journal homepage: www.intl.elsevierhealth.com/journals/cmpb

# Microarray-MD: A system for exploratory analysis of microarray gene expression data

*D.E. Maroulis*[a,*], *I.N. Flaounas*[a], *D.K. Iakovidis*[a], *S.A. Karkanis*[b]

[a] *Real-Time Systems & Image Analysis Group, Department of Informatics and Telecommunication, University of Athens, Panepistimiopolis, Ilisia, 15784 Athens, Greece*

[b] *Department of Informatics and Computer Technology, Lamia Institute of Technology, 3rd Kilometer, Old National Road, 35100 Lamia, Greece*

ARTICLE INFO

*Article history:*
Received 17 January 2005
Received in revised form 30 May 2006
Accepted 8 June 2006

*Keywords:*
Biomedical system
DNA Microarrays
Gene selection
SVM

ABSTRACT

In this paper, we present Microarray Medical Data explorer (Microarray-MD), a novel software system that is able to assist in the exploratory analysis of gene expression microarray data. It implements a combination scheme of multiple Support Vector Machines, which integrates a variety of gene selection criteria and allows for the discrimination of multiple diseases or subtypes of a disease. The system can be trained and automatically tune its parameters with the provision of pathologically characterized gene expression data to its input. Given a set of new, uncharacterized, patient's data as input, it outputs a decision on the type or the subtype of a disease. A graphical user interface provides easy access to the system operations and direct adjustment of its parameters. It has been tested on various publicly available datasets. The overall accuracy it achieves was estimated to exceed 90%.



## 1. Introduction

DNA microarray technology is the premier tool for the study of gene expression on a genomic scale. Scientists seeking to harness the potential of this technique are often challenged by the prodigious quantities of data produced. Well-designed, user-friendly software is the key to tracking, integrating, qualifying, and ultimately deriving scientific insight from experimental results. A variety of software systems has been developed to assist researchers in their attempt to tackle several microarray related problems ranging from the simple gene expression levels normalization to the modelling of biomolecular network graphs.

Such software systems have been implemented since the early beginnings of the millennium. Do et al. [1] proposed the GeneClust software for microarray data analysis which implements hierarchical clustering and gene shaving algorithms [2]. Li and Wong [3] proposed the dChip software which implements a model-based expression analysis of oligonucleotide arrays and several high-level analysis procedures, including comparative analysis and hierarchical clustering. Peterson [4] proposed Clusfavor, a software package oriented in unsupervised analysis of microarrays. A powerful software suite named Genesis has been developed by Sturn et al. [5] for large-scale gene expression analysis. It includes filters, normalization and visualization tools, distance measures as well as clustering and classification algorithms such as hierarchical clustering, self-organizing maps, *k*-means, principal component analysis, and Support Vector Machines (SVMs). Colantuoni et al. [6] developed a web-based tool named Snomad for the standardization and normalization of DNA microarray data, using two non-linear transformations which correct both bias and variance of microarray element signal intensities. Saal et al. [7] developed Base, a software system for

* *Corresponding author*. Tel.: +30 210 7275307; fax: +30 210 7275333.
E-mail address: rtsimage@di.uoa.gr (D.E. Maroulis).

0169-2607/$ – see front matter © 2006 Elsevier Ireland Ltd. All rights reserved.
doi:10.1016/j.cmpb.2006.06.008

the management of biomaterial information, raw data and images, which provides integrated and "plug-in"-able normalization, data viewing and analysis tools. An open source suite of tools named TM4 has been developed by Saeed et al. [8]. It consists of four major applications, namely Microarray Data Manager (MADAM) which is a data entry and management tool for DNA microarray experiments, TIGR_Spotfinder which is a semi-automated image analysis software, Microarray Data Analysis System (MIDAS) which is used for data normalization and filtering, and Multiexperiment Viewer (MeV) which is a data mining tool that implements a variety of clustering algorithms. Another software suite provided in open source is Bioconductor. It comprises of several packages that provide innovative tools for the analysis and comprehension of genomic data [9]. Su et al. developed RankGene, a software system which integrates a variety of popular ranking criteria, ranging from the traditional t-statistic to the one-dimensional SVMs [10]. A minimum spanning tree representation of gene expression data is being exploited by Excavator, a software system for DNA microarray data clustering [11]. Toyoda and Konagaya [12] developed KnowledgeEditor, a graphical aid for biologists on biomolecular network modelling. Recently Pieler et al. proposed ArrayNorm a versatile and platform-independent application for the visualization, normalization and statistical identification of genes with significant changes in expression [13].

The afore-mentioned software systems basically address experienced bioinformatics specialists rather than specialists in medicine and biology as they involve complicated operations and adjustment of parameters. Moreover the extravagant cost of both DNA microarrays and the necessary laboratory equipment as well as the high degree of variability between the different experiments, makes it quite uncommon nowadays to use such systems for diagnostic purposes. These reasons explain the current trend for highly specialized research software. However, researchers have suggested that within this decade microarrays will routinely be used for medical diagnosis by physicians [14].

In this paper, we propose Microarray Medical Data explorer (Microarray-MD), a novel software system that is able to assist in the exploratory analysis of gene expression microarray data, by class prediction. It implements a cascading architecture of SVMs which integrates a variety of gene selection criteria and allows for the discrimination of multiple classes that correspond to multiple diseases or subtypes of a disease. SVMs are binary classification algorithms that have been originally proposed by Vapnik [15] in the framework of structural risk minimization. The studies of Brown et al. [16] and Furey et al. [17] advocate to that SVMs are advantageous over other classification methods for microarray data, as they are remarkably robust, their performance is not easily affected by sparse or noisy data, and they resist to overfitting and to the "curse of dimensionality" [18]. Moreover preliminary research of our group [19,20] has shown that the cascading architecture of SVMs adopted is capable of achieving comparable and, in many cases, higher classification accuracy than other relevant approaches that have been proposed in the literature. Given a set of new, uncharacterized, patient's data as input, the system outputs a decision on the type or the subtype of a disease. The proposed system offers a graphical user interface (GUI) which combines simplicity with practicality and enables users to easily understand its operations and exploit its capabilities to the greatest extent.

The rest of this paper is organized as follows: in Section 2, a short background on DNA microarrays is provided. The Microarray-MD system and the methods it implements are described in Section 3. In Section 4 the GUI of the system is presented. In Section 5, we appose example runs and results of its application and in the last section the conclusions of this study are summarized.

## 2. DNA microarray data

A DNA microarray consists of a collection of nucleic acid sequences immobilized onto some solid support so that each unique sequence forms a tiny feature, called a *spot* or *probe* [21]. The spots are organized in an orderly and consistent way, as their location is used to identify the particular gene sequences. A glass slide or glass wafer acts as solid support onto which up to tens of thousands of spots can be arrayed in a total area of a few square centimetres.

An mRNA sample of a test subject called *target* is prepared and labelled using fluorescent dye or radioactive marker. To be able to compare a large number of samples a reference RNA which is used in all experiments is labelled with a different fluorescent dye. The use of different fluorescent dyes allows mRNAs from the two different cell populations, the sample and the reference, to be mixed and hybridized to the same array, which results in competitive binding of the target to the arrayed sequences. After hybridization and washing, the slide is laser-scanned using two different wavelengths, corresponding to the dyes used. The intensity of the same spot in both channels is measured with special image analysis software and the ratio of these channel intensities is used for the computation of the gene expression levels [22].

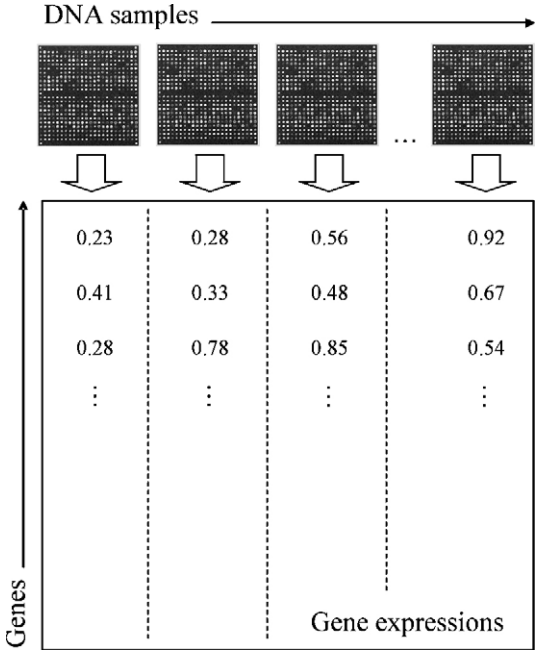



**Fig. 1 – Arrangement of a gene expression matrix.**

The set of gene expression measurements acquired from a single microarray experiment can be considered as a large feature vector. Low quality gene expression measurements may result in the appearance of missing values within these vectors [23]. Results of multiple microarray experiments using identical probes for different samples lead to the construction of the so-called *gene expression matrix*. This matrix consists of rows that correspond to different genes and of columns that correspond to different samples (Fig. 1). However, for efficient gene expression analysis, the establishment of standard DNA microarray protocols and laboratory methods is required to make the different samples comparable before they become parts of the same gene expression matrix [24].

## 3. System description

Microarray-MD is a system capable of "learning" to recognize the pathology of samples provided to its input through a supervised training procedure. The block diagram of Microarray-MD is illustrated in Fig. 2. It includes two processing units, a Pre-processing and a Decision Unit. The Pre-processing Unit prepares the gene expression data to passing into the Decision Unit, which is the main processing unit of the proposed system.

The user may switch between two modes of operation: the training and the testing mode. The training mode of operation requires a gene expression matrix of pathologically characterized samples as input. During training the system organizes its internal structure and tunes its pre-processing and classification parameters for a given medical problem. These parameters are then stored for use during the testing mode of operation. Given a patient's gene expression vector, the trained system is able to classify it is based on prior knowledge that has been encoded in the stored training parameters.

### 3.1. *Pre-processing Unit*

The Pre-processing Unit handles the management of missing values as well as the normalization of the gene expression levels. Poor quality in the preparation of the mRNA targets contributes to low quality gene expression measurements, as

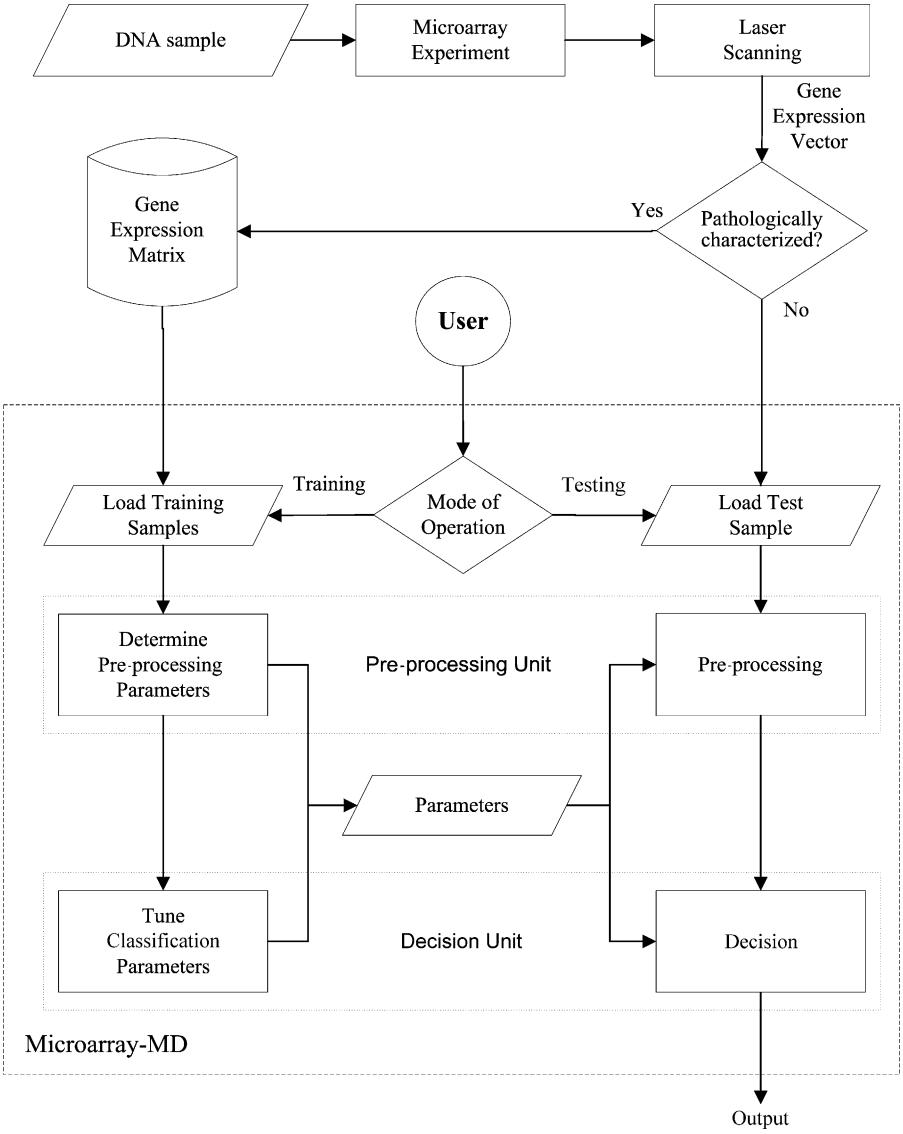



**Fig. 2 – Block diagram of the Microarray-MD system.**

it affects the mean values and the standard deviation of the intensities of the spots, their size and their contrast to the local background areas [23]. These low quality measurements are usually discarded and missing values appear.

A straightforward approach when dealing with samples containing missing values would be to discard them [18]. Unfortunately, most DNA microarray datasets consist of a very limited number of samples and thus it would be a luxury to drop available data. Therefore, a variety of methods have been reported in the literature for handling missing gene expression measurements. Most of these methods suggest that the missing values should be replaced by others deriving from the rest of the available data set. These include simple approaches such as the replacement of the missing values with the row-average of the gene expression matrix [23] or more sophisticated imputation methods based on $k$-nearest neighbours [25], singular value decomposition [25] and the Bayesian principal components analysis [26]. In the Pre-processing Unit of Microarray-MD we have incorporated (a) the row-average method, as it is simple and effective [23] and (b) the $k$-nearest neighbours method ($k$-NN) which is more robust than the row-average method but requires more computations [25].

In addition to the estimation of missing values, the Pre-processing Unit incorporates data normalization methods which aim to the adjustment of the gene expression levels so that meaningful biological comparisons between different DNA microarray experiments can be made. There are a number of reasons why data should be normalized, including differences in the amounts of targets hybridized in the array, and differences in the gains of the microarray during the scanning process [23]. In our implementation two normalization methods, widely used in the literature, have been included. The first method normalizes the gene expression levels of each sample to conform to zero mean and unitary variance [23]. The second method normalizes the gene expression levels by subtracting its median and by dividing the result by its quartile range (the difference between the first and the third quartiles). The median and quartile range are more robust estimators for the center and the dispersion of a distribution respectively [27].

### 3.2. *Decision Unit*

The Decision Unit handles medical problems as multi-class classification problems. It is capable of classifying the input gene expression vectors to $N$ classes noted as $\omega_i$, $i=1, 2, \ldots N$. Each class corresponds to samples acquired from healthy patients, from patients suffering from the same disease or from patients suffering from a subtype of a particular disease. It comprises of $N-1$ cascading blocks $B_j$, $j=1, 2, \ldots N-1$ as illustrated in Fig. 3.

Each block consists of a gene selection module $S_j$ and a classification module $C_j$. Module $S_j$ uses the output of the Pre-processing Unit as input. Module $C_j$ is autonomously trained with a subset $X_j$ of the available training samples $X$, where $X_j$ is defined as

$$X_j = \{x \in (\omega_j \cup \omega_h)\}, \quad \omega_h = \bigcup_{p=j+1}^{N} \omega_p \tag{1}$$

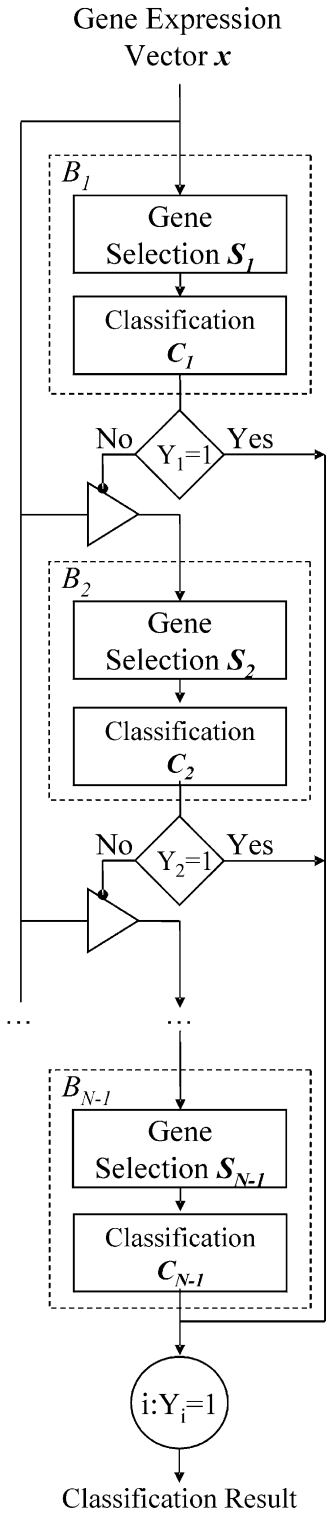



**Fig. 3 – The Decision Unit.**

Module $S_j$ selects a subset of $\tau_j$ gene expression measurements which best discriminates class $\omega_j$ from class $\omega_h$ and maximizes the classification performance of the module $C_j$. Given a test vector $x$, the module $C_j$ is fed with $\tau_j$ gene expression measurements and outputs 1, if $x \in \omega_j$, or $-1$, if $x \notin \omega_j$. If $x \notin \omega_j$, the next block $B_{j+1}$ will be activated to classify the test vector using the corresponding $\tau_{j+1}$ gene expression mea-

**Table 1 – Gene ranking criteria**

| | |
|---|---|
| Prediction strength [30] | $Z(g)=\frac{m_g^j-m_g^h}{\sigma_g^j+\sigma_g^h}$ |
| Welch's $t$-test [27] | $Z(g)=\frac{m_g^j-m_g^h}{\sqrt{(\sigma_g^j)^2/n_j+(\sigma_g^h)^2/n_h}}$ |
| Sun's et al. [31] criterion | $Z(g)=\frac{n_j(m_g^j-m_g)^2+n_h(m_g^h-m_g)^2}{\sum_{i\in\omega_j}(x_{gi}-m_g^j)^2+\sum_{i\in\omega_h}(x_{gi}-m_g^h)^2}$ |

surements. Otherwise the classification task terminates and $x$ is assigned to class $\omega_j$. The last block $B_{N-1}$ decides whether $x\in\omega_{N-1}$ or $x\in\omega_N$.

### 3.2.1. *Gene selection modules*

Among the gene selection methods that have been proposed in the literature, the statistical gene-ranking techniques are of particular interest as they are less computationally demanding than wrapper or embedded techniques [28,29]. The gene selection modules of the Decision Unit integrate three ranking criteria for the selection of differentially expressed genes (Table 1). Golub et al. [30], Pan [27] and Sun and Xiong [31] have shown that these criteria can be efficiently used for the identification of differentially expressed genes. These criteria suggest that the genes are ranked in descending order based on the absolute value of the $Z(g)$ statistic for each gene $g$.

The $(m_g^j, \sigma_g^j)$ and $(m_g^h, \sigma_g^h)$ correspond to the mean and standard deviation of the expression levels of the gene $g$ for the training samples that belong to $\omega_j$ and $\omega_h$ classes respectively and $m_g$ is the mean expression level of gene $g$ for the entire training set. The $x_{gi}$ is the $(g, i)$ element of the gene expression matrix that corresponds to the expression level of gene $g$ for the sample i. The number of samples belonging to each of the above classes is denoted by $n_j$ and $n_h$. The $\tau_j$ top-ranked genes are selected as they lead to a large between-class distance and a small within-class variance.

### 3.2.2. *Classification modules*

The classification module of each block of the Decision Unit implements a binary SVM classifier. SVM training involves a quadratic programming optimization procedure which aims to the identification of a subset of important vectors from the training set, called *support vectors*. These vectors are utilized for the drawing of a separating hypersurface between the two classes. In summary this algorithm proceeds as follows.

Let $I$ be an input space of vectors $x_i$, $i=1, 2, \ldots n$, distributed to two classes, labelled as $y_i\in\{-1, 1\}$. Considering $\Phi$ as a non-linear mapping from the input space $I\subseteq\Re^\nu$ to a Euclidean space H, the training results in finding a hypersurface defined by the equation

$$w\Phi(x)+w_0=0 \tag{2}$$

so that the *margin of separation* between the two classes is maximized. It is easy to prove [15,32] that for the *maximal margin* hypersurface,

$$w=\sum_{i=1}^{n}\lambda_i y_i\Phi^{\mathrm{T}}(x_i) \tag{3}$$

and $w_0$ is estimated from the Karush-Kuhn-Tucker *complementarity condition* [32]. The variables $\lambda_i$ are Lagrange multipliers which are estimated by maximizing the Lagrangian

$$L_D=\sum_{i=1}^{n}\lambda_i-\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\lambda_i\lambda_j y_i y_j K(x_i, x_j) \tag{4}$$

with respect to $\lambda_i$. The vectors $x_i$ for which $0<\lambda_i\leq c$, are called *support vectors* and $c$ is a positive cost parameter. As $c$ increases a higher penalty for errors is assigned.

The function $K(x_i, x_j)$, known as *kernel function*, is defined as the inner product

$$K(x_i, x_j)=\Phi^{\mathrm{T}}(x_i)\Phi(x_j) \tag{5}$$

and should satisfy Mercer's condition [15].

Most commonly used kernel functions are the linear $K(x_i, x_j)=x_i\cdot x_j$, the polynomial $K(x_i, x_j)=(\gamma\cdot x_i\cdot x_j+1)^p$ of second ($p=2$) and third order ($p=3$), and the radial basis functions (RBF) $K(x_i, x_j)=e^{-\left\|x_i-x_j\right\|^2/\gamma}$, where $\gamma$ is a strictly positive constant. Apparently the linear kernel is less complex than the polynomial and the RBF kernels. The RBF kernel usually has better boundary response as it allows for extrapolation, and most high-dimensional data sets can be approximated by Gaussian-like distributions similar to those used by RBF networks [32]. The appropriate kernel function, the $c$ and $\gamma$ parameters for a given classification problem are automatically determined by grid search aiming at the minimization of the leave-one-out cross validation classification error [34], i.e. systematic search for the optimal set of parameter values by testing all possible combinations of values in a given discretized parameter space. Leave-one-out cross validation is an error estimation method where the classifier is trained using $(n-1)$ samples and evaluated on the one remaining sample; this is repeated $n$ times with different training sets of size $(n-1)$ [18,33]. This method leads to an almost unbiased estimate of the classification error probability, if gene selection and parameter tuning take place on the $(n-1)$ samples during each iteration. So, for each iteration of the leave-one-out procedure, the $\tau_j$ selected genes and the classification parameters are retuned using the training set of the nested leave-one-out cross validation iteration, for each block of the Decision Unit [35,36].

The hypersurface separating of the two classes can be finally derived by the following equation:

$$\sum_{\forall i:0<\lambda_i\leq c}\lambda_i y_i K(x_i, x)+w_0=0 \tag{6}$$

In the testing mode of operation, given a test vector $x$, the trained SVM outputs a label Y:

$$Y=\mathrm{sign}\left(\sum_{\forall i:0<\lambda_i\leq c}\lambda_i y_i K(x_i, x)+w_0\right) \tag{7}$$

which designates the class the $x$ belongs to. This information is used as a clue for the final decision.

## 4. Graphical user interface

The development of a user interface should take into account user requirements that can be determined by user needs and the tasks the system is intended to support [37,38]. Compatibility between the users' understanding of the system and their skill or knowledge can be achieved by a user interface design which takes into account the user's level of skill or knowledge, the functionality of user tasks, their important procedural characteristics and the type of information use. The user interface of Microarray-MD has been designed mainly for scientists specialized in the field of medicine and biology. Permitting a level-structured approach to learning [39], novice users can be taught a minimal subset of objects and actions with which to get started and progressively expand their potential to more complex tasks. On the other hand, users having strong knowledge of the supported tasks and interface concepts can show rapid progress, work faster and soon learn how to take advantage of most options provided by the system.

At the beginning of the program, the user is prompted to choose between the two operating modes of the system. On user's response, a window associated with the corresponding mode of operation is opened: the Training Window, for the training mode and the Testing Window for the testing mode. Moreover in the case of the training mode, which involves more complex options than the testing mode, novice users are offered a Wizard interface which can guide them to go through the configuration of the training parameters and the initiation of the training procedure.

### 4.1. *The Training Window and the Wizard interface*

The Training Window is illustrated in Fig. 4. It consists of three input panels (Panel-1, Panel-2 and Panel-3), each of which can be used to select certain options and two output panels (Panel-4 and Panel-5) facilitating the presentation of the training results and the current status of the application.

Panel-1 is provided for the management of input/ouput operations. The user can designate the location of an input file containing the desired gene expression matrix (GEM) training data. These files can be directly opened or they can be constructed by combining several gene expression vectors encoded in standard file formats supported by the GenePix Pro DNA microarray image analysis software. The GEM file format is compatible with the tab-delimited pre-clustering file format (pcl) supported by the Stanford Microarray Database [40]. Moreover the user is provided with options to modify GEM files by inserting or removing samples. A sub-panel located on the right of the first panel has been assigned to save or load the training parameters of the system.

Once a GEM file is loaded, Panels-2 and 3 are activated. Panel-2 contains graphical controls for the specification of the pre-processing parameters. The user may choose between the row-average and the $k$-NN methods for the imputation of missing values, and between the mean/variance and median/quartile range - based normalization methods, described in Section 3.1.

Panel-3 contains graphical controls for the specification of the classification parameters. The various classes as well as the distribution of the samples involved in the medical problem the system is intended to solve, are apposed in a list-box control. The ordering of the classes corresponds to the ordering of the blocks of the cascading architecture. Two arrow-buttons allow for the users to reorder the classification blocks.

The gene selection sub-panel located underneath the list-box control serve for the selection of a gene ranking criterion, as described in Section 3.2.1, and the range of top-ranked genes to be tested as inputs to the classification modules of the cascading architecture. The system will incrementally search this range to identify a single subset of top-ranked genes that maximizes the classification performance. For example, if the range is set to 1–15 genes, the system will test fifteen sets of top-ranked genes, namely gene (1), genes (1, 2), . . ., genes (1, 2, 3, . . .15). The classification parameters can be adjusted by the sub-panel located next to the gene selection sub-panel.

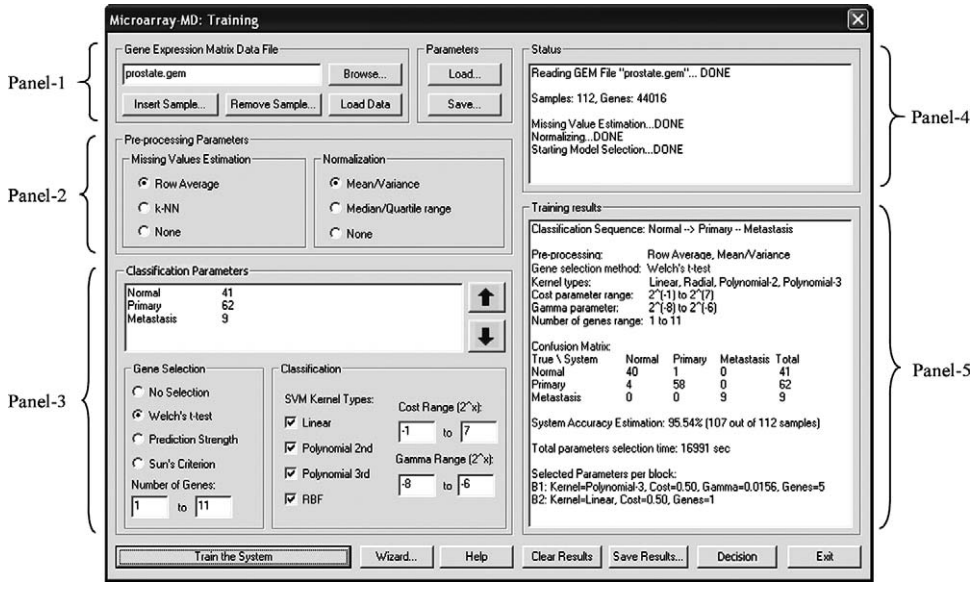



**Fig. 4 – The Training Window.**

The user may select up to four different kernel functions to be tested to minimize the leave-one-out cross validation error in each classification module and define the search ranges for the $c$ and $\gamma$ parameters.

On the upper right side of the Training Window, Panel-4 provides information on the status of the application, e.g. loading, training, etc., information related to the open GEM file, such as the total number of samples and the dimension of the gene expression vectors.

As soon as the required parameters are set the user may initiate the training process. During the training procedure the system searches within the ranges of the parameter values set by the user, to automatically determine the optimal ones for a particular medical problem. Therefore, the user needs not be concerned with the technical details of the different possible training options. After the training finishes, the results are printed in Panel-5 and can be saved for archiving purposes. The results include the classification performance of the system presented by means of confusion matrix and average accuracy, and the optimal system configuration details as these occur by the almost unbiased leave-one-out parameter tuning process mentioned.

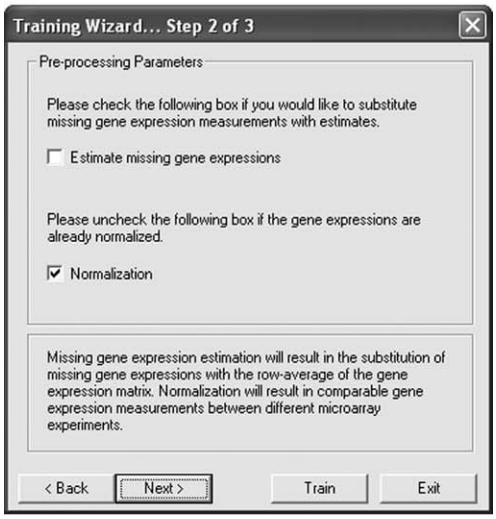
Novice users are always provided with the option of training Microarray-MD system by using the Wizard interface (Fig. 5). The Wizard provides a step-by-step interactive process accompanied by helpful information, and allows only for the selection of key options, such as: which file to use for training, to apply or not to apply pre-processing, to select or not to select a subset of differentially expressed genes in the classification process. The purpose of these options as well as the default parameters used, are described in the help legends appearing at the bottom of the Wizard's dialogs. The most effective parameters in most of the medical problems tested have been considered as default. These include the row-average missing values imputation method, the mean/variance normalization method and the Welch's $t$-test ranking criterion. For gene selection the Wizard assumes a search range of one to a number of top-ranked genes, which equals to one tenth of the total number of the available samples in order to avoid performance degradation phenomena attributed to the "curse of dimensionality" [44].

### 4.2. *The Testing Window*

Given a new patient's gene expression data, a decision on the class they belong to, can be made through the Testing Window illustrated in Fig. 6. Testing the system is much simpler than training it for the user, as the Testing Window requires only two filenames as input. The first filename corresponds to the file containing the system parameters produced as a result of the training process. This file also contains information that can be used to identify and retrieve the names of the genes selected by the gene selection module of each block. The second filename corresponds to the gene expression data of one or more patients as these are quantified by means of a DNA microarray image analysis software. This file should conform to the formats supported by the GenePix Pro software or to the GEM file format.

The user may proceed to the classification of the input gene expression data by just clicking on the "Decision" but-


(a)




(b)


(c)

**Fig. 5 – The Wizard dialogs for the inquisition of (a) input, (b) pre-processing and (c) classification parameters.**

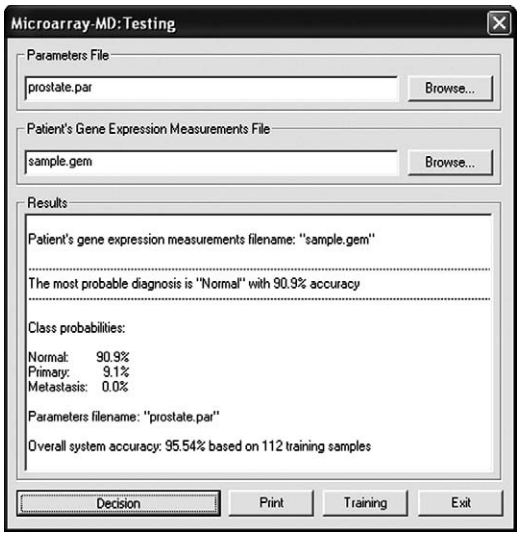



**Fig. 6 – The Testing Window.**

ton. This action triggers the retrieval of the pre-processing and the classification parameters from the parameters file. Then the internal structure of the Decision Unit is automatically determined and the system is fed with the patient's uncharacterized gene expressions as these are loaded from the gene expression measurements file. The results are printed in the Results panel of the Testing Window. These include the decision made by the system as well as the probabilities of the input sample to belong to each of the classes the system was trained on. These probabilities are based on the confusion matrix obtained during the training process.

## 5. Results

In this section, we present the results of experiments performed using the Microarray-MD system with prostate and colon cancer gene expression data.

### 5.1. *Experiments with prostate cancer data*

The experiments with the prostate cancer data were performed in two phases. The primary phase uses a multi-class dataset, whereas the secondary phase uses two binary-class datasets.

**Table 3 – Classification of prostate cancer samples from Lapointe's et al. [41] dataset**

| Actual | Microarray-MD | | |
|---|---|---|---|
| | Normal | Primary | Metastasis |
| Normal | 40 | 1 | 0 |
| Primary | 4 | 58 | 0 |
| Metastasis | 0 | 0 | 9 |

The dataset used in the primary experimental phase was first studied by Lapointe et al. [41] and it is available from the Stanford Microarray Database [40]. It consists of 112 samples with 44,016 gene expressions spanning three classes, namely 62 primary prostate tumors, 41 normal prostate samples and 9 pelvic lymph node metastases.

The gene expression matrix data file of the prostate cancer dataset was loaded to the system and the structure of the Decision Unit was determined to two blocks. The first block was assigned to the discrimination of the normal from the joint primary and metastatic samples, whereas the second block was assigned to the discrimination of primary from metastatic samples. The ranges of the parameter values set for training are apposed in Table 2.

Fig. 4 illustrates a screenshot of the Training Window with the selected options and the output of the system after training. The overall classification accuracy for the prostate cancer data was estimated to be 95.5% by leave-one-out cross validation on the available samples. The classification results per class are summarized in the confusion matrix of Table 3. This table shows that the system is capable of classifying 100% of the metastatic samples correctly, 2.4% of the normal samples as primary tumors and 6.5% of the primary tumors as normal. The system concluded that the optimal classification parameters for the particular medical problem include five input genes, 3rd-order polynomial kernel, $c = 0.5$ and $\gamma = 0.0156$ for the first block, and one input gene, linear kernel and $c = 0.5$ for the second block. These parameters were saved and applied for the classification of a sample of a phantom patient as illustrated in Fig. 6.

The prostate cancer datasets used in the secondary experimental phase are also publicly available. The first was intro-

**Table 2 – Training parameters used for prostate cancer classification**

| Parameters/dataset | Lapointe et al. [40] | Singh et al [42] | Welch et al. [43] |
|---|---|---|---|
| Pre-processing parameters | | | |
| Missing values estimation | Row average | – | – |
| Normalization | Mean/variance | Mean/variance | Mean/variance |
| Diagnostic parameters | | | |
| Classification sequence | Normal → Primary–Metastasis | Normal–Cancer | Normal–Cancer |
| Gene selection method | Welch's *t*-test | Welch's *t*-test | Welch's *t*-test |
| SVM kernel types | Linear, polynomial-2nd, polynomial-3rd, RBF | Linear, polynomial-2nd, polynomial-3rd, RBF | Linear, polynomial-2nd, polynomial-3rd, RBF |
| Cost ($c$) parameter range | $2^{-1}$–$2^{7}$ | $2^{-1}$–$2^{7}$ | $2^{-1}$–$2^{7}$ |
| Gamma ($\gamma$) parameter range | $2^{-8}$–$2^{-6}$ | $2^{-8}$–$2^{-6}$ | $2^{-8}$–$2^{-6}$ |
| Number of genes range | 1–11 | 1–10 | 1–3 |

**Table 4 – Classification of prostate cancer samples from Singh's et al. [42] dataset**

| Actual | Microarray-MD | |
|---|---|---|
| | Normal | Cancer |
| Normal | 43 | 7 |
| Cancer | 6 | 46 |

**Table 5 – Classification of the prostate cancer samples from Welch's et al. [43] dataset**

| Actual | Microarray-MD | |
|---|---|---|
| | Normal | Cancer |
| Normal | 9 | 0 |
| Cancer | 3 | 22 |

duced by Singh et al. [42]. It consists of 102 samples with 12,600 gene expressions. Its samples are distributed into 50 normal and 52 tumor samples. The second was studied by Welch et al. [43]. A total of 9 normal and 25 tumor samples with 12,626 gene expressions each were considered. As both of these datasets involve two classes a single-block structured Decision Unit was determined by the Microarray-MD system. The training parameter values set to the Training Window are presented in Table 2. The overall classification accuracy achieved for the first and the second dataset was 87.25% (linear kernel, $c = 0.5$ and four input genes) and 91.17% (linear kernel, $c = 2$ and two input genes), respectively. The corresponding confusion matrices are illustrated in Tables 4 and 5.

### 5.2. *Experiments with colon cancer data*

The colon cancer dataset used was first studied by Alon et al. [45]. It consists of 62 samples with 2000 gene expressions spanning two classes, namely 40 tumors and 22 normal colon samples.

The ranges of the training parameter values set to the Training Window are presented in Table 6. The colon cancer classification problem involves only two classes and thus a single-block structured Decision Unit was determined. The overall classification accuracy for the particular medical problem was estimated to be 90.3%. The classification results per class are summarized in the confusion matrix of Table 7. From this table it can be derived that Microarray-MD system provides a sensitivity of 92.5% and a specificity of 86.4%. The system concluded that the optimal classification parameters for the colon cancer classification problem include one input gene, linear kernel and $c = 0.5$.

**Table 6 – Training parameters used for colon cancer classification**

| Parameters/dataset | Alon et al. [45] |
|---|---|
| Pre-processing parameters | |
| Missing values estimation | – |
| Normalization | Mean/variance |
| Diagnostic parameters | |
| Classification sequence | Normal–Cancer |
| Gene selection method | Welch's *t*-test |
| SVM kernel types | Linear, polynomial-2nd, polynomial-3rd, RBF |
| Cost (c) parameter range | $2^{-1}$–$2^{7}$ |
| Gamma ($\gamma$) parameter range | $2^{-8}$–$2^{-6}$ |
| Number of genes range | 1–6 |

**Table 7 – Classification of the colon cancer samples from Alon's et al. [45] dataset**

| Actual | Microarray-MD | |
|---|---|---|
| | Normal | Cancer |
| Normal | 19 | 3 |
| Cancer | 3 | 37 |

## 6. Conclusions and prospects

Microarray-MD is a biomedical software system that is able to assist in the exploratory analysis of gene expression data, produced by microarray experiments. The major contribution of Microarray-MD is that it can provide physicians with substantial molecular-level information by exploiting gene expressions. The gene expression measurements are pre-processed and consequently used for the classification of the corresponding samples in two or more categories depending on their pathology. Through the simple and practical GUI of the proposed system novice users are offered the potential of using it with guidance provided by a helpful Wizard interface. The system is capable of performing automatic tuning of its parameters, thus simplifying the microarray analysis process for both novice and expert users. Moreover, expert users are offered the options to tune all the relevant parameters of the algorithms applied for decision making in medical research.

After testing different orderings of the blocks in the architecture of cascading classifiers, the system's accuracy has not been found to be significantly affected. However, if one would like to proceed to further fine tuning of the system, the ordering of the blocks could be determined (a) by the available knowledge provided by the medical experts on the particular medical problem [18] and (b) by considering the complexity and the overall classification accuracy of the architecture. Studies on cascading classifiers [33,46] suggest that the classifiers should be ordered in ascending complexity; that is, the less complex classifiers should be ordered before the more accurate and complex ones. However, the architecture of the proposed system consists only of SVM classifiers and embodies a search algorithm for the determination of their parameters, which in turn affects their complexity. For example, a large cost parameter could lead to an increase in the number of support vectors. Therefore, the ordering of the classifiers in ascending complexity is not directly feasible.

The Microarray-MD system has been tested on various publicly available DNA microarray datasets, including those provided by Stanford Microarray Database [40]. In most cases the overall classification accuracy it provides, measured by the almost unbiased leave-one-out procedure, exceeds 90%. Its high accuracy has been avouched in this paper by demon-

strating its application for the classification of prostate cancer and colon cancer data. As the number of samples increases a possible improvement of the system's accuracy would be feasible by considering balanced class distributions for each block [47]. In any way the generality of the results obtained by the system require further and careful consideration, from a biological and a medical point of view, by expert biologists and physicians.

Within our prospects is the enhancement of Microarray-MD by:

1. The incorporation of image processing techniques and analysis methods that could be applied directly to the images acquired from the microarray laser scanner, aiming at the automation of the whole process.
2. The incorporation of more sophisticated gene selection methods, such as genetic algorithms, aiming at the improvement of the classification accuracy.

## 7. Hardware and software specifications

Microarray-MD was developed in Microsoft Visual C++ 6.0 for Windows XP operating system. The implementation of the SVMs was based on the publicly available LibSVM library [34] which was modified to meet the needs of the particular application. The hardware requirements of the proposed system are minimal as it can run on most modern PCs. The example program runs presented were performed on a Pentium-4 PC 2800 MHz with 512 MB RAM.

## 8. Availability of the software

A version of the presented software is available for downloading from our web site http://rtsimage.di.uoa.gr/download.htm.

## Acknowledgments

We would like to thank Prof. M. Tzivras M.D., University of Athens, Medical School and his research group for their effort on the evaluation of the Microarray-MD system and their substantial suggestions. Moreover, we gratefully acknowledge the contribution of the anonymous referees to the improvement of this paper.

This work was realized under the framework of the Operational Program for Education and Vocational Training Project "Pythagoras" cofunded by European Union and the Ministry of National Education and Religious Affairs of Greece.

## REFERENCES

[1] K.-A. Do, B.M. Broom, S. Wen, GeneClust, in: G. Parmigiani, E.S. Garrett, R.A. Irizarry, S.L. Zeger (Eds.), The Analysis of Gene Expression Data: Methods and software, Springer, New York, NY, 2003, pp. 342–361.

[2] T. Hastie, R. Tibshirani, M.B. Eisen, A. Alizadeh, R. Levy, L. Staudt, W.C. Chan, D. Botstein, P. Brown, 'Gene shaving' as a method for identifying distinct sets of genes with similar expression patterns, Genome Biol. 1 (2) (2000), research0003.1–research0003.21.

[3] C. Li, W.H. Wong, Model-based analysis of oligonucleotide arrays: expression index computation and outlier detection, PNAS 98 (2001) 31–36.

[4] L.E. Peterson, CLUSFAVOR 5.0: hierarchical cluster and principal-component analysis of microarray-based transcriptional profiles, Genome Biol. 3 (2002), software0002.1–software0002.8.

[5] A. Sturn, J. Quackenbush, Z. Trajanoski, Genesis: cluster analysis of microarray data, Bioinformatics 18 (2002) 207–208.

[6] C. Colantuoni, G. Henry, S. Zeger, J. Pevsner, SNOMAD (Standardization and Normalization of MicroArray Data): web-accessible gene expression data analysis, Bioinformatics 18 (2002) 1540–1541.

[7] L.H. Saal, C. Troein, J. Vallon-Christersson, S. Gruvberger, A. Borg, C. Peterson, BioArray software environment: a platform for comprehensive management and analysis of microarray data, Genome Biol. 3 (2002), software0003.1–software0003.6.

[8] A. Saeed, V. Sharov, J. White, J. Li, W. Liang, N. Bhagabati, J. Braisted, M. Klapa, T. Currier, M. Thiagarajan, A. Sturn, M. Snuffin, A. Rezantsev, D. Popov, A. Ryltsov, E. Kostukovich, I. Borisovsky, Z. Liu, A. Vinsavich, V. Trush, J. Quackenbush, TM4: a free, open-source system for microarray data management and analysis, Biotechniques 34 (2003) 374–378.

[9] R. Gentleman, A.J. Rossini, S. Dudoit, Bioconductor FAQ, 2006, (http://www.bioconductor.org).

[10] Y. Su, T.M. Murali, V. Pavlovic, M. Schaffer, S. Kasif, RankGene: identification of diagnostic genes based on expression data, Bioinformatics 19 (2003) 1578–1579.

[11] D. Xu, V. Olman, L. Wang, Y. Xu, EXCAVATOR: a computer program for efficiently mining gene expression data, Nucleic Acids Res. 31 (2003) 5582–5589.

[12] T. Toyoda, A. Konagaya, Knowledge Editor: a new tool for interactive modeling and analyzing biological pathways based on microarray data, Bioinformatics 19 (2003) 433–434.

[13] R. Pieler, F. Sanchez-Cabo, H. Hackl, G.G. Thallinger, Z. Trajanoski, ArrayNorm: comprehensive normalization and analysis of microarray data, Bioinformatics 20 (2004) 1971–1973.

[14] E. Petricoin, J. Hackett, L. Lesko, R. Puri, S. Gutman, K. Chumakov, J. Woodcock, D. Feigal Jr., K. Zoon, F. Sistare, Medical applications of microarray technologies: a regulatory science perspective, Nat. Genet. 32 (2002) 474–479.

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

[16] M.P.S. Brown, W.N. Grundy, D. Lin, N. Cristianini, C.W. Sugnet, T.S. Furey, M. Ares, D. Haussler, Knowledge-based analysis of microarray gene expression data by using support vector machines, Proc. Natl. Acad. Sci. 97 (2000) 262–267.

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

[18] S. Theodoridis, K. Koutroumbas, Pattern Recognition, Academic Press, 1999.

[19] D.K. Iakovidis, I.N. Flaounas, S.A. Karkanis, D.E. Maroulis, A cascading support vector machine system for gene expression data classification, in: 2nd IEEE Conference on Intelligent Systems, Bulgaria, Varna, 2004, pp. 344–347.

[20] I.N. Flaounas, D.K. Iakovidis, D.E. Maroulis, S.A. Karkanis, Intelligent Analysis of Genomic Measurements, 13th

International Symposium on Measurements for Research and Industry Applications, IMEKO, Greece, Athens, 2004, 463–467.
[21] B. Phimister, Going global, Nat. Genet. Suppl. 1 (1999) 1.
[22] A. Schulze, J. Downward, Navigating gene expression using microarrays—a technology review, Nat. Cell Biol. 3 (2001) E190–E195.
[23] W. Zhang, I. Shmulevich (Eds.), Computation and Statistical Approaches to Genomics, Kluwer Academic Publishers, Boston, 2002.
[24] P. Hegde, R. Qi, R. Abernathy, C. Gay, S. Dharap, R. Gaspard, et al., Biotechniques 29 (2000) 548–562.
[25] O. Troyanskaya, M. Cantor, G. Sherlock, P. Brown, T. Hastie, R. Tibshiran, D. Botstein, R.B. Altman, Missing value estimation methods for DNA microarrays, Bioinformatics 17 (2001) 520–525.
[26] S. Oba, M. Sato, I. Takemasa, M. Monden, K. Matsubara, S. Ishii, A Bayesian missing value estimation method for gene expression profile data, Bionformatics 19 (2003) 2088–2096.
[27] W. Pan, A comparative review of statistical methods for discovering differentially expressed genes in replicated microarray experiments, Bioinformatics 18 (2002) 546–554.
[28] I. Guyon, A. Elisseeff, An introduction to variable and feature selection, J. Machine Learn. Res. 3 (2003) 1157–1182.
[29] I. Guyon, J. Weston, S. Barnhill, V. Vapnic, Gene selection for cancer classification using support vector machines, Machine Learn. 46 (2002) 389–422.
[30] T. Golub, D. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. Mesirov, H. Coller, M. Loh, J. Downing, M. Caligiuri, C. Bloomfield, E. Lander, Molecular classification of cancer: class discovery and class prediction by gene expression monitoring, Science 286 (1999) 531–537.
[31] M. Sun, M. Xiong, A mathematical programming approach for gene selection and tissue classification, Bioinformatics 19 (2003) 1243–1251.
[32] C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Kluwer Academic Publishers, 1998.
[33] A.K. Jain, R.P.W. Duin, J. Mao, Statistical pattern recognition: a review, Trans. Pattern Anal. Machine Intelligence 22 (2000) 4–37.
[34] C.C. Chang, C.J. Lin, LIBSVM: A Library for Support Vector Machines, 2001. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm, accessed November 2004.
[35] C. Ambroise, G. McLachian, Selection bias in gene extraction on the basis of microarray gene-expression data, Proc. Natl. Acad. Sci. 99 (2002) 6562–6566.
[36] S. Varma, R. Simon, Bias in error estimation when using cross-validation for model selection, BMC Bioinformatics 7 (2006) 91, http://www.biomedcentral.com/1471-2105/7/91.
[37] B. Myers, S.E. Hudson, R. Pausch, Past, present and future of user interface software tools, ACM Trans. Human Comput. Interact. 7 (2000) 3–28.
[38] A. Berrais, Knowledge-based expert systems: user interface implications, Adv. Eng. Software 28 (1997) 31–41.
[39] B. Shneiderman, Designing the User Interface: Strategies for Effective Human–Computer Interaction, 3rd ed., Addison-Wesley, Massachusetts, 1998.
[40] Stanford Microarray Database, http://genome-www5.stanford.edu, accessed November 2004.
[41] J. Lapointe, C. Li, J.P. Higgins, M. van de Rijn, E. Bair, K. Montgomery, M. Ferrari, L. Egevad, W. Rayford, U. Bergerheim, P. Ekman, A. DeMarzo, R. Tibshirani, D. Botstein, P. Brown, J. Brooks, J. Pollack, Gene expression profiling identifies clinically relevant subtypes of prostate cancer, Proc. Natl. Acad. Sci. 101 (2004) 811–816.
[42] 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 1 (2) (2002) 203–209.
[43] J.B. Welsh, L.M. Sapinoso, A.I. Su, S.G. Kern, J. Wang-Rodriguez, C.A. Moskaluk, H.F. Frierson Jr., G.M. Hampton, Analysis of gene expression identifies candidate markers and pharmacological targets in prostate cancer, Cancer Res. 61 (16) (2001) 5974–5978.
[44] A.K. Jain, B. Chandrasekaran, Dimensionality and Sample Size Considerations in Pattern Recognition Practice, Handbook of Statistics, vol. 2, 1982, pp. 835–855.
[45] 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, Proc. Natl. Acad. Sci. 96 (1999) 6745–6750.
[46] E. Alpaydin, C. Kaynak, Cascading classifiers, Kybernetika 34 (1998) 369–374.
[47] G.M. Weiss, F. Provost, The Effect of Class Distribution on Classifier Learning, Technical Report ML-TR-43, Dept. of Computer Science, Rudgers University, January 2001.