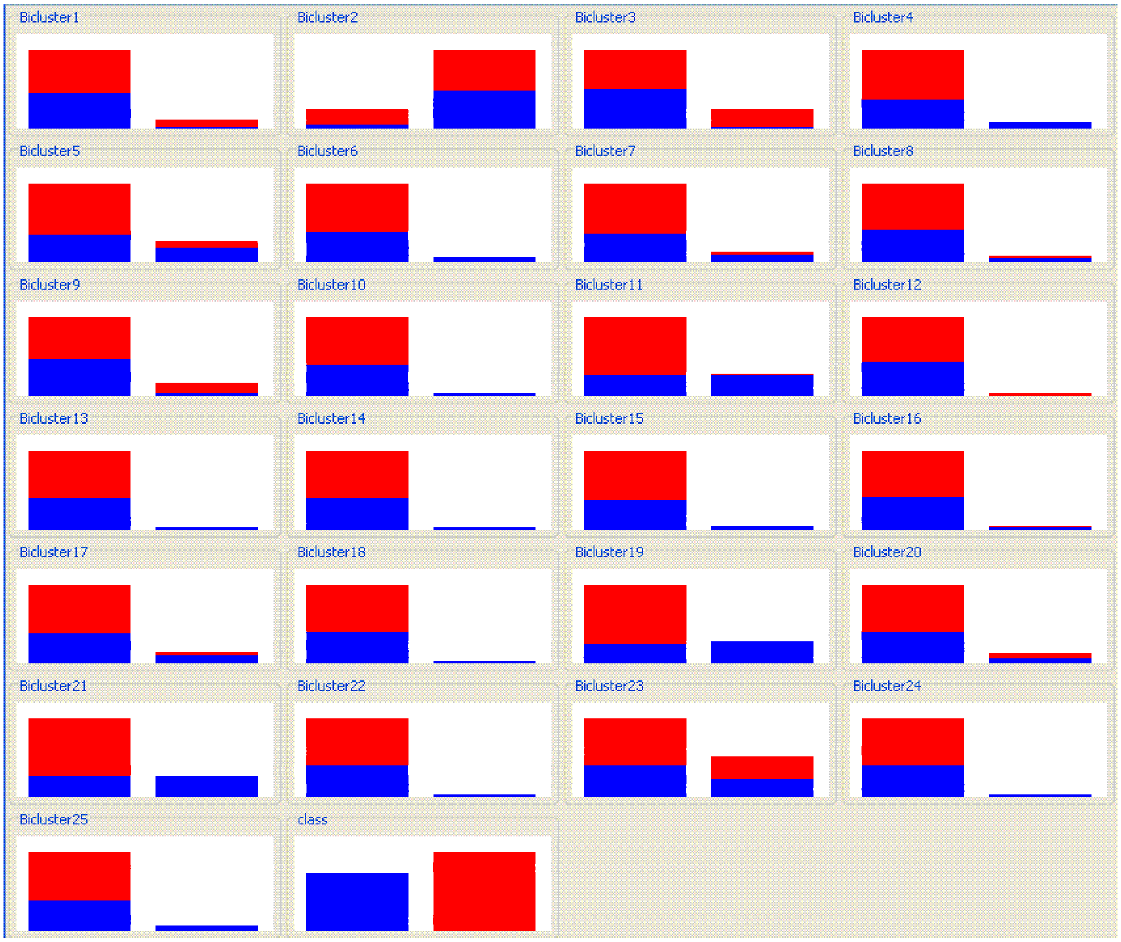


Figure 5.1: Samples and their class in each bi-cluster from RoBic for breast cancer data (screen from Weka). The right column in each bi-cluster shows the samples in that particular bi-cluster. Two different color represent two classes of samples.

higher than our model) for performing 1000 permutation tests on class labels. A permutation test is a type of statistical significance test in which the class labels for the data points will be re-arranged (with the original probability distribution of the labels). Then learning and classification will be done on the new data set (with re-arranged labels). If the labels are exchangeable (the prediction accuracy in the re-arranged data is almost the same as the prediction accuracy for the original data) for many permutation tests, then the likelihood of matching the success of our model by chance is high. For the AML data only 27 of 1000 tests produced models whose accuracy matched or exceeded ours. Therefore the probability that our model classification is by chance is around 0.027.

In Chapter 2, we mentioned that the "shrunken centroid method" [18] is a widely used classification approach for microarrays. We used it to classify the breast cancer data. The best accuracy that we found using this system, 82.89%, was based on a threshold $\delta = 0$.

In the original paper for the prostate cancer (outcome) data [14], the authors obtain 90% accuracy for predicting the outcome. They perform 1000 permutation tests on class labels (recurrence versus non-recurrence), 37 of which yield models whose accuracy matched or exceeded the 90% accuracy of their system. Thus, the likelihood of matching the success of their model by chance alone was estimated by probability 0.037. We ran 1000 permutation tests on our model, using SVM classifier, the maximum accuracy is 90.47%, and only 5 of the tests yields models with 90% accuracy or more. Figure 5.2 shows the histogram of the accuracies. Note that the average prediction accuracy for 1000 permutation test is 56.77% which is the class average.

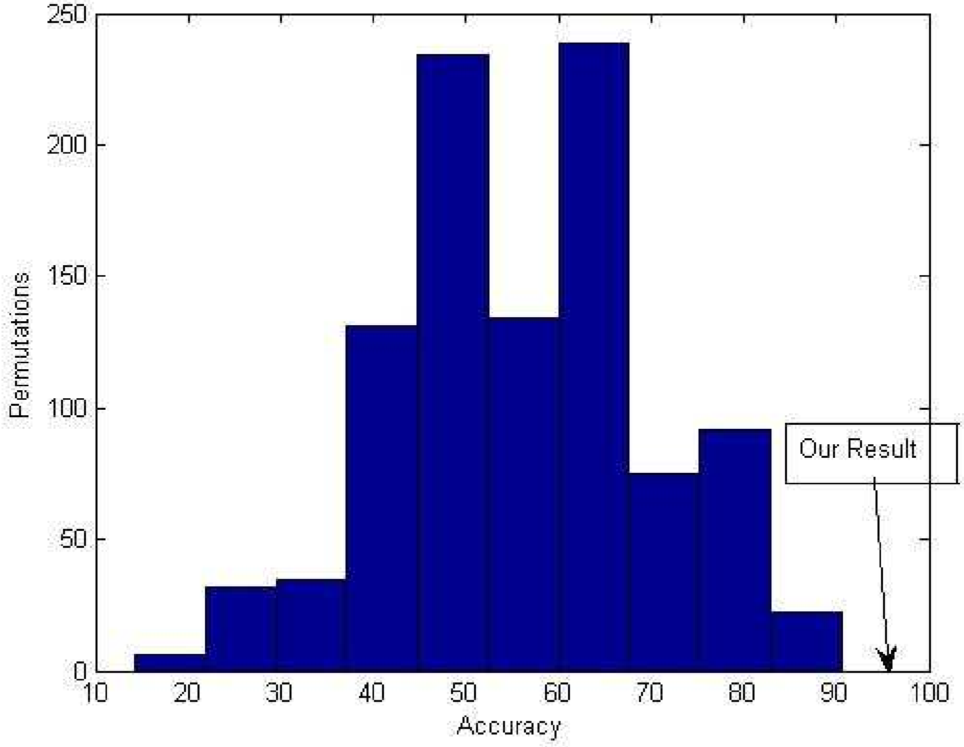


Figure 5.2: Histogram of the accuracies for 1000 permutation tests on the Prostate cancer data.

5.1 Another Hinge Function

For finding the subsets of genes and samples from the α and β vectors, we have tried other hinge functions (see Appendix A for more details). One of them that works very well for the breast cancer data is as follows:

First, approximate the values in the patients vector by a *single* best line, using the absolute values. That is, in $\beta^{(s)}$ (the sorted patients vector), find the best fitting line for $\beta_1^{(s)}$ to $\beta_j^{(s)}$, which has error e_j.Then choose j, the separation point for patients, such that e_j is minimum. Therefore, the patients from index 1 to j are in the current bicluster.Subtract their values from the data matrix. Now calculate α vector from this remainder matrix and choose the subset of genes from this α, with the same function that finds the subset of samples (fit the absolute values into a single best line). Subtract the values of the genes in the current bi-cluster from the matrix and repeat the process on the remainder of the matrix.

In summary, for this version we find the bicluster for the samples *first* and then the genes based on the *absolute values* in vectors, by fitting the values in α and β to *one line*. Applying this approach on the breast cancer data, we get 96.05% prediction accuracy, based on 5-fold cross-validation. The number of bi-clusters that gives the mentioned accuracy is $\{16\}$. In fact, if we apply the in-fold feature selection for the bi-clusters, bi-cluster number #16 alone is sufficient to obtain the mentioned accuracy.

In order to assess this result, we performed the permutation test on the class labels in the dataset based on the bi-clusters. We ran 1000 permutations of the class labels, the maximum accuracy in 1000 tests, using SVM classifier, is 73.68%, and the average over all tests is 55.98%. Figure 5.3 shows the histogram of the accuracies.

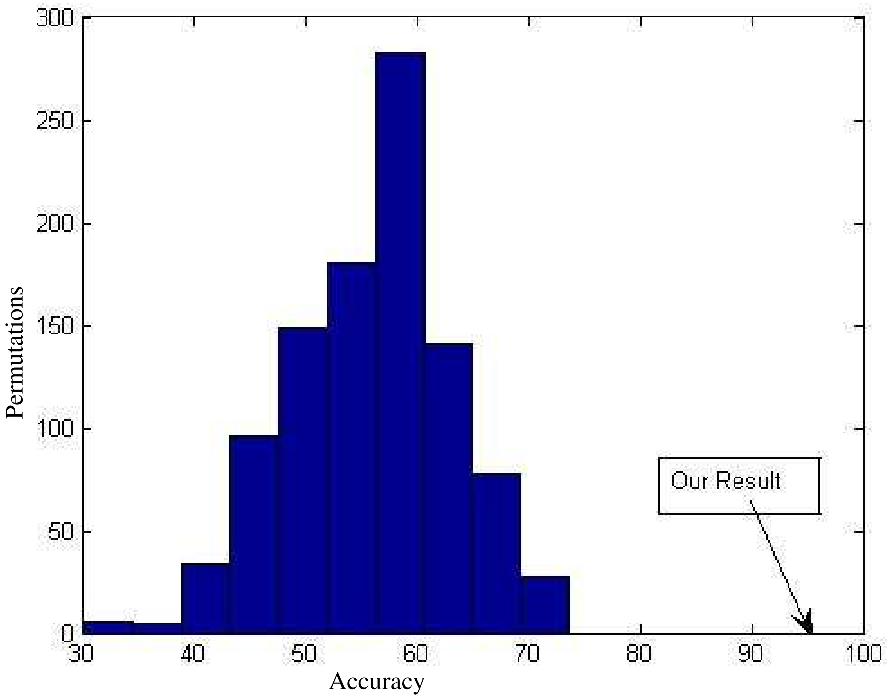


Figure 5.3: Histogram of the accuracies for 1000 permutation tests on the breast cancer data.

5.2 Other Results

In addition to the microarray data, we applied the rank-1 bi-cluster classifier (RoBicC) approach on a metabolite dataset, in which the number of features is far less than microarray data. It contains 53 samples and 72 features. Table 5.3 summarizes the characteristics for the dataset and presents the result.

Number of Samples	53
Number of Features	72
Number of Classes	2
Class 1 (Count)	negative (26)
Class 2 (Count)	positive (27)
Baseline %	50.94%
Best Known Result	71.9%
RoBicC + SVM {Bi-clusters}	86.70% {17}

Table 5.3: Metabolite dataset information and RoBicC prediction accuracy, based on 30 bi-clusters and 5-fold cross-validation.

Chapter 6

Concluding Remarks

6.1 Conclusion

In this thesis, we presented some new results and a survey of the previous results about the clinical outcome prediction for cancer patients, using microarray data. We presented a new bi-clustering algorithm based on the best rank-1 matrix approximation, and a method for sample classification based on the found bi-clusters.

Finding patterns in the gene expression levels helps the biologists to understand the expression level of the genes under different conditions. Therefore, having a robust and rich bi-clustering algorithm to model the complex interactive process in microarray data is very important.

The main goal of this research was to find a method for sample classification by reducing the dimensionality of the data matrix, using *bi-clusters*. We also proposed an algorithm for finding bi-clusters from the microarray data, based on the best rank-1 matrix approximation. We can use the bi-cluster classifier algorithm in order to compare the quality of the bi-clusters obtained from different algorithms. The idea of using bi-clusters to reduce the dimension of the data matrix and predict the samples class, to our knowledge, has not been used by any published approach.

Most of the bi-clusters that we find have the property that most (or all) of the samples in a bi-cluster belongs to a specific class of samples, see Figure 5.1.

In addition to the microarray data, we applied the RoBicC approach on a metabolite dataset. Even though the number of features is very small compared to microarray data, we got better prediction accuracy than the known result. This suggests that RoBicC can be used on other types of datasets to produce an effective classifier.

6.2 Future Work

Appendix A describes many other heuristics that we have tried, but which did not work as well as RoBicC. Here we describe other methods that we would like to do in the future.

- Fit the values in the α vector to a normal distribution (see Figure 6.1). Then choose the subset of genes, $\{1, \ldots, i\}$, whose values are greater than a specific amount based on the mean and variance of the normal distribution. Then choose the subset of samples $\{1, \ldots, j\}$ such that

  $$\|M(1..i, j) - \alpha(1..i).\beta_j\|$$

  is minimum.

  The same approach can be applied to the α vector, but instead of looking at the genes from index $\{1, \ldots, i\}$, we can look at both ends of the α vector. That is, fit two normal distributions to the data and for each of them find genes from index $\{1, \ldots, i_1\}$ and $\{i_2, \ldots, n\}$, whose values are greater than a specific amount based on the mean and variance of the normal distributions.

- Our current bi-cluster classifier, first builds a binary matrix, R, such that each element r_{jk} of this matrix is 1 if the j^{th} sample is in the k^{th} bi-cluster, otherwise it is 0. Instead, we can use continuous values in R, for example, from 1 to 0; where the value 1 means the sample is the first element chosen for this

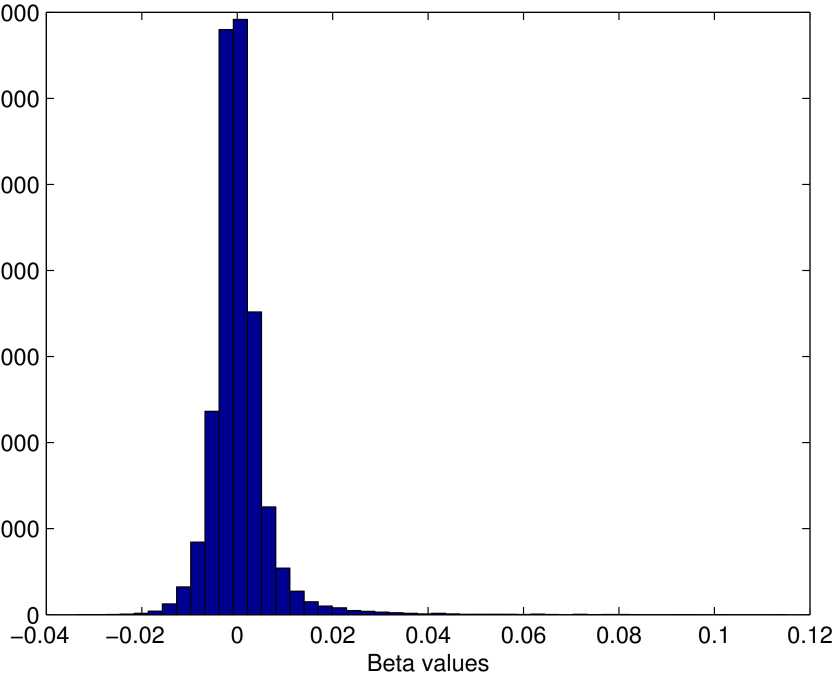


Figure 6.1: Histogram of the β values for breast cancer data.

bi-cluster, $1 - \frac{1}{k}$ means the sample is the second element chosen for this bi-cluster, . . ., and $\frac{1}{k}$ means the sample is the last element, all other elements are 0. To justify this, note that most of the time, the values at the top of the vector correspond to patients of one class and the values at the bottom correspond to the other class. By doing this, we give more weights to the top patients that might be in the same class, and the classifier will use the real values for prediction instead of nominal (0/1) values.

- All the datasets that we use have two classes, because we focused on predicting the binary clinical outcomes, i.e. whether a patient responds to a certain treatment or not. It is also possible to apply the algorithm on multi-class datasets, that have more than two classes of samples, using the same bi-clusters, but perhaps a different classifier. Since we do not use any priori knowledge about the class labels when finding the bi-clusters, we can use the same algorithm for multi-class datasets.

- We used the bi-clusters to predict the samples' class, but they can also be used to predict the gene's class (based on their functionality). The functions of a gene product are the jobs that it does or the "abilities" that it has. These may include transporting things around, binding to things, holding things together and changing one thing into another [34].

- Right now, the algorithm removes all the gene whose expression value is missing for at least one sample. It might be the case that the gene that we are removing is an important and informative gene. We can try to estimate the missing value based on the other known values.

  We also need some way of finding the outliers in data. For example, in Figure 3.5, if we remove the value for the last sample, we might get better results.

Bibliography

[1] http://www.ncbi.nlm.nih.gov/About/primer/microarrays.html

[2] http://www.genome.gov

[3] SARA C. MADEIRA AND ARLINDO L. OLIVEIRA. “Biclustering Algorithms for Biological Data Analysis: A Survey.” *IEEE/ACM Transactions on Computational Biology and Bioinformatics* Vol. 1, No. 1, pp. 24-45. (2004).

[4] AMELA PRELI, STEFAN BLEULER, PHILIP ZIMMERMANN, ANJA WILLE, PETER BHLMANN, WILHELM GRUISSEM, LARS HENNIG, LOTHAR THIELE AND ECKART ZITZLER. “A Systematic Comparison and Evaluation of Biclustering Methods for Gene Expression Data.” *Bioinformatics* Vol. 22, No. 9, pp. 1122-1129. (2006).

[5] http://www.improvedoutcomes.com/docs/WebSiteDocs/Clustering/Clustering_Overview.htm

[6] http://www.cs.ualberta.ca/∼nasimeh/Results/

[7] LAURA LAZZERONI AND ART B. OWEN. “Plaid Models for Gene Expression Data.” *Statistica Sinica* Vol. 12, No. 1, pp. 61-86. (January, 2002).

[8] GEOFFREY J. MCLACHLAN, KIM-ANH DO, CHRISTOPHE AMBROISE. “Analyzing Microarray Gene Expression Data.” *John Wiley and Sons, Inc.* 1^{st} edition. (21 July 2004).

[9] Q. SHENG, Y. MOREAU, AND B.D. MOOR. “Biclustering Microarray Data by Gibbs Sampling.” *Bioinformatics* Vol. 19, Suppl. 2, pp. ii196-ii205. (2003).

[10] Y. KLUGER, R. BASRI, J.T. CHANG, M. GERSTEIN. “Spectral Biclustering of Microarray Data: Coclustering Genes and Conditions.” *Genome Research* Vol. 13, pp. 703-716. (2003).

[11] LAURA J. VAN ’T VEER, HONGYUE DAI, MARC J. VAN DE VIJVER, YUDONG D. HE, AUGUSTINUS A. M. HART, MAO MAO, HANS L. PETERSE, KARIN VAN DER KOOY, MATTHEW J. MARTON, ANKE T. WITTEVEEN, GEORGE J. SCHREIBER, RON M. KERKHOVEN, CHRIS ROBERTS, PETER S. LINSLEY, REN BERNARDS AND STEPHEN H. FRIEND. “Gene expression profiling predicts clinical outcome of breast cancer.” *Nature* No. 415, pp. 530-536. (31 January 2002).

[12] T.R. Golub, D.K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J.P. Mesirov, H. Coller, M.L. Loh, J.R. Downing, M.A. Caligiuri, C.D. Bloomfield, E.S. Lander. "Molecular Classification of Cancer: Class Discovery and Class Prediction by Gene Expression Monitoring." *Science* Vol. 286, pp. 531-537. (15 October 1999).

[13] S.L. Pomeroy, P. Tamayo, M. Gaasenbeek, L.M. Sturla, M. Angelo, M.E. McLaughlin, J.Y. Kim, L.C. Goumnerova, P.M. Black, C. Lau, J.C. Allen, D. Zagzag, J.M. Olson, T. Curran, C. Wetmore, J.A. Biegel, T. Poggio, S. Mukherjee, R. Rifkin, A. Califano, G. Stolovitzky, D.N. Louis, J.P. Mesirov, E.S. Lander, T.R. Golub "Prediction of Central Nervous System Embryonal Tumour Outcome Based on Gene Expression." *Letters to Nature* Vol. 415, pp. 436-442. (24 January 2002).

[14] 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, W.R. Sellers. "Gene Expression Correlates of Clinical Prostate Cancer Behavior". *Cancer Cell* No. 1, pp. 203-209. (March, 2002).

[15] U. Alon, N. Barkai, D.A. Notterman, K. Gish, S. Ybarra, D. Mack, A.J. Levine. "Broad Patterns of Gene Expression Revealed by Clustering Analysis of Tumor and Normal Colon Tissues Probed by Oligonucleotide Arrays." *PNAS* Vol. 96, Issue 12, pp. 6745-6750. (8 June 1999).

[16] Gavin J. Gordon, Roderick V. Jensen, Li-Li Hsiao, Steven R. Gullans, Joshua E. Blumenstock, Sridhar Ramaswamy, William G. Richards, David J. Sugarbaker and Raphael Bueno. "Translation of Microarray Data into Clinically Relevant Cancer Diagnostic Tests Using Gege Expression Ratios in Lung Cancer And Mesothelioma." *Cancer Research* No. 62, pp. 4963-4967. (1 September 2002).

[17] Philip M. Long and Vinsensius Vega. "Boosting and Microarray Data." *Machine Learning* Vol. 52, No. 1-2, pp. 31-44. (2 January 2003).

[18] Robert Tibshirani,, Trevor Hastie, Balasubramanian Narasimhan, and Gilbert Chu. "Diagnosis of Multiple Cancer Types by Shrunken Centroids of Gene Expression." *PNAS* Vol. 99, No. 10, pp. 6567-6572. (14 May 2002).

[19] R.L. SOMORJAI, B. DOLENKO, R. BAUMGARTNER "Class Prediction and Discovery Using Gene Microarray and Proteomics Mass Spectroscopy Data: Curses, Caveats, Cautions." *Bioinformatics* Vol. 19, No. 12, pp. 1484-1491. (2003).

[20] Erinija Pranckeviciene, and Ray Somorjai. "On Classification Models of Gene Expression Microarrays: The Simpler the Better." *2006 International Joint Conference on Neural Networks* pp. 6878-6885. (July 2006).

[21] `http://www.gems-system.org/`

[22] C.J. Wu, S. Kasif. "GEMS: a Web Server for Biclustering Analysis of Expression Data." *Nucleic Acids Research* Vol. 33, Web Server Issue pp. W596-W599 (2005). `http://genomics10.bu.edu/terrence/gems/gems.html`

[23] S. Barkow, S. Bleuler, A. Prelic, P. Zimmermann, and E. Zitzler. "BicAT: a Biclustering Analysis Toolbox." *Bioinformatics* Vol. 22, No. 10, pp. 1282-1283. (2006). http://www.tik.ee.ethz.ch/sop/bicat/index.html

[24] Mining High-Throughput Biological Data. http://www.kdd2006.com/tutorials.html

[25] G.H. Golub, C.F. Van Loan. "Matrix Computations" *The John Hopkins University Press* Second Edition. (1989).

[26] G.W. Stewart. "Introduction to Matrix Computations." *Academic Press Inc*

[27] Steven J. Leon "Linear Algebra With Applications." *Macmillan Publishing Co.* (1980).

[28] Ethem Alpaydin. "Introduction to Machine Learning." *The MIT Press*

[29] http://orcl.gov/hgmis

[30] http://cellbio.utmb.edu/cellbio/ribosome.htm

[31] www.affymetrix.com

[32] http://sdmc.lit.org.sg/GEDatasets/Datasets.html

[33] http://www.cs.waikato.ac.nz/ml/weka/

[34] http://www.geneontology.org/

[35] http://mathworld.wolfram.com/SingularValueDecomposition.html

Appendix A

Other Heuristics

This chapter presents other hinge functions that we have tried for finding the subsets of genes and samples (bi-clusters) from the data matrix. It also describes other approaches for finding the bi-clusters that we have tested.

A.1 Hinge Functions

Given an $n \times p$ matrix M, we find two vectors, α and β, of size $|\alpha| = n$ and $|\beta| = p$ such that $\alpha.\beta^T$ is the least square rank-1 approximation of M. Here, α represents the gene's vector and β represents the sample's vector; see Chapter 3 for more details. Our goal is to find a subset of genes and a subset of samples that have similar patterns in these two vectors. Table A.1 provides prediction accuracy for different models on breast cancer data.

In models 1 - 9, we sort the matrix, M, according to the values in α and β vectors. Here is a brief explanation for these models:

- **Model 1:** In this model, we use the original α and β for matrix M. We increase i from 1 to n for genes (500 at a time, because of the large number of genes that we have in the microarray data), and j from 1 to p to get partial matrix, M_{part}, and partial vectors α_{part} and β_{part}. For each (i, j) we calculate the residual error between M_{part} and $\alpha_{part}.{\beta_{part}}^T$. Therefore, we have:

$$E = M(1..i, 1..j) - \alpha(1..i).\beta(1..j)^T.$$

where E is the residual matrix. We used three different residual error calculations:

- **(a):** the largest eigenvalue of matrix E.
- **(b):** the sum of the square of the elements in the residual matrix E.
- **(c):** the second largest eigenvalue of matrix E.

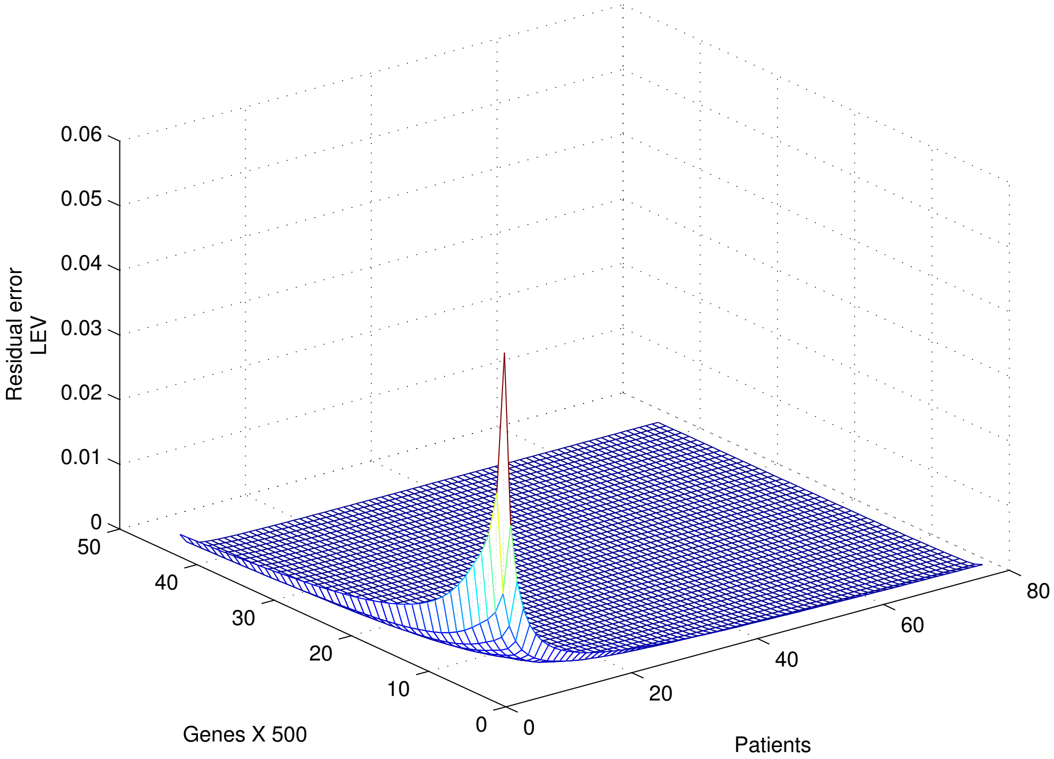


Figure A.1: Residual error (LEV/Matlab norm) for model 1.a.

The goal here was to find some i and j such that the residual error for $i \times j$ sub-matrix is much smaller than for the next sub-matrix ($(i+1) \times j$ or $i \times (j+1)$).

Figure A.1 shows the residual error for breast cancer data using model 1.a (using Matlab norm (LEV)). It is clear from the figure that we can not find such i and j easily.

- **Model 2:** Similar to model 1, we increase i from 1 to n for genes (500 at a time), and j from 1 to p to get partial matrix, M_{part}. But at each step, we calculate the best rank-1 approximation to this matrix, i.e. α'_{part} and β'_{part} for this partial matrix. Then we calculate the residual error at each step. Figure A.2 shows the residual error for breast cancer data using model 2.a (using Matlab norm (LEV)). We could not find such i and j from this figure either.

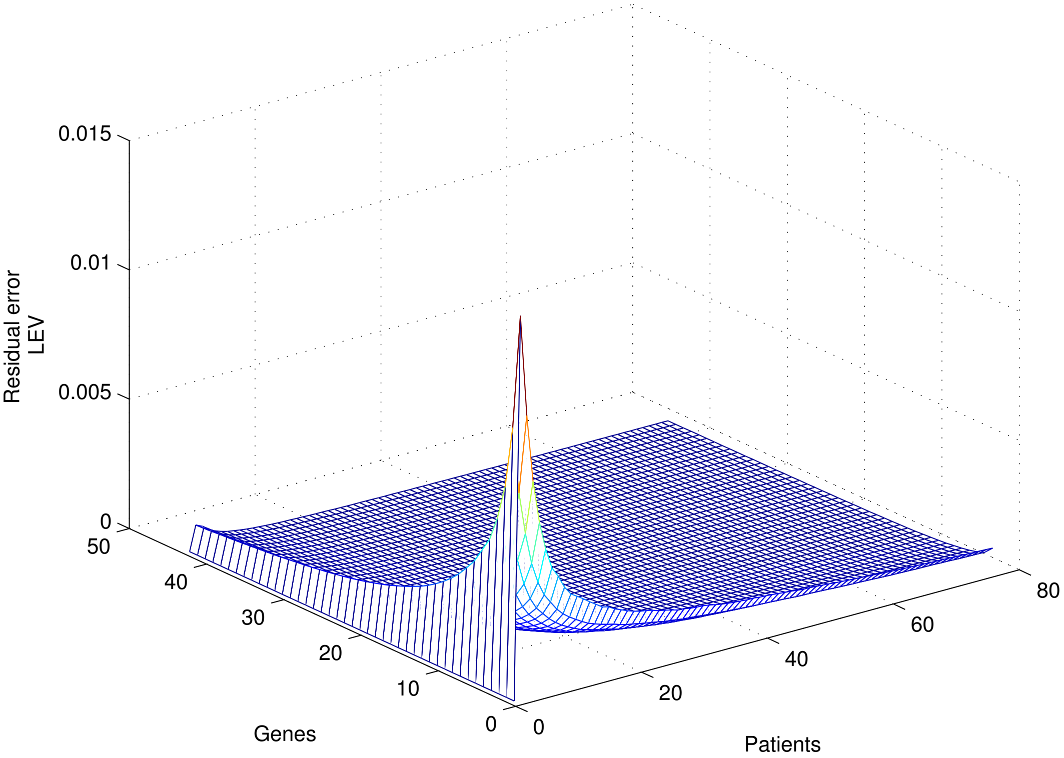


Figure A.2: Residual error (LEV) for model 2.a.

We also considered variations of models 1 and 2, where instead of looking at the largest eigenvalues in the residual matrix, we looked at the second largest eigenvalues in the residual matrix (divided by the size of the matrix). See Figure A.3.

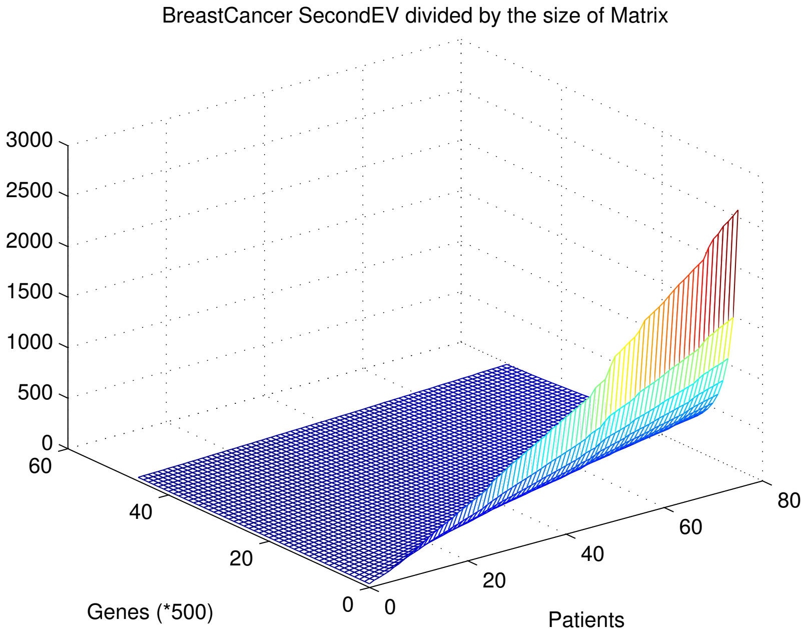


Figure A.3: Residual error, second largest EV for model 1.c.

In the next models, instead of looking at the partial matrices, we looked at the vectors to find the bi-clusters. These approaches are similar to the algorithm in Chapter 3, with different hinge functions for finding the bi-clusters.

- **Model 3:** This model is almost identical to the RoBiC, except that it uses the *absolute values* in α and β vectors, i.e. use $|\alpha^{(s)}|$ and $|\beta^{(s)}|$, for finding the best two lines for $\alpha^{(s)}$ and for $\beta^{(s)}$ that fit into their values.

- **Model 4:** This model is also similar to RoBiC, but it approximates the values in the patients vector and the genes vector by a *single* best line. That is, in $\beta^{(s)}$ (the sorted patients vector), find the best fitting line for values $\beta_1^{(s)}$ to $\beta_j^{(s)}$, which has error e_j.

$$e_j = error(bestlinefor(\beta_1^{(s)}, \ldots, \beta_j^{(s)}).$$

Then choose j, the separation point for patients, such that e_j is minimum. Therefore, the patients from index 1 to j are in the current bicluster, with similar pattern that fits into a single line. Find the genes' separation point, i, using the same procedure.

We also tried this approach by using the absolute values in the vectors as well.

- **Model 5:** In this model, we find the subset of samples *first* and then the subset of genes. We have tried this approach using both exact values and absolute values in α and β. We also tried it by approximating the vectors with two lines or a single line. Here is the details of this model:

  It approximates the values in the patients vector by a *single* best line (or two lines), using the exact values (or absolute values) in α and β. This process gives j, the separation point for patients. Therefore, the patients from index 1 to j are in the current bi-cluster. Subtract their values from the data matrix. Now calculate α vector from this remainder matrix and choose the subset of genes from this α, with the same function that finds the subset of samples. Subtract the values of the genes in the current bi-cluster from the matrix and repeat the process on the remainder of the matrix.

- **Model 6:** This model is very similar to the RoBiC, except that it find the best two lines in α from index 1 to i and from $i + 1$ to $i + 10$. Similarly for β, it finds the best two lines from 1 to j and $j + 1$ to $j + 10$. We did this to try to avoid outliers.

- **Model 7:** This model is almost identical to the RoBiC, except that it uses the the best two *parabolas* that fit into the values in α and β.

- **Model 8:** Find the separation points for genes and samples by finding the average of the top 3 values in each vector. Then choose the genes and samples whose values are higher than half of this average for each vector.

- **Model 9:** Find the separation points for genes, i, by finding the mean and variance (μ and σ) for the *second half* of the genes in α. Then choose the genes, whose values are more than $\mu + 10\sigma$. Find the separation point for samples, j, such that

Model No.	Breast Cancer
3.2.	75%
4.1.(e)	86.84%
4.1.(a)	86.84%
5.1.(e)	67.11%
5.1.(a)	96.05%
5.2.(e)	69.73%
5.2.(a)	69.73%
6.(e)	77.63%
6.(a)	75%
7	78.95%
8	69.74%
9	53.94%

Table A.1: Prediction accuracy for each model on breast cancer data, using a SVM classifier, based on 5-fold cross-validation. The first number in "Model No." is the model number, the second number indicates the number of lines we used to estimate the values in the vectors. The last element is either (e), which means we used the exact values in the vectors for approximation, or (a) which means we used the absolute values in the vectors. Model 1 and 2 are not presented in this table, because we could not find the separation point for patients and genes.

$$\|M(1..i,j) - \alpha(1..i).\beta_j\|$$

is minimum.

Table A.1 shows the results for each model on breast cancer data.

A.2 Other Approaches

The section presents other approaches that we have tested. Table A.2 shows the results for each approach on the breast cancer data.

- **Approach 1:** In the current RoBicC algorithm, we find the bi-clusters using both the training set and the test set. Another possibility is to find the bi-clusters using the training set only. Then for each of the bi-clusters, $B_1, \ldots, B_K$, build a classifier, $C_1, \ldots, C_K$ respectively, based on the expression values of the genes in that bi-cluster. Now, when each new sample comes in (the test set), run each classifiers $C \in \{C_1, \ldots, C_K\}$ on it to determine whether this sample belongs to this bi-cluster. This produces a bit-vector for each new sample. The rest of this model is the same as the bi-cluster classifier in Chapter 4.

- **Approach 2:** This approach is similar to Approach 1 with added feature selection. That is, find the bi-clusters for the training set. Use feature selection to find a subset of genes for each bi-cluster (build the classifiers based on these selected genes). Use these selected genes or the union of the selected genes to predict if a new patient is in each bi-cluster or not.

- **Approach 3:** In RoBicC, we use the bi-clusters to reduce the dimensionality of the data matrix. Instead, we can use the union of the genes in the found bi-clusters to reduce the dimensionality. That is, we learn a classifier based only on these genes in the union of all bi-clusters.

- **Approach 4 (Compare to PCA (Principal Component Analysis)):** In this approach, we find K vectors for genes, for some user specific $K \in \mathbb{N}$ and $K \leq min(n, p)$. Then we reduce the dimension of the data matrix, by mapping the data matrix based on the genes' vector. That is,

$$[U, S, V] = SVD(M)$$

  - The data matrix, M, is $n \times p$
  - U is $n \times K$ with orthonormal columns,
  - S is a diagonal matrix of eigenvalues in decreasing order and size $K \times K$,
  - V is $p \times K$ with orthonormal columns.

Therefore, $M_{mapped} = M^T.U$ is a $p \times K$ matrix. We reduced the dimension of the data matrix from $n \times p$ to $p \times K$. Therefore, instead of having n features for each sample, we have K features. Using 5-fold cross-validation, we split the data into training set and test set. Now for each fold, we can learn a classifier using the training set and K features, then predict the class labels for the test set.

- **Approach 5:** In this approach we use SVD (Singular Value Decomposition) [35] to find K eigenvectors for genes and samples at the same time (instead of finding them one at a time from the residual matrix). Then we find the subset of genes and samples from each of these vectors by fitting the two best lines to the values in vectors. The result here is different from the RoBicC, because the bi-clusters that RoBicC finds can have overlaps in both genes and samples, when finding the k^{th} bi-cluster, it considers data matrix that has already subtracted away previous $k-1$ bi-clusters. But this approach does not consider the interactions between bi-clusters, i.e. it does not subtract away the values for genes and patients in each bi-cluster.

- **Approach 6:** For finding the subset of samples, instead of looking at the elements at the beginning of the vector, look at the elements at the *start* and *end* of the vector. That is, find samples from index $\{1, \ldots, j_1\}$ and $\{j_2, \ldots, p\}$ in α (by best line approximation, for example). As mentioned earlier, when we find the β vector and sort it, most of the values at the beginning of the sorted vector corresponds to a specific class of samples, and (most of) the values at the end of the sorted vector corresponds to the other class of samples.

  Then we can build the classifier matrix, R, such that each element r_{ij} of the matrix is:

  - 1 if the j^{th} sample is in the k^{th} bi-cluster, and $j \in \{1, \ldots, j_1\}$.
  - -1 if the j^{th} sample is in the k^{th} bi-cluster, and $j \in \{j_2, \ldots, p\}$
  - Otherwise, it is 0.

We have tested this approach on the breast cancer data, using only 3 biclusters, we get the prediction accuracy of 76%, which is better than the best known published result.

Approach No.	Breast Cancer
1	56.25%
2	73.75%
3	67.11%
4	59.21%
5	60.53%
6	76%
7	65.79%

Table A.2: Prediction accuracy for each approach on breast cancer data, using a SVM classifier based on 5-fold cross-validation.

- **Approach 7:** In addition to all of these approaches, we tried 1-dimensional clustering algorithms combined with classification algorithms, too. The best prediction accuracy that we found for the breast cancer data is 65.79%, using K-mean clustering ($K = 2$).

Appendix B

Cross Cancer Institute Data

This project was started with a microarray data from the Cross Cancer Institute. They were investigating the in-vivo effectiveness of a specific chemotherapy on ovarian cancer. To do this, they recruited a cohort of ovarian cancer patients, and collected two ascites samples from each – one taken before chemotherapy, and one after. They then ran microarrays on each of these samples, each producing expression levels for around $53,000$ genes. They currently have data for 9 patients, 5 of whom responded favorably to the chemotherapy.

They wanted to characterize the differences in expression level between:

1. pre- vs post- chemotherapy ascites samples;
   e.g., seeking the genes whose expression level changed for all patients.

2. responders and non-responders;
   e.g., seeking some patterns that indicate a specified subset of genes that are interrelated for a subset of patients.

Before analyzing the data, we had to deal with the missing values in the data. We removed the genes that their expression values were missing for at least one patient. Then, the first task was done by comparing the differences between the expression values before and after treatment for all the patients. Here we found a set of $5,686$ genes were found whose expression values changed for more than one patient.

For the second task, we built the data matrix based on the differences in gene expression levels before and after the treatment. Then we found the α and β vectors for this data matrix. The top 5 values in β vector corresponds to the responder patients and the bottom 4 values corresponds to the non-responders patients.

But the number of samples is very small in this dataset, therefore, we used some publicly available microarray datasets in order to verify both the bi-clustering algorithm and the bi-cluster classifier.