

# Gene feature selection

Ioan Tabus and Jaakko Astola

Institute of Signal Processing

Tampere University of Technology

P.O Box 553, FIN-33101 Tampere, Finland

## Abstract

This chapter presents an overview on the classes of methods available for feature selection, paying special attention to the problems typical to microarray data processing, where the number of measured genes (factors) is extremely large, in the order of thousands, and the number of relevant factors is much smaller. The main ingredients needed in the selection of an optimal feature set consist in: the search procedures, the underlying optimality criteria, and the procedures for performance evaluation. We discuss here some of the major classes of procedures which are apparently very different in nature and goals: a typical Bayesian framework, several deterministic settings and finally information theoretic methods. Due to space constraints only the major issues are followed, with the intent to clarify the basic principles and the main options when choosing one of the many existing feature selection methods.

## 1 Introduction

There are two major distinct goals when performing gene feature selection: the first is *discovering the structure* of the genetic network or of the genetic mechanisms responsible for the onset and progress of a disease; the second is eliminating the irrelevant genes from a classification (or prediction) model with the final end of *improving the accuracy* of classification or prediction. While there are many cases when both goals are equally relevant, there are others when only one of them is of primary focus.

This possible distinction of goals is certainly reflected at the methodological level, where the feature selection methods are usually split into two groups: filter methods and wrapper methods [22]. With *the filter methods* [7][50], the genes are ranked according to some general properties (correlation, mutual information, discriminative power) that are relevant for the prediction or classification problem at hand (e.g., correlation with a disease type), but without making it explicit at this stage what is the particular prediction model that is going to be used subsequently. After ranking of the single genes or of the various groups of genes, a suitable set of genes is identified and proposed as the feature set to be used for all subsequent analysis tasks. This hints that filter methods should have a more general goal than simply improving a certain type of predictor (e.g., a neural network), but rather should aim at finding the true structure of the interactions recorded by the experimental data, and as such, they would provide a useful set of features

for very different classification tools. On the other hand, the *wrapper methods* [10][20][22][32] are more straightforward, since they intend to restrain the set of factors so that the prediction ability of a certain given method is improved. With the wrapper method the prediction capability of a particular method is investigated for all possible groups of genes (or only for a chosen subset of them, if complete search is computationally too demanding), and the group offering the best performance is declared an optimal set of feature genes, which certainly maximizes the prediction abilities of the studied class of models, but may not be relevant for other classes of models. The same dichotomy between filter and wrapper methods is relevant for the evaluation stage, where the available set of data may be used in various ways for assessing the performance of a given feature set. The above distinction, made in terms of goals and methodology, appears clearly with most of the existing feature selection techniques.

Although ideally feature selection is a main step in the attempt to discover true biological relationships, rarely a feature set is claimed to have full biological relevance without further validation. The apparent optimal or quasi-optimal behavior observed over the studied experimental data is only the starting point for more detailed biological experimentation and validation. In light of this, many times the feature selection procedure ends up proposing *several* likely outstandingly performing feature sets, which can be used later in biological studies [20][21].

In pattern recognition tasks the information typically appears as vectors of a large number of variables. We can view the recognition algorithm as operating directly on the variables and consider the variables as features whence feature selection means selecting a useful subset of variables for further processing. In microarray data analysis this is often the case and feature selection essentially means gene selection. Sometimes it is better to draw the line between pre-processing and recognition closer to the end and assume that the actual recognition is done using perhaps quite complicated functions of the original variables. In this situation feature selection means both selection of the relevant variables and the process of forming suitable informative expressions of groups of variables.

The necessity of feature selection is well documented in the machine learning literature with examples where the performance of classification or prediction algorithms degrades quickly if irrelevant features are added, and even if relevant features are added, when they are correlated with the current features. This degradation is even more drastic in the realistic scenarios, where the underlying distribution is not known, and the classification algorithm must estimate the parameters from data; if the number of features is high, the variance of the large number of corresponding parameters is also high, and it becomes attractive to trade off the high number of features (which may be required for a low bias) for a smaller number of parameters, and implicitly a smaller variance (but unfortunately a higher bias). Since the achievable accuracy depends on the size of the available data sets, a somehow surprising situation occurs: the estimated optimal feature set depends on the size of the training set, being larger when the number of data points is large, but smaller when the available data sets are small.

We start the discussions giving a working definition of the optimal feature set, adapted here from [22], by which we attempt to clarify main issues to be tackled, rather than trying to give a general definition valid across all approaches. The dataset  $\mathcal{D} = \{(X_1(t), \dots, X_N(t), Y(t)) | t = 1, \dots, n\}$  has  $n$  instances of the features

and their corresponding class labels  $Y \in \{1, \dots, K\}$ . The full set of features is formed of all  $N$  available features,  $X_i$ ,  $i = 1, \dots, N$ , and is denoted by the vector  $\mathbf{X} = [X_1, \dots, X_N]^T$ , where  $T$  denotes transposition. A subset of selected features  $\{X_{i_1}, \dots, X_{i_k}\}$  will be specified in two equivalent ways: either by the set of selecting indices  $A = \{i_1, \dots, i_k\}$  or by a vector  $\boldsymbol{\gamma}$  of length  $N$  that has zeros everywhere, except the positions  $i_1, \dots, i_k$ , i.e.,  $\gamma_{i_1} = \dots = \gamma_{i_k} = 1$ , and consequently the feature set can be denoted as  $\mathbf{X}\boldsymbol{\gamma} = [X_{i_1}, \dots, X_{i_k}]^T$ .

*Definition* Given a classifier  $g(x_1, \dots, x_r; \boldsymbol{\theta})$  (able to operate on a variable number  $r$  of inputs, and depending on the tunable parameter vector  $\boldsymbol{\theta}$ ) and a data set  $\mathcal{D}$  with features  $X_1, \dots, X_N$  and target  $Y$ , sampled from a (unknown) joint distribution, the optimal feature set  $\{X_{i_1}, \dots, X_{i_k}\}$  is a subset of  $k$  features that maximizes the classification accuracy of the classifier  $g(X_{i_1}, \dots, X_{i_k}; \boldsymbol{\theta}^*)$  having the best parameter vector  $\boldsymbol{\theta}^*$ .

The most sensitive issues in the above definition are: the specification of the family of parametric classifiers; the selection of a suitable measure of accuracy; and the estimating accuracy when the underlying distribution is not known.

The organization of the chapter is as follows: in Section 2 we present several instances of the feature selection problem under idealistic settings, stressing on the importance of the proper definition of an optimal feature set, dependent on the considered types of models and adopted performance measures. The two important classes of feature selection methods, filter methods and wrapper methods, are discussed in Sections 3 and 4, respectively. A distinct group of methods is treated in 5, where the simultaneous search for the best feature set and the best model parameters makes the methods particularly efficient. Finally, in Section 6 we discuss the minimum description length (MDL) based feature selection, which is an information theoretic method grounded in the fundamental principle of choosing the models according to their description length.

## 2 Feature selection under ideal settings

It will be useful to review various attempts to formally define an optimal feature set under idealized and restricted scenarios. Incorporating all intuitive requirements which are usually associated to the (common sense) denomination *feature set* will show to be difficult even in the simple case of binary classification problems, indicating that precise qualifications need to be added to make the concept of *optimal feature set* well defined.

### 2.1 Bayes classification

In a typical Bayesian scenario, the set of all features contains  $N$  features (real valued random variables)  $\{X_i | 1 \leq i \leq N\}$ , which are related to the target value  $Y$  (or class label), which is a binary random variable. The full description of this dependency is provided by the joint distribution, specified by the pair  $(\mu, \eta)$ , where  $\mu$  is a probability measure for  $X$  (i.e., for a set  $B \subseteq \mathcal{R}^N$ , we have  $\mu(B) = P(X \in B)$ ) and  $\eta(x) = P(Y = 1 | X = x)$ . Using the description  $(\mu, \eta)$  we can compute the probability  $P((X, Y) \in C)$ , where for convenience we split the set  $C$  into the union

of two subsets corresponding to  $Y = 0$  and  $Y = 1$ , as follows:  $C = (C_0 \times \{0\}) \cup (C_1 \times \{1\})$ . Then the joint probability can be evaluated as  $P((X, Y) \in C) = \int_{C_0} (1 - \eta(x))\mu(dx) + \int_{C_1} \eta(x)\mu(dx)$ , see [4]. In the most ideal case, the joint distribution is known, for the purpose of evaluating the performance of a given classifier. The goal is to find a binary classifier  $g(X_{i_1}, \dots, X_{i_k})$ , which is a function of only  $k$  of the  $N$  features, such that the probability of error  $P\{g(X_{i_1}, \dots, X_{i_k}) \neq Y\}$  is as small as possible. The set of feature genes will be completely determined by the set of indices  $A_k = \{i_1, \dots, i_k\}$ . To not obscure the following discussion by the particular form of the function  $g$  (which may be e.g., a perceptron, a logistic regression, etc.) we consider here the best possible unrestricted classifier  $g^* : \mathcal{R}^k \rightarrow \{0, 1\}$ , which achieves the infimum of the Bayes classification error [4]

$$\varepsilon(A_k) = \inf_{g: \mathcal{R}^k \rightarrow \{0,1\}} P\{g(X_{i_1}, \dots, X_{i_k}) \neq Y\}. \quad (1)$$

Thus, we can evaluate the performance of a feature set by its Bayesian error. Obviously,  $\varepsilon(A_k) \geq \varepsilon(B_{k+1})$  whenever  $A_k \subset B_{k+1}$ , which expresses the monotonicity property under nesting, a property which is also found with many other performance measures.

In the above setting, the usefulness of a feature set is considered in a plausible way, but the monotonicity under nesting leads to a counter-intuitive solution: the full  $\mathbf{X}$  is one of the (many possible) best feature sets, because there is no gain in the Bayesian error by restricting the feature set. The non-uniqueness of the best feature set, and the fact that the full feature set  $\mathbf{X}$  is already an optimal solution, make the Bayesian error as such a non-attractive measure of optimality. Thus additional requirements must be added, e.g., we may ask which is the smallest size  $k$  at which the Bayes error is the best possible, i.e., ask for the smallest  $k^*$  for which there is a set  $A_{k^*}^*$  such that

$$\varepsilon(A_{k^*}^*) = \varepsilon(A_N) = \inf_{g: \mathcal{R}^N \rightarrow \{0,1\}} P\{g(X_1, \dots, X_N) \neq Y\}. \quad (2)$$

Unfortunately, for a generic joint distribution of  $(X, Y)$  the best set  $A_{k^*}$  will be identical to  $A_N$ , since generically each feature will carry some non-zero information regarding  $Y$ . So, the problem “what is the cardinality of the feature set” will receive the trivial answer, “ $N$ ”, for most distributions.

For that reason, further restrictions are needed for getting non-trivial answers, e.g., fix the value of  $k$  and ask which is the best feature set of cardinality  $k$ . This optimality problem is very difficult from a computational viewpoint, since one has to test  $\binom{N}{k}$  possible subsets of cardinality  $k$ , which is a feasible task only for small values of  $N$  and  $k$ . Thus, for a practical solution we should ask if there is a structural property of the optimal Bayes solution which may help in restricting the search. An answer in the negative is offered by the following result, due to Cover [3]: Choose an arbitrary ordering of all possible subsets of  $X$ , and index them as  $B^1, B^2, \dots, B^{2^N}$ ; if the consistency constraint  $i < j$  for  $B^i \subset B^j$  is satisfied (therefore  $B^1 = \emptyset, B^{2^N} = X$ ), then there exists a joint distribution of  $(X, Y)$  such that

$$\varepsilon(B^1) > \varepsilon(B^2) > \dots > \varepsilon(B^{2^N}). \quad (3)$$

According to this theorem, every possible subset of  $X$  with size  $k$  can be an optimal feature set  $A_{k^*}^*$ , and therefore there is no straightforward way to restrict the search

for  $A_{k^*}^*$ . In particular the following algorithm (dubbed here *Best individual k genes*) in general is doomed to fail: evaluate each feature  $X_i$  according to its Bayes error  $\varepsilon(\{i\})$  when used as a singleton feature set, and build the candidate set of  $k$  features using the best ranking features according to  $\varepsilon(\{i\})$ . Even when exhaustive search is computationally unfeasible, heuristical solutions are available and most notably branch-and-bound algorithms based on dynamic programming exist, which exclude from the search some subsets due to the monotonicity under nesting property [29].

The difficulties revealed in defining and finding an optimal feature set for the ideal case of Bayes error using an unrestricted classifier are indicative of the problems that will be faced with the realistic problem, that of finding the feature set for a constrained class of classifiers, and under a finite data set.

## 2.2 Learning classifiers under finite data

We briefly review the problems encountered when looking for a proper feature set by learning under finite data specification, especially under the ideal error free scenario. To simplify notations, in this section the features are discrete valued. We explore here various definitions of the intuitive notion of feature *relevance*, and analyze the relevance of features contained in the optimal feature set (the one that has the optimum accuracy).

Let  $S_i = \{X_1, \dots, X_{i-1}, X_{i+1}, \dots, X_N\}$  be the set of all features except  $X_i$ . Let  $S'_i$  be a subset of  $S_i$ . *Weak relevance* [22] of the feature  $X_i$  means that there exist some  $S'_i, x_i, y_i$  and  $s'_i$  for which  $p(X_i = x_i, S'_i = s'_i) > 0$  such that

$$p(Y = y|X_i = x_i, S'_i = s'_i) \neq p(Y = y|S'_i = s'_i), \quad (4)$$

i.e., the probability of a feature given a partial set of features will change if the information regarding feature  $X_i$  is withdrawn. Thus, at least for the class label  $y$ , the weakly relevant gene contains information which no other gene in  $S'_i$  can substitute. The *strong relevance* [22] of a feature means that the feature has weak relevance and in addition, the set  $S'_i$  satisfying condition (4) is the full feature set,  $S'_i = S_i$ .

As an example let consider a space of five binary features, and the case of observing a (target) random variable for which there is an error free representation as  $Y = X_1 \oplus X_2$ , and error free measurements are available from all the feature values, which are connected as follows:  $X_4 = \overline{X_2}$  and  $X_5 = \overline{X_3}$  [22]. The input space and the corresponding output value are represented as follows:

$X_1$	$X_2$	$X_3$	$X_4$	$X_5$	$Y$
0	0	0	1	1	0
0	0	1	1	0	0
0	1	0	0	1	1
0	1	1	0	0	1
1	0	0	1	1	1
1	0	1	1	0	1
1	1	0	0	1	0
1	1	1	0	0	0

We also suppose that all feature vectors compatible with the constraints (there are 8 such vectors) are equiprobable. Thus  $p(Y = 0|X_1 = 0, X_2 = 0) = p(Y =$

$0|X_1 = 1, X_2 = 1) = p(Y = 1|X_1 = 0, X_2 = 1) = p(Y = 1|X_1 = 1, X_2 = 0) = 1$ .  
 Feature  $X_1$  is strongly relevant, because  $p(Y = 0|X_1 = 0, X_2 = 0, X_3 = 0, X_4 = 1, X_5 = 1) = 1$  while  $p(Y = 0|X_2 = 0, X_3 = 0, X_4 = 1, X_5 = 1) = \frac{1}{2}$ . Further,  $X_2$  is weakly relevant, because  $p(Y = 0|X_1 = 0, X_2 = 0) = 1$  while  $p(Y = 0|X_2 = 0) = \frac{1}{2}$ . Weakly relevant is also  $X_4$ , but  $X_3$  and  $X_5$  are irrelevant (i.e. they are neither strongly relevant nor weakly relevant).

At this point it can be seen that the dependencies of the features affect their relevance with respect to the target, and restraining the feature set can lead to changes of status from weakly relevant to strongly relevant.

An unexpected fact is that the optimal feature set with respect to accuracy may not contain all relevant features. An example of this behavior is in the following scenario: there are two binary features, and all feature combinations are equiprobable. The “true” target is again supposed to be a deterministic function of the features,  $Y = X_1 \oplus X_2$ . Both features are strongly relevant. Suppose now that we want to find an optimal classifier in the family of threshold functions, e.g., the classifier needs to be representable as  $g(X_1, X_2; \theta) = \delta(\theta_1 X_1 + \theta_2 X_2 > \theta_0)$  for some positive numbers  $\theta_0, \theta_1, \theta_2$ , where  $\delta(\cdot)$  is the Kronecker symbol (equal to 1 if the argument is true and 0 otherwise). There are two optimal threshold functions: one has  $\theta = [0.5 \ 1 \ 0]^T$  and it is equal to  $g^*(X_1, X_2; \theta) = X_1$ , the other has  $\theta = [0.5 \ 0 \ 1]^T$  and it is equal to  $g^*(X_1, X_2; \theta) = X_2$ , and both have accuracy  $\frac{3}{4}$  (each makes two mistakes), and all other threshold functions have lower accuracy. It is thus possible that some strongly relevant features are left out of the feature set optimal as accuracy under a restricted class of classifiers. Despite of this, in practice, it is necessary to restrict the class of classifiers but the above phenomenon shows that it is important to do the restriction in a way that does not prevent the useful information entering the classification. The reasons why one may wish to restrict the class of allowed classifiers are at least two-folded: with a restricted class the search space can be considerably reduced, partly easing the computational burden; or even better, under restricted classes closed form optimal solutions may exist, completely eliminating the need for search.

### 2.3 Learning classifiers in realistic scenarios

The really important practical scenario: a finite experimental data set  $\mathcal{D} = \{(X_1(t), \dots, X_N(t), Y(t)) | 1 \leq t \leq n\}$  is given and a restricted family of classifiers is specified. One is interested in finding the optimal feature set, which minimizes the estimated accuracy of the classifier over the data set. Since now the underlying probability distribution of the random variable  $(X, Y)$  is not known, the accuracy of the classifier has to be estimated from the data themselves, and the estimate of an optimal feature set will depend on the chosen estimation method. In particular, the estimated accuracy will be subject to the common variance-bias trade-off: on one extreme the estimate will be variant but unbiased, on the other less variant but biased. Not only the found optimal feature set for a given classifier is affected by the estimation procedure, but also the attempt to compare the accuracy reached by various classifiers becomes difficult. A common recipe for lowering the variance of the estimate is to use the available data in a circular fashion in turns as training and test set, however this may lead to another, more subtle, form of over-fitting, making the estimated errors overly optimistic.

There is a large body of literature focusing on the application of various classification methods for the analysis of microarray data, including discussion of the feature selection step, many of which are briefly reviewed in the next sections. In general, any classification method can be used, and will present specific issues, in conjunction with a feature selection method. The support vector machines were successfully used in [10][14][30][31][38][48][49][52]. Various tree classification methods have been used and compared for microarray data in [7][13]. The  $k$ -nearest neighbor classification algorithms have been used in [8][27], and the Fisher linear discrimination tested in [7][51]. Many other methods are briefly reviewed in the rest of the chapter.

### 3 Filter methods

Several intuitive methods are used to assess the dependency between the features and the targets, such as mutual information, or correlation coefficient. One of the most often used measures for finding the discriminative power of a gene  $j$  is the ratio

$$BSS(j)/WSS(j) = \frac{\sum_{t=1}^n \sum_{k=1}^K \delta(Y(t)=k) (\bar{X}_j^{(k)} - \bar{X}_j)^2}{\sum_{t=1}^n \sum_{k=1}^K \delta(Y(t)=k) (X_j(t) - \bar{X}_j^{(k)})^2}, \quad (5)$$

where  $\delta(\cdot)$  is the Kronecker symbol;  $\bar{X} = 1/n \sum_t X_j(t)$  is the sample mean of feature  $j$ ,  $\bar{X}_j^{(k)} = \frac{1}{n^{(k)}} \sum_{t|Y(t)=k} X_j(t)$  is the sample conditional mean of the feature  $j$  given class  $k$ ,  $(\sigma_j^{(k)})^2 = \frac{1}{n^{(k)}} \sum_{t|Y(t)=k} (X_j(t) - \bar{X}_j^{(k)})^2$  is the sample variance of feature  $j$  conditional to class  $k$ , and  $n^{(k)}$  is the number of data points from the training set  $\mathcal{D}$  falling in class  $k$ . The criterion (5) was used as a filter method to select the feature space for an experimental comparison of a large number of different classifiers in microarray gene expression data [7].

Simple manipulations show that

$$BSS(j)/WSS(j) = \frac{\sum_k n_k (\bar{X}_j^{(k)} - \bar{X}_j)^2}{\sum_k n_k (\sigma_j^{(k)})^2}, \quad (6)$$

which explains the intuition behind the discriminative power of this measure: the total sum of squares  $TSS(j) = \sum_t (X_j(t) - \bar{X}_j)^2$  can be decomposed into two terms  $TSS(j) = BSS(j) + WSS(j)$ , where the first,  $BSS(j)$  shows the spread of the class averages with respect to the joint average (and ideally we want this spread to be as large as possible) while the second,  $WSS(j)$ , shows the spread of the points inside each class (and ideally this should be as low as possible).

For the case of only two classes ( $K = 2$ ) and equal number of instances in each class ( $n^{(1)} = n^{(2)}$ ), the measure (5) can be seen to be proportional to the Fisher discriminant ratio

$$FDR(j) = \frac{(\bar{X}_j^{(1)} - \bar{X}_j^{(2)})^2}{(\sigma_j^{(1)})^2 + (\sigma_j^{(2)})^2}. \quad (7)$$

The main disadvantage of filter methods is that they look at each feature independently when considering its merit in discriminating the target. If one feature  $X_{i_1}$  will be deemed to be highly discriminative for the target, so will be another feature,  $X_j$ , if  $X_{i_1}$  and  $X_j$  are highly correlated. Thus, when the feature set is redundant, a filter method will recommend the inclusion of all features which are individually highly discriminative, even though the information brought in by a redundant feature is not improving at all the prediction accuracy of the classifier. Including redundant features may degrade the accuracy of a predictor, and to avoid that, a preprocessing stage of clustering for identifying the highly correlated genes may be needed. Other preprocessing, like principal component analysis (PCA), which is intensively used in signal processing for tasks as signal compression or signal analysis, may also reduce very efficiently the number of features to be considered, if the features are highly correlated.

More elaborated filter techniques were tested in tandem with wrapper methods on the microarray data, like the Markov Blanket filter [50].

In a more loose sense, all methods for finding feature sets can be used as filters for other methods, e.g., for restraining the feature sets before resorting to methods that are more computationally intensive.

## 4 Wrapper methods

If the final goal is to improve the classification accuracy, the method for feature selection should consider the particular method for classification, with its biases and trade-offs regarding the evaluation of performance. For this reason, the most natural procedure is to cascade the feature selection process with the classifier design, and to iterate both in a loop until the best performance is achieved, the feature selection process being comparable with a wrapper over the classifier design and evaluation.

### 4.1 A generic wrapper method

The classifier design enters the wrapper algorithm as a black box, so the wrapper approach can be applied to a wide family of classifications algorithms. The algorithm for a generic wrapper is presented in Figure 2, while one of the blocks from Figure 2, namely the block *Find best feature set and parameters*, is detailed in Figure 1.

As presented, any wrapper method must consist of two nested cross-validation loops. In the most outer loop, the available data is split into training and test data (in Figure 2 the split is in 4 equal parts). There are many ways to make the split, and for each of them the algorithm in Figure 2 is run once, resulting in an estimate of the performance offered by the block *Find best feature set and parameters*, with a given classification method. The overall performance will be evaluated by the average of all errors  $\hat{\epsilon}$  over a large number of different 4-fold splits of the available data.

The blocks *Find best feature set and parameters* are the inner cross-validation loops, and one such block is illustrated in Figure 1 for a 3-fold split of the training data (remark that the available data here, is  $\mathcal{D}_A, \mathcal{D}_B, \mathcal{D}_C$ , i.e., only the training data

from the outer cross-validation loop). The selected feature set and the classifier parameters should perform well not only with the data used for design, they have also to generalize well over unseen data. To achieve this, at the stage of search of the feature set, trust should be given only to the performance over a set which was not seen when optimizing the parameters of the classifier. Thus, the search for a feature set is directed by a cross-validation error, as illustrated in Figure 1 with a three-fold error evaluation.

An important over-fitting situation was signaled repeatedly from the early attempts of finding feature sets (see the discussions in [10],[22]): the cross-validation error that was used for guiding the search process for the best feature should not be used as final performance measure of the chosen feature set for comparisons with other methods, since it is overly optimistic, in other words it is not fair to use the full data set  $\mathcal{D}$  for selecting the optimal features  $X_{i_1}, \dots, X_{i_{k^*}}$ , and then use the same data for the computation of the cross-validation error. Thus, the error  $\hat{\varepsilon}$  represented in Figure 2 should be used for reporting performance results.

## 4.2 The search of best feature set

The search process can be performed according to several heuristics, since in general exhaustive search is not computationally feasible. The greedy methods are usually based either on growing the optimal feature set starting from the empty set (in the *forward selection* methods), or on iteratively removing features starting from the full feature set (in the *backward* methods). The simplest search engine working in a forward selection mode is the *hill climbing*, where new candidate sets are obtained by combining the current feature set with each of the remaining features, and the performance is evaluated for each of the new candidates, the best performing being selected as new current feature set and the process continues until none of the candidates has performance better than the current set. A more laborious forward selection process is *the best first search* engine, where at each iteration all previously tested sets are checked for the best performer which was not already expanded, the best performer is then combined with the remaining features, and the process continues as long as needed (a fair stopping criterion is to check whether in the last  $k$  steps no improvement was obtained). In return for the increased complexity, the best first search algorithm sometimes reaches better accuracies than hill climbing, but there are also reports of no improvement situations. The mentioned forward selection searches have the disadvantage known as nesting property, namely that a feature, once entered into a feature set, it can not be excluded latter. To eliminate the nesting drawback, in the sequential forward floating selection (SFFS) after each inclusion of a new feature a backtracking process is started to check for possible improvements by removal of an existing feature, and when all paying-off removals have been executed, the algorithm resumes to adding a new feature. With this new search policy one checks more subsets, but it takes a longer time to run. It was widely believed that this computational expense is compensated by better results. However, a recent paper, [32], pointed out that the comparisons in some publications reporting better results were obtained using the wrong cross-validation methodology; when performed in the correct cross-validation framework, the results of the simple forward selection search are as good or better than SFFS in many cases. The results of [32] are of even more general interest, they show that selecting the

wrong cross-validation evaluation environment is not benign, it does not lead only to too optimistic reported errors uniformly for all methods, but it may also lead to (involuntarily) dishonest inversions of ranking of the tested methods.

### 4.3 Optimization criteria and regularization

The search for the best set of genes  $\{X_{i_1}, \dots, X_{i_k}\}$  and for the parameter vector  $\hat{\boldsymbol{\theta}}$  of the best classifier  $g(X_{i_1}, \dots, X_{i_k}; \hat{\boldsymbol{\theta}})$  should be directed by an as relevant criterion as possible. The sample variance

$$J(\boldsymbol{\theta}, \gamma) = \frac{1}{n} \sum_{t=1}^n (Y(t) - g(X_{i_1}(t), \dots, X_{i_k}(t); \boldsymbol{\theta}))^2 \quad (8)$$

is relevant in itself and has the obvious merit of simplicity and ease of evaluation, though usually requires additional constraints for making the solution unique.

The particular case of the perceptron, which can be defined as a linear combiner followed by a threshold, i.e.,  $g(X_{i_1}, \dots, X_{i_m}; \hat{\boldsymbol{\theta}}) = T(\sum_{j=1}^m X_{i_j} \theta_j)$  with  $T(x) = 0$  for  $x < 0$  and  $T(x) = 1$  for  $x \geq 0$ , was successfully considered in several gene features methods [20][21]. In order to obtain a closed form solution of (8) the classifier is further approximated by the simpler linear combiner (or linear regression) during the optimization stage, as follows: denoting the vector of observations  $\mathbf{X}_\gamma(t) = [X_{i_1}(t), \dots, X_{i_k}(t)]^T$ , the least square solution minimizing (8) for the linear model  $g(\mathbf{X}_\gamma(t), \boldsymbol{\theta}) = \boldsymbol{\theta}^T \mathbf{X}_\gamma(t)$  is

$$\hat{\boldsymbol{\theta}} = \left( \sum_{t=1}^n \mathbf{X}_\gamma(t) \mathbf{X}_\gamma(t)^T \right)^{-1} \sum_{t=1}^n \mathbf{X}_\gamma(t) Y(t) \quad (9)$$

and it is unique if  $\hat{R} = \sum_{t=1}^n \mathbf{X}_\gamma(t) \mathbf{X}_\gamma(t)^T$  is non-singular (otherwise  $\hat{R}^{-1}$  should be taken as a notation for the pseudo-inverse of  $\hat{R}$ ). Unfortunately, for large values of  $k$  (and in particular for all  $k > n$ ) which are of interest in gene feature selection problems, the matrix  $\hat{R}$  is singular, signaling that many optimal vectors  $\boldsymbol{\theta}$  exist that reach the lowest possible value of the criterion (8),  $J(\hat{\boldsymbol{\theta}}, \gamma) = 0$ . Avoiding these degenerate situations can be achieved by standard methods, known as regularization techniques, the most used being ridge regression and cross-validation .

#### 4.3.1 Ridge regression

With ridge regression a penalty term  $\sigma^2 \|\boldsymbol{\theta}\|^2$  is added to  $J(\boldsymbol{\theta})$  to prevent solutions with too large parameters, and  $\sigma^2$  is a weighting on how strong this penalty should be. The minimum value of the new criterion

$$J(\boldsymbol{\theta}, \gamma) + \sigma^2 \|\boldsymbol{\theta}\|^2 = \frac{1}{n} \sum_{t=1}^n (Y(t) - \boldsymbol{\theta}^T \mathbf{X}_\gamma(t))^2 + \sigma^2 \|\boldsymbol{\theta}\|^2 \quad (10)$$

is reached by

$$\hat{\boldsymbol{\theta}} = \left( n\sigma^2 I + \sum_{t=1}^n \mathbf{X}_\gamma(t) \mathbf{X}_\gamma(t)^T \right)^{-1} \sum_{t=1}^n \mathbf{X}_\gamma(t) Y(t), \quad (11)$$

which also reminds of a well known regularization method for solving ill-conditioned system of equations.

The same solution (11) is optimal solution for a very interesting related problem: consider new feature  $X'_i$ , obtained by adding white noise with variance  $\sigma^2$  to all features, i.e.,  $X'_i(t) = X_i(t) + e_i(t)$ , where  $e_i(t)$  is white and independent of  $X_j(t)$  and  $Y(t)$ , and consider the problem of minimizing the expected value

$$EJ(\boldsymbol{\theta}, \boldsymbol{\gamma}) = E \sum_{t=1}^n (Y(t) - \boldsymbol{\theta}^T \mathbf{X}'_{\boldsymbol{\gamma}}(t))^2 \quad (12)$$

where  $E$  denotes the expectation operator. The optimal solution minimizing (12) is long known to be exactly (11) (see, e.g., [11],[40]). The attractive interpretation of the newly generated features  $X'_i$  is that of an additional training set (of infinite size), which transforms the set of points of coordinates  $X_{\boldsymbol{\gamma}}$  into a set of spheres centered at  $X_{\boldsymbol{\gamma}}$ , the common radius of the spheres being controlled by  $\sigma^2$ . With original training sets of small size too many “perfect” solutions exist, if there is a perfect solution, then there are an infinity of hyperplanes separating perfectly the points, and they may be considered artifacts due to the degeneracy of the criterion at the current sample size. The additional training set helps in removing those degenerate solutions. This principle is applied and further developed in [21] where learning of the parameter vector  $\boldsymbol{\theta}$  of the perceptron classifier, is approximated by learning of the parameter vector  $\boldsymbol{\theta}$  of the associated linear classifier, in order to accelerate the training process.

#### 4.3.2 Cross-validation as a regularization technique

cross-validation is intuitively a proper method for evaluating the performance of a classifier in more objective way than the sample variance of the classification errors over the training set [18][41][42]. But apart of that, cross-validation is also a tool for regularizing the solution of the criterion (8). In the case of linear regression, the cross-validation criterion can be computed in closed form, making it attractive e.g., for evaluating the performance of the closed form solutions offered by the ridge regression methods, for various values of the parameter  $\sigma^2$ .

#### 4.3.3 Coefficient of determination (COD)

The coefficient of determination is a normalized version of the classification error variance, in which the performance of the tested classifier is normalized with respect to the performance of a classifier that uses no features [6], i.e.,

$$COD = \frac{\sigma_{\emptyset}^2 - \sigma_{\boldsymbol{\gamma}}^2(\boldsymbol{\theta}_{\boldsymbol{\gamma}})}{\sigma_{\emptyset}^2}, \quad (13)$$

where  $\sigma_{\emptyset}^2$  in the case of linear classifiers is the variance of the class label  $Y$  (i.e., the cost of predicting the class by the average over the full dataset). Different types of COD can be defined for each different type of variance: a true COD corresponds to true variances, estimated in a Bayesian setting where the true joint distribution is known; a re-substitution COD corresponds to the variance of the re-substitution errors, obtained over the full data set for the best predictor  $\boldsymbol{\theta}_{\boldsymbol{\gamma}}^*$  designed over the

full data set; and a cross-validation COD corresponding to the cross-validation errors. The COD is more advantageous than the variance itself when discussing the performance of a predictor, since it is normalized to have the maximum value of 1. This normalization does not affect otherwise the ranking of performance of two predictors as compared to the ranking performed by the variance itself, but proves useful in presenting comparatively the results for different data-sets [17][19][20].

#### 4.3.4 Information theoretic criteria

Apart of the criteria based on the variance of the errors, there are many information theoretic methods to define the amount of information about the target contained in a set of genes. Any of these criteria can be used when performing the search for the best set of genes in the context of Figure 2. However, a true wrapper method will need to use the inner cross-validation loops depicted in Figure 1 for avoiding the over-fitting of the model, while the information theoretic criteria [1] such as minimum description length, or Akaike information criterion (AIC), or Bayes information criterion (BIC) are in themselves protected against over-fitting, and therefore it is no need to iterate over the expensive cross-validation loops presented in Figure 1.

Information theoretic measures are used in different forms by various procedures for feature selections and predictor design [5][9][15][43][44][45][46][47][54]. In Section 6 we review the minimum description length principle and its use for feature selection.

## 4.4 Methodology for practical evaluation

The evaluation of feature extraction procedure is often confused with the evaluation of a classification procedure, and for that reason it is carried on with the wrong methodology.

In all gene selection experiments a data set  $\mathcal{D}$  is available, containing measurements of the  $N$  gene expressions  $\{X_1(t), \dots, X_N(t) | 1 \leq t \leq n\}$  for each of the  $n$  patients and the class labels  $\{X_1(t), \dots, X_N(t) | 1 \leq t \leq n\}$  for each patient. Two important problems can be distinguished:

1. *Predictor evaluation* A set of selected gene features  $X_{\gamma_0} = \{X_{i_1}, \dots, X_{i_k}\}$  has been specified apriori, through the selection vector  $\gamma_0$ , (but it was not inferred from a gene selection from the data  $\mathcal{D}$ !) and the goal is to compare different predictors classes,  $g(X_{\gamma_0}, \theta)$ , e.g., compare perceptrons against multilayer perceptrons. A methodology which avoids over-fitting is the one presented in Figure 1, and the quantity  $\hat{\varepsilon} | \gamma_0$  is a good measure of accuracy achieved by different predictors classes (the number of folds in cross-validation can certainly be chosen other than three, but kept the same if several predictors are compared).
2. *Evaluation of a procedure for feature selection* If a certain procedure for feature selection needs to be compared with another one, the cross-validation scenario should be the one presented in Figure 2 (with a number of folds conveniently chosen, four was used only for illustration) and the quantity  $\hat{\varepsilon}$  is a good measure of accuracy. With the leave-one-out cross-validation the data is

folded in  $n$ , such that each data point is left-out during the training and used as a test point. If the number of errors in a leave-one-out cross-validation experiment is zero for all tested feature selection methods, one needs to reduce the number of folds, in order to test the generalization power of different schemes in a more stringent experiment. By reducing the number of folds, the number of possible splits of the data into training and test also increases, thus a larger number of tests can be carried on.

## 5 Simultaneously searching the best feature set and best model parameters

### 5.1 Bayesian approaches

In the Bayesian setting, a model of the conditional probability  $p(Y_t = 1|\boldsymbol{\theta}, \boldsymbol{\gamma})$  is specified and some prior distributions are suitably chosen, such that the posterior probability  $P(\boldsymbol{\theta}, \boldsymbol{\gamma}|\mathcal{D})$  can be either computed by integration, or can be sampled, and then the optimal feature set is taken to be the maximum a posteriori estimation  $\boldsymbol{\gamma}^*$  that reaches the maximum of  $P(\boldsymbol{\theta}, \boldsymbol{\gamma}|\mathcal{D})$ .

There are several possible models to be considered for the conditional probability  $p(Y_t = 1|\boldsymbol{\theta}, \boldsymbol{\gamma})$ : logistic regression [25][28] and probit regression [24][26][37][53].

The probit regression model of the conditional probability is  $p(Y_t = 1|\boldsymbol{\theta}, \boldsymbol{\gamma}) = \Phi(\mathbf{X}_\gamma^T \boldsymbol{\theta})$ , where the so called probit link function  $\Phi(\cdot)$  is the normal cumulative distribution,  $\Phi(z) = (1/\sqrt{2\pi}) \int_{-\infty}^z \exp(-x^2/2) dx$ . Due to the computationally demanding integrations required, the use of this model was quite limited until the introduction of Gibbs sampling algorithms based on data augmentation (for a recent account see [12]). In addition to  $\boldsymbol{\theta}, \boldsymbol{\gamma}$ , a set of latent (unobserved) variables  $Z(1), \dots, Z(n)$  is introduced, with  $Z(t) \sim N(\mathbf{X}_\gamma^T(t)\boldsymbol{\theta}, 1)$ , i.e., the link to the regressors is provided by  $Z(t) = \mathbf{X}_\gamma^T(t)\boldsymbol{\theta} + e_i$  where  $e_i \sim N(0, 1)$ . Prior distributions are selected such that they are easily interpretable and the resulting Markov chain Monte Carlo (MCMC) simulations have a good computational speed and reliable model fitting. The unknowns  $(Z, \boldsymbol{\theta}, \boldsymbol{\gamma})$  will be obtained by Gibbs sampling from the posterior distribution.

A prior distributions  $\boldsymbol{\gamma}$  should reflect the need to select small sets of features, and a simple way to enforce a small number of units in  $\boldsymbol{\gamma}$  is to take a small value for  $P(\gamma_i = 1) = \pi_i$ , and consider that the elements of  $\boldsymbol{\gamma}$  are independent one of another, so  $P(\boldsymbol{\gamma}) = \prod_{i=1}^N \pi_i^{1-\gamma_i} (1-\pi_i)^{\gamma_i}$ . Given  $\boldsymbol{\gamma}$ , the prior on the regressor vector is  $\boldsymbol{\theta}_\gamma \sim N(0, c(X_\gamma^T X_\gamma)^{-1})$  where  $X_\gamma$  is the matrix having as column  $t$  the vector  $\mathbf{X}_\gamma(t)$  and  $c$  is a constant to be chosen by the user.

By applying Bayesian linear model theory, the following conditional distributions can be computed after evaluating several integrals:  $p(\boldsymbol{\gamma}|Z)$ ,  $p(Z|\boldsymbol{\theta}_\gamma, \boldsymbol{\gamma})$  and  $p(\boldsymbol{\theta}_\gamma|Z, \boldsymbol{\gamma})$  [26]. By iteratively drawing  $\boldsymbol{\gamma}, Z$  and  $\boldsymbol{\theta}$  from these distributions one obtains a MCMC, which will allow to make inference even in this case, in which there is no explicit expression for the posterior distribution. After an initial burn-in period it is assumed that the samples are taken from the posterior distribution, and one can compute the relative number of times each gene was selected for making a decision about including it in the feature set [26]. An extension to the multi-class classification problems can be obtained by generalizing the probit model to

multinomial probit models [53]. After finding the average frequency  $\bar{\gamma}_i$  by which each gene was selected during the MCMC simulation, the most important genes for a (sub)optimal feature set are chosen, e.g., those for which  $\bar{\gamma}_i$  exceeds a certain threshold. With the selected genes, either the probit model is used to make classifications, or other models may be used, if they give better accuracy [53].

A different optimization approach is the expectation-maximization (EM) method in which Laplacian priors for  $\gamma$  and  $\theta$  are chosen to reflect our need of a sparse solution. The hyperparameters  $\tau, \rho$  of these distributions, and the latent variables  $Z(1), \dots, Z(n)$  are used in an EM optimization algorithm, where in the first step (E-step) the expected value  $Q(\theta, \gamma | \hat{\theta}^\ell, \hat{\gamma}^\ell)$  of the posterior  $\log P(\theta, \gamma | D, Z, \tau, \rho)$  is computed conditioned on the current estimates  $\hat{\gamma}^\ell$  and  $\hat{\theta}^\ell$ , and in the second step (M-step) new estimates are obtained by the maximization  $\hat{\theta}^{\ell+1}, \hat{\gamma}^{\ell+1} = \arg \max_{\theta, \gamma} Q(\theta, \gamma | \hat{\theta}^\ell, \hat{\gamma}^\ell)$  [24].

## 5.2 Deterministic criteria allowing one-step solutions

We consider here the binary classification problem and the task is to find a perceptron which optimally separates the classes. If the problem has a perfect solution,  $\theta^*$ , the hyperplane  $z = \theta^{*T} \mathbf{X} + b$ , will be such that the available data points  $(\mathbf{X}(t), Y(t))$  satisfy  $Y(t) = \text{sign}(z(t))$  and  $Y(t)(\theta^{*T} \mathbf{X}(t) + b) \geq 1$ . To encourage a sparse solution  $\theta^*$ , i.e., most of the elements of  $\theta^*$  to be zero, the following criterion should be minimized:  $\|\theta\|_0 \triangleq \sum_i \theta_i^0 = \sum_i \gamma_i$ , subject to the constraint that all points are classified correctly (with the convention  $0^0 = 0$ ).

$$\begin{aligned} \min_{\theta, b} \quad & \|\theta\|_0 \\ \text{s.t.} \quad & Y(t)(\theta^T \mathbf{X}(t) + b) \geq 1, \quad 1 \leq t \leq n. \end{aligned} \quad (14)$$

The problem has to be relaxed in several ways, to make it tractable and to account for more realistic, noisy, situations.

The foremost modification is the substitution of  $\ell^0$  norm with the  $\ell^1$  norm in the criterion, to transform the problem into a convex optimization problem, for which efficient numerical solutions exist. As shown in [16], for a large number of problems the transformation of an  $\ell^0$  optimization problem into a  $\ell^1$  optimization problem leads to a solution for the later identical to the solution of the former, provided that the solution is sufficiently sparse. Even though in our general scenario the sufficient conditions established in [16] may not hold, and so the solutions of the two problems may be different, the resulting solution of the  $\ell^1$  problem has a high degree of sparsity. The situation is completely different with the  $\ell^2$  norm, where a lot of small (nonzero) values for the  $\theta_i$ 's are encouraged by their down-weighting in the criterion through taking the square.

A second major modification accounts for possible classification errors, and in order to minimize their effect, the non-negative slack variables  $\xi(t)$  are introduced to relax each constraint,  $Y(t)(\theta^T \mathbf{X}(t) + b) \geq 1 - \xi(t)$  and the penalty term  $\sum_{t=1}^n \xi(t)$  is added to the criterion, weighted by a constant  $C$ [2]:

$$\min_{\theta, b} \quad \|\theta\|_1 + C \sum_{t=1}^n \xi(t)$$

$$\begin{aligned}
s.t. \quad & Y(t)(\boldsymbol{\theta}^T \mathbf{X}(t) + b) \geq 1 - \xi(t), \\
& \xi(t) \geq 0, \quad 1 \leq t \leq n.
\end{aligned} \tag{15}$$

The similarity between (10) and (15) is obvious when  $n\sigma^2 = 1/C$ : one considers the sum of squared errors regularized by the  $\ell^2$  norm of the parameter vector, the other considers the sum of absolute values of the errors regularized by the  $\ell^1$  norm of the parameter vector.

The differences come from the fact that by directly optimizing (10) the solution is in general non-sparse, so one has to optimize for the huge number of  $2^N$  combinatorial choices of  $\boldsymbol{\gamma}$  in order to find the best sparse solution, while solving (15) can be realized in a single optimization step, as described next. To allow both positive and negative elements in  $\boldsymbol{\theta}$  the positive variables  $u_i$  and  $v_i$  are introduced such that  $\theta_i = u_i - v_i$  and  $|\theta_i| = u_i + v_i$  and thus (15) is transformed into the new optimization problem [2]:

$$\begin{aligned}
\min_{\mathbf{u}, \mathbf{v}, b} \quad & \sum_{i=1}^N (u_i + v_i) + C \sum_{t=1}^n \xi(t) \\
s.t. \quad & Y(t)((\mathbf{u} - \mathbf{v})^T \mathbf{X}(t) + b) \geq 1 - \xi(t), \\
& \xi(t) \geq 0, \quad 1 \leq t \leq n, \\
& u_i \geq 0, v_i \geq 0, \quad 1 \leq i \leq N,
\end{aligned} \tag{16}$$

which is a standard linear programming (LP) problem with inequality constraints.

Therefore, once the regularization constant  $C$  is chosen, solving the optimization problem (16) for the sparse vector of parameters  $\boldsymbol{\theta}$  (and hence for the optimum  $\boldsymbol{\gamma}$  which has the same pattern of zeros as  $\boldsymbol{\theta}$ ) can be done in a very fast way, using standard numerical optimization tools.

### 5.3 Joint feature clustering and classification

The feature selection problem refers to extracting an informative and non-redundant set from the existing features. The somehow related problem of forming new features (e.g., by linear combination of some existing genes) is traditionally considered a distinct problem and it was well studied in the past. However, there are many methods having as a preliminary stage (before proceeding to classification) the formation of new features, notable examples being principle component analysis and clustering. The *SimClust* algorithm described in [15] is one such approach and is mentioned here because it is well related to the method described in Section 6. *SimClust* solves a combined problem: simultaneously find clusters (groups of related genes) and classify the samples using as classification features the ‘‘average’’ genes which are the centers of the obtained clusters. To this goal, a MDL cost is associated to describing the clusters, and another MDL cost is associated to the regression model for optimal scoring. Relative weights are set such the two MDL costs have the same importance when the sample size is increased. We remark that the method is an involved mixture of techniques, e.g., prior to starting the computational demanding part algorithm, a filter approach is used to restrain the set of genes to only  $T$  genes, those having the largest values of between-to-within-class sum of squares (5).

## 6 Minimum description length based feature selection

Minimum description length was used as a basis for statistical inference in a large number of problems dealing with finding the structure of models. Its bases were laid down about 25 years ago in [33] inspired by the work on complexity in [23][39], and further refined in a number of papers [34][35][36]. As a fundamental principle, it states that given a model class, the best model should be chosen based on its ability to represent the data in the most compact form, i.e., with the minimum description length. Evaluating the description codelength is closely related to probabilistic modelling, since if one knows the probability  $P(x^n)$  of a string  $x^n$ , the optimal description length can be shown to be  $-\log_2 P(x^n)$ . For making inference, the value itself of the description length is sufficient, but it is worth noting that this value can really be achieved in practice when using arithmetic coding (within a very good precision), so the description length of sequences or parameters in this section really refer to short descriptions of the data or parameters, descriptions that can be decoded into the original sequences. The overall MDL is based on efficient lossless codes, and we can show at anytime a real message encoding the data with the claimed description length, since MDL takes care of all the costs involved by a real description. In contrast, many other methods of feature selection based on entropy or mutual information systematically neglect parts of the costs of encoding the data, i.e., when the empirical entropy of a sequence is used for inference, the cost of encoding the probability model is not included, and thus comparisons across models of different complexities may be unfair.

### 6.1 MDL using two part codes

In order to apply MDL to feature selection, first a model class should be chosen and then the total description length needs to be computed. If a two part code is used, as in [33], the total description length can be defined as the sum of the description of the model parameters and the description of the data given the model parameters. This instance of MDL was applied to genomic data for the first time in [43] in the following way. Given a class of predictors, e.g., perceptrons, the optimal predictor of the target  $\{Y(t)|1 \leq t \leq n\}$  given the input data  $\{\mathbf{X}(t)|1 \leq t \leq n\}$  is first found and its optimal parameters are denoted  $\theta_\gamma^*$ . The prediction errors  $\{\varepsilon(t) = Y(t) - T(\mathbf{X}^T(t)\theta_\gamma^*)|1 \leq t \leq n\}$  obtained with the optimal predictor  $\theta_\gamma^*$  are then encoded by a simple code e.g., by encoding the locations of the nonzero errors, using  $L_\varepsilon$  bits. For encoding the optimum perceptron parameters  $\theta_\gamma^*$  the most simple code will be constructed by assuming that all distinct perceptrons having  $k = \sum_i \gamma_i$  inputs (there are  $n_{\theta_\gamma}$  of them) are equally likely, and it will require  $L_{\theta_\gamma} = \log_2(n_{\theta_\gamma})$  bits. The total description length of the two part code,  $L_{tot}(\gamma) = L_\varepsilon + L_{\theta_\gamma}$  can be used as a criterion for discriminating between different structures  $\gamma$ . The penalty introduced by encoding the parameters of the model,  $L_{\theta_\gamma}$ , is clearly increasing with the perceptron size,  $k$ , and therefore it constitutes a penalty on the complexity of the model. If the order  $k$  of a model is too large, the optimal predictions obtained with the model will be very good, thus the cost

of encoding the errors,  $L_\varepsilon$ , may be very small or even zero, but such a model is discouraged by the complexity penalty. On the other hand, a too simple model will be unable to predict well the target, given the input, so the cost of the errors will be high, even though the cost of the model (its complexity  $L_{\theta_\gamma}$ ) will be small. So the proper balance between modelling power and model complexity is established by the MDL principle, to guide the selection of the right structure  $\gamma$ . The nice theoretical properties of MDL for selecting the structure of models have been established for a wide class of models, and the simplicity of its use makes it an attractive tool for feature selection.

## 6.2 MDL using normalized maximum likelihood models

Although the pioneering work on MDL with two part codes proved to be practical and well justified theoretically, the later developments in universal data compression have introduced more efficient codes to be used for the evaluation of description lengths [1][36]. In the rest of this section we present the approach introduced in [44], to which we refer for more details and proofs.

The target, or class label, is a string  $Y^n = Y(1), \dots, Y(n)$  of  $n$  realizations of the random variable  $Y$ , taking values in the set  $\{0, \dots, M-1\}$ . Each  $Y(t)$  is observed together with the  $k$ -tuple (a column vector)  $\mathbf{X}(t) = [X_{i_1}(t) \dots X_{i_k}(t)]^T$ , where the features are discrete valued  $X_i(t) \in \{0, \dots, n_q - 1\}$ , being quantized to  $n_q$  levels. The sequence of regressors  $\mathbf{X}^n = \mathbf{X}(1), \dots, \mathbf{X}(n)$  contains vectors which may occur repeatedly, and let  $K$  denote the number of distinct vectors,  $K \leq n$ , which means that the various vectors  $\mathbf{X}(t)$  belong to a finite set denoted  $\{\mathbf{b}_1, \dots, \mathbf{b}_K\}$ . In the end we need to collect counts of the symbols conditioned on a specific value of the regression vector in order to estimate model parameters, so we introduce the following notations: let  $\mathcal{I}_j$  denote the set of indices at which  $\mathbf{b}_j$  is found in the sequence  $\mathbf{X}(1), \dots, \mathbf{X}(n)$ ; i.e.,  $\mathcal{I}_j = \{t : \mathbf{X}(t) = \mathbf{b}_j\}$ , which has cardinality  $M_j = |\mathcal{I}_j|$ . The count  $m_q^j = |\{i : y_i = q, i \in \mathcal{I}_j\}|$  is the number of observations  $Y(t) = q \in \{0, \dots, M-1\}$  at which the regressor vector was  $\mathbf{X}(t) = \mathbf{b}_j$  and the counts should obey  $m_0^j + \dots + m_{M-1}^j = M_j$ .

The probability distributions  $P(Y^n | \mathbf{X}^n; \boldsymbol{\eta}, \boldsymbol{\nu})$  of the class labels  $Y^n$  conditioned on a given value of the regressor  $\mathbf{X}^n$  are parameterized by the sets of parameters  $\boldsymbol{\eta}$  and  $\boldsymbol{\nu}$ . We call the model a discrete regression model, since  $y$  takes values from a discrete set, and the joint observations  $\mathbf{X}$  take values also in a discrete set.

The number  $K$  of different vectors  $\mathbf{b}_1, \dots, \mathbf{b}_K$  appearing in the regressor sequence  $\mathbf{X}(1), \dots, \mathbf{X}(n)$  can be large, and if it is close to  $n$  it will be difficult to use at each of the regressors  $\mathbf{b}_i$  a conditional multinomial model with distinct parameters, since not enough observations will be available to estimate the probabilities of the symbols  $Y(t)$  conditional on  $\mathbf{X}(t) = \mathbf{b}_i$ . Because of this dilution phenomenon, pooling of the frequencies of occurrence of  $Y(t)$  at different context together will be beneficial, but pooling should be performed after re-ordering the counts at a given  $\mathbf{b}_i$  in a decreasing sequence.

The permutations  $\nu_i(\cdot) \in \Upsilon_M$  (where  $\Upsilon_M$  denotes the set of  $M!$  permutations of the set  $\mathcal{Y} = \{0, 1, \dots, M-1\}$ ) can be used to reorder the class labels  $j \in \mathcal{Y}$  such that the frequencies of occurrence of the class labels (observed at the time moments  $t$  with  $\mathbf{X}(t) = \mathbf{b}$ ) are arranged in decreasing order.

To make clear the reordering, we introduce a new string  $Z^n$ , obtained from the class labels  $Y^n$  by use of the set of permutations  $\boldsymbol{\nu} = (\nu_1(\cdot), \dots, \nu_K(\cdot))$  as follows:

$$\begin{aligned} Z(t) &= \nu_\ell^{-1}(Y(t)) \\ Y(t) &= \nu_\ell(Z(t)), \end{aligned} \quad (17)$$

where  $\ell$  is the index for which  $\mathbf{X}(t) = \mathbf{b}_\ell$  and  $\nu_\ell(\cdot)$ ,  $\nu_\ell^{-1}(\cdot)$  are a permutation, and its inverse, respectively. Since  $\nu_\ell(\cdot)$  is a permutation of  $0, \dots, M-1$ , the transformation is reversible, i.e., one can recover  $Y(t)$  from  $Z(t)$ . The re-aligned string  $\{Z^n\}$  is further modelled as a multinomial trial process with parameters  $P(Z=0) = \eta_0, \dots, P(Z=M-1) = \eta_{M-1}$ . The symbol  $i$  is observed in the string  $Z^n$  exactly  $\sum_{\ell=1}^K m_{\nu_\ell(i)}^\ell$  times, and thus the probability of the class label string is given by

$$P(Y^n | \mathbf{X}^n; \boldsymbol{\eta}, \boldsymbol{\nu}) = P(Z^n(\boldsymbol{\nu}); \boldsymbol{\eta}, \boldsymbol{\nu}) = \eta_0^{\sum_{\ell=1}^K m_{\nu_\ell(0)}^\ell} \dots \eta_{M-1}^{\sum_{\ell=1}^K m_{\nu_\ell(M-1)}^\ell}, \quad (18)$$

where the sequence of new data  $Z^n$  is determined by the set of permutations  $\boldsymbol{\nu} = \{\nu_i(\cdot) : i = 1, \dots, K\}$  as parameters, and the multinomial parameters of the sequence  $Z^n(\boldsymbol{\nu})$  are grouped in the vector  $\boldsymbol{\eta} = (\eta_0, \dots, \eta_{M-1})$ .

To make the set of pairs  $(\boldsymbol{\eta}, \boldsymbol{\nu})$  non-redundant, the vector  $\boldsymbol{\eta} = (\eta_0, \dots, \eta_{M-1})$  needs to be restricted such that  $\eta_0 \geq \eta_1 \geq \dots \geq \eta_{M-1}$  and it can be shown that with this constraint there is no reduction in the flexibility of the model. Finally, based on the above consideration the model class named here *discrete regression* is formalized as:

$$\begin{aligned} \mathcal{M}(\boldsymbol{\eta}, k, \boldsymbol{\nu}) &= \{P(Y^n | \mathbf{X}^n; \boldsymbol{\eta}, \boldsymbol{\nu}) : \boldsymbol{\eta} \in [0, 1]^M; \eta_0 \geq \eta_1 \geq \dots \geq \eta_{M-1}; \sum_{i=0}^{M-1} \eta_i = 1; \\ &\quad \boldsymbol{\nu} \in (\Upsilon_M)^K\}, \end{aligned} \quad (19)$$

where  $k$  is the dimensionality of the regression vectors  $\mathbf{X}(t)$  and  $K$  is the number of distinct vectors in the sequence  $\mathbf{X}^n$ . The key parameter to be determined during the feature selection process is the number of features  $k$ . In the following the optimal codes for the specified class of models will be obtained, and by computing the optimal description length for various values of  $k$  the minimum description length principle will be used to determine the optimal  $k^*$ .

The optimal description length for a strings  $y^n$  using the class  $\mathcal{M}(\boldsymbol{\eta}, k, \boldsymbol{\nu})$  can be computed by constructing first the normalized maximum likelihood model  $q(y^n | \mathbf{x}^n)$  for the class and then taking the description length to be  $\mathcal{L}(y^n | \mathbf{x}^n) = -\log_2 q(y^n | \mathbf{x}^n)$ . The normalized maximum likelihood (NML) model for a class of probability models can be obtained by first computing the maximized likelihood  $P(y^n | \mathbf{x}^n; \hat{\boldsymbol{\eta}}(y^n), \hat{\boldsymbol{\nu}}(y^n))$  using the ML parameters  $\hat{\boldsymbol{\eta}}(y^n)$ ,  $\hat{\boldsymbol{\nu}}(y^n)$  and then normalizing it to a probability distribution  $q(y^n | \mathbf{x}^n)$ . The optimality properties of the NML model for universal data compression have been established in [36].

To compute the ML parameters, the maximization problem is split into two subproblems, first optimize  $\boldsymbol{\nu}$  for a given  $\boldsymbol{\eta}$ , then optimize  $\boldsymbol{\eta}$  for the optimum  $\boldsymbol{\nu}^*(\boldsymbol{\eta})$ :

$$\max_{\boldsymbol{\eta}} \left[ \max_{\boldsymbol{\nu}} P(y^n | \mathbf{x}^n; \boldsymbol{\eta}, \boldsymbol{\nu}) \right]. \quad (20)$$

The first stage

$$\max_{\boldsymbol{\nu}} P(y^n | \mathbf{x}^n; \boldsymbol{\eta}, \boldsymbol{\nu}) = \max_{\nu_1(\cdot) \dots \nu_K(\cdot)} \prod_{\ell=1}^K \eta_0^{m_{\nu_\ell(0)}^\ell} \dots \eta_{M-1}^{m_{\nu_\ell(M-1)}^\ell} \quad (21)$$

can be immediately seen to decouple into  $K$  independent subproblems, one for each permutation  $\nu_\ell(\cdot)$ , and the optimal permutation is the permutation  $\hat{\nu}_\ell(\cdot)$  for which  $m_{\hat{\nu}_\ell(0)}^\ell \geq m_{\hat{\nu}_\ell(1)}^\ell \geq \dots m_{\hat{\nu}_\ell(M-1)}^\ell$ . The permutations  $\hat{\nu}_\ell(\cdot)$  are the ML set of permutations  $\hat{\boldsymbol{\nu}}$ , no matter what the values of  $\boldsymbol{\eta}$  are [44].

By ordering decreasingly the sequence of numbers  $m_0^j, \dots, m_{M-1}^j$  a new sequence can be defined:  $\hat{n}_0^j = m_{(M-1)}^j, \dots, \hat{n}_{M-1}^j = m_{(0)}^j$ , where the standard notation for the order statistics is used. The number of occurrences of the dominant symbol in each of the sets  $\{y_i : i \in \mathcal{I}_1\}, \dots, \{y_i : i \in \mathcal{I}_K\}$  is collected and the final pooled counts are obtained as  $n_0^* = \hat{n}_0^1 + \dots + \hat{n}_0^K$ . Similarly denote by  $n_j^* = \hat{n}_j^1 + \dots + \hat{n}_j^K$ , the total number of occurrences of the  $j$ 'th dominant symbol in each of the sets  $\{y_i : i \in \mathcal{I}_1\}, \dots, \{y_i : i \in \mathcal{I}_K\}$ .

By performing the outer maximization in (20) the optimal parameters  $\hat{\boldsymbol{\eta}}$  results to be  $\hat{\eta}_i = \frac{n_i^*}{n}$ , which is consistent with the assumed ranking,  $\eta_0 \geq \eta_1 \geq \dots \eta_{M-1}$ . The counts  $n_i^*(Y^n)$  depend on  $Y^n$  in a complicated manner through the order statistics of the counts  $m_i^\ell(Y^n)$ .

Since the ML values of the parameters are now available, the NML model in the model class  $\mathcal{M}(\boldsymbol{\eta}, k, \boldsymbol{\nu})$  can be defined as follows:

$$\begin{aligned} q(y^n | \mathbf{x}^n) &= \frac{P(y^n | \mathbf{x}^n; \hat{\boldsymbol{\eta}}(y^n), \hat{\boldsymbol{\nu}}(y^n))}{C_n(M_1, \dots, M_K)} \\ C_n(M_1, \dots, M_K) &= \sum_{w^n \in \{0, \dots, M-1\}^n} P(w^n | \mathbf{x}^n; \hat{\boldsymbol{\eta}}(w^n), \hat{\boldsymbol{\nu}}(w^n)) \\ &= \sum_{w^n \in \{0, \dots, M-1\}^n} \prod_{i=0}^{M-1} \left( \frac{n_i^*(w^n)}{n} \right)^{n_i^*(w^n)}. \end{aligned} \quad (22)$$

The computation of the normalizing constant directly from (22) is almost impossible for most gene expression data, since it requires the summation of  $M^n$  terms, but a more practical approach can be found, as described next.

The computations needed in (22) can be rearranged as a single sum as follows

$$C_n(M_1, \dots, M_K) = \sum_{n_0^* + n_1^* + \dots + n_{M-1}^* = n} S_{M_1, \dots, M_K}(n_0^*, n_1^*, \dots, n_{M-1}^*) \prod_{\ell=0}^{M-1} \left( \frac{n_\ell^*}{n} \right)^{n_\ell^*}, \quad (23)$$

where  $S_{M_1, \dots, M_K}(n_0^*, n_1^*, \dots, n_{M-1}^*)$  denotes the number of strings  $w^n$  having the same  $n_0^*(w^n), \dots, n_{M-1}^*(w^n)$ . The summation needs to be done over the set of numbers obeying  $n_0^* \geq n_1^* \geq \dots \geq n_{M-1}^*$ , which can be enforced by defining  $S_{M_1, \dots, M_K}(n_0^*, n_1^*, \dots, n_{M-1}^*) = 0$  for all the strings  $n_0^*, n_1^*, \dots, n_{M-1}^*$  which are not decreasing strings. The numbers  $S_{M_1, \dots, M_K}(n_0^*, n_1^*, \dots, n_{M-1}^*)$  can be computed recursively in  $K$ , according to the recurrence formula:

$$S_{M_1, \dots, M_K}(n_0^*, n_1^*, \dots, n_{M-1}^*) =$$

$$\begin{aligned}
&= \sum_{i_0+i_1+\dots+i_{M-1}=M_K} S_{M_1,\dots,M_{K-1}}(n_0^* - i_0, n_1^* - i_1, \dots, n_{M-1}^* - i_{M-1}) \times \\
&\times h(i_0, \dots, i_{M-1}) \tag{24}
\end{aligned}$$

where  $h(i_0, \dots, i_{M-1})$  denotes the number of ways in which a string having  $n' = M_1 + \dots + M_{K-1}$  letters and  $K - 1$  distinct regressor vectors can be extended to a string having  $n = M_1 + \dots + M_K$  letters and  $K$  distinct regressor vectors, such that in the set  $\{y_i : i \in \mathcal{I}_K\}$  the counts of symbols are  $i_0, \dots, i_{M-1}$ , regardless of order; it is also required that  $i_0 \geq i_1 \geq \dots \geq i_{M-1}$ . The newly introduced convolving sequences  $h()$  are defined as

$$h(i_0, \dots, i_{M-1}) = \binom{i_0 + \dots + i_{M-1}}{i_0, \dots, i_{M-1}} \binom{k_0 + \dots + k_{r-1}}{k_0, \dots, k_{r-1}} \tag{25}$$

in the case of a decreasing sequence of arguments  $i_1 \geq \dots \geq i_{M-1}$ , while for all other arguments ( $i_0, \dots, i_{M-1}$  not being decreasing) the sequence is  $h(i_0, \dots, i_{M-1}) = 0$ . In (25)  $r$  is the number of distinct values in the string  $i_0, \dots, i_{M-1}$  and  $k_0, \dots, k_{r-1}$  is the number of repetitions of the distinct values in the string  $i_0, \dots, i_{M-1}$ , respectively. For example with the arguments  $i_0 = 6, i_1 = 4, i_2 = 4, i_3 = 3, i_4 = 3$  the values occurring in (25) are  $r = 3, k_0 = 2, k_1 = 2, k_2 = 1$ .

The computation of  $\mathcal{L}(y^n | \mathbf{x}^n)$  can be accomplished very fast, the computation of the normalization constant is most demanding, but its evaluation with (23), by means of the convolution sums (24), is very fast, its implementation in Matlab is run in less than 1 second.

The search for the best features set, the one that minimizes the description length  $\mathcal{L}(y^n | \mathbf{x}^n)$  can be performed in any of the traditional ways for wrapper methods, as described in Section 4.2.

The classification model  $\hat{y} = g(x_{i_1}, \dots, x_{i_k})$ , discovered during the feature selection process can be extended to cases which were absent in the training set, to obtain a well defined classifier. For unseen cases the decision is taken according to the class labels (votes) of the nearest neighbors at Hamming distance  $d$ , where  $d$  is the smallest value at which a class label is a definite winner, for examples see [46].

The presented MDL method can be considered to be a wrapper method where the optimization criterion is not the classification accuracy of a specific classifier, but the description length achieved when using the information from the feature set. The selection process is motivated by an information theoretic principle, and thus the method can be seen as a powerful tool for discovering informative feature sets, which are very likely to have biological significance. However, the method can be used also as a filter stage, after which the best classifier in a certain class can be easily designed and tested for classification accuracy.

## 7 Conclusions

Feature selection is an involved process, which needs knowledge of the available techniques for guiding which tool to be used and for assessing correctly the performance for the application at hand. An impressive number of studies of feature selection techniques for gene expression data has shown that relevant biological information can be gathered using various feature selection techniques, at a computational cost

which is affordable with the current computer technology. Future studies will most likely reduce even further the computational cost of the methods, making it possible to compare larger candidate feature sets. As another challenge for the future, the biological interpretation of the feature sets needs to be integrated within the feature selection methods themselves, and not used as they are now, just as a validation stage after the feature selection process was finished.

## References

- [1] A. Barron, J. Rissanen, and B. Yu. The minimum description length principle in coding and modeling. *IEEE Trans. on Information Theory, Special commemorative issue: Information Theory 1948-1998*, vol.44, no. 6, 2743–2760, 1998.
- [2] C. Bhattacharyya, L. R. Grate, A. Rizki, D. Radisky, F. J. Molina, M. I. Jordan, M. J. Bissell, and I. S. Mian. Simultaneous relevant feature identification and classification in high-dimensional spaces: Application to molecular profiling data. *Signal Processing, Special issue on Genomic Signal Processing*, Vol. 83, No.4, pp. 729–743, 2003.
- [3] T. Cover and J. Van Campenhout. On the possible orderings in the measurement selection process. *IEEE Transactions on System, Man, and Cybernetics*, 7: 657-661, 1977.
- [4] L. Devroye, L. Györfi, and G. Lugosi. *A probabilistic theory of Pattern Recognition*. Springer Verlag, 1996.
- [5] C. Ding and H. Peng. Minimum redundancy feature selection from microarray gene expression data. *Proceedings of the Computational Systems Bioinformatics (CSB03)*, pp. 523-529, 2003.
- [6] E.R. Dougherty, S. Kim, and Y. Chen. Coefficient of determination in nonlinear signal processing. *Signal Processing*, vol. 80, pp. 2219–2235, 2000.
- [7] S. Dudoit, J. Fridlyand, and T.P. Speed. Comparison of discrimination methods for the classification of tumors using gene expression data. Technical Report 576, Dept. of Statistics, University of California, Berkely, 2000.
- [8] G. Fuller, K. Hess, C. Mircean, I. Tabus, I. Shmulevich, C. Rhee, K. Aldape, J. Bruner, A. Sawaya, and W. Zhang. Human glioma diagnosis from gene expression data. In *Computational And Statistical Approaches To Genomics* (W. Zhang and I. Shmulevich, eds.), Kluwer Academic Publishers, p. 241–256, 2002.
- [9] C. Furlanello, M. Serafini, S. Merler, and G. Jurman. Entropy-based gene ranking without selection bias for the predictive classification of microarray data. *BMC Bioinformatics*, 4:54, 2003.
- [10] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. *Machine Learning*, 46, 1-3, 389–422, 2002. See also *Erratum* at <http://clopinet.com/isabelle/Papers/RFE-erratum.html>.

- [11] Simon Haykin. *Adaptive filter theory*. Prentice Hall International, 1992.
- [12] K. Imai and D.A. van Dyk. A Bayesian analysis of the multinomial probit model using marginal data augmentation. *Journal of Econometrics*, to appear 2004, <http://www.princeton.edu/~kimai/research/mnp.html>.
- [13] I. Inza, B. Sierra, R. Blanco, and P. Larrañaga. Gene selection by sequential search wrapper approaches in microarray cancer class prediction. [cite-seer.nj.nec.com/541379.html](http://citeseer.nj.nec.com/541379.html).
- [14] J. Jaeger, R. Sengupta, and W.L. Ruzzo. Improved gene selection for classification of microarrays. *Pacific Symposium on Biocomputing*, Kauai, Hawaii, Jan., 2003.
- [15] R. Jörnsten and B. Yu. Simultaneous gene clustering and subset selection for sample classification via MDL. *Bioinformatics*, vol. 19, no. 9, pp. 1100–1109, 2003.
- [16] D.L. Donoho and M. Elad. Optimally-sparse representation in general (non-orthogonal) dictionaries via  $\ell^1$  minimization. Technical Report Stanford University, 2002. available at <http://www-stat.stanford.edu/~donoho/reports.html>
- [17] E.R. Dougherty, S. Kim and Y. Chen. Coefficient of determination in nonlinear signal processing. *Signal Processing* 80, 2219–2235, 2000.
- [18] S. Geisser. The predictive sample reuse method with applications. *Journal of the American Statistical Association* 70(350):320–328, 1975.
- [19] R.F. Hashimoto, E.R. Dougherty, M. Brun, Z.Z. Zhou, M. L. Bittner and J.M. Trent. Efficient selection of feature sets possessing high coefficients of determination based on incremental determinations. *Signal Processing, Special issue on Genomic Signal Processing*, Vol. 83, No.4, pp. 695–712, April, 2003.
- [20] S. Kim, E.R. Dougherty, Y. Chen, K. Sivakumar, P. Meltzer, J.M. Trent, and M. Bittner. Multivariate measurement of gene expression relationships. *Genomics* 67, 201-209, 2000.
- [21] S. Kim, E.R. Dougherty, J. Barrera, Y. Chen, M.L. Bittner, and J.M. Trent. Strong feature sets from small samples. *J. Comput. Biol.*, 9, 127-146, 2002.
- [22] R. Kohavi and G. John. Wrapper for feature subset selection. *Artificial Intelligence*, 97, 273–324, 1997.
- [23] A.N. Kolmogorov. Three approaches to the quantitative definition of information. *Problems Inform. Transmission* 1(1), 1–7, 1965.
- [24] B. Krishnapuram, L. Carin, A.J. Hartemink. Joint classifier and feature optimization for comprehensive cancer diagnosis using gene expression data. *Journal of Computational Biology*, (to appear 2004).
- [25] B. Krishnapuram, A.J. Hartemink, and L. Carin. Applying logistic regression and RVM to achieve probabilistic cancer diagnosis from gene expression profiles. *GENSIPS: Workshop on Genomic Signal Processing and Statistics*, October 2002.

- [26] K.E. Lee, N. Sha, E.R. Dougherty, M. Vannucci, and B.K. Mallick. Gene selection: a Bayesian variable selection approach. *Bioinformatics*, vol. 19, no. 1, pp.90-97, 2003.
- [27] L.Li, C.R. Weinberg, T.A. Darden and L.G. Pedersen. Gene selection for sample classification based on gene expression data: study of sensitivity to choice of parameters of the GA/KNN method. *Bioinformatics*, vol. 17, no. 12, pp.1131-1142, 2001.
- [28] W. Li, I. Grosse. Gene selection criterion for discriminant microarray data analysis based on extreme value distributions. *RECOMB 2003*: pp. 217-223, 2003.
- [29] P. Narendra and K. Fukunaga. A branch and bound algorithm for feature subset selection. *IEEE Transactions on Computers*: 26(9):917-922, 1977.
- [30] P. Pavlidis, J. Weston, J. Cai and W.N. Grundy. Learning gene functional classifications from multiple data types. *Journal of Computational Biology*. 9(2):401-411, 2002.
- [31] A. Rakotomamonjy. Variable selection using SVM-based criteria. *Journal Machine Learning Research*, Special Issue on Variable and Feature Selection, 3(Mar):1357-1370, 2003.
- [32] J. Reunanen. Overfitting in making comparisons between variable selection methods. *Journal Machine Learning Research*, Special Issue on Variable and Feature Selection, 3(Mar):1371-1382, 2003.
- [33] J. Rissanen. Modelling by shortest data description. *Automatica*, vol. 14, pp. 465-471, 1978.
- [34] J. Rissanen. Stochastic complexity and modeling. *Ann. Statist.*, vol. 14, pp. 1080-1100, 1986.
- [35] J. Rissanen. Fisher information and stochastic complexity. *IEEE Trans. on Information Theory*, vol.42, No. 1, pp. 40-47, 1996.
- [36] J. Rissanen. Strong optimality of the normalized ML models as universal codes and information in data. *IEEE Trans. on Information Theory*, vol.IT-47, No. 5, 1712-1717, 2001.
- [37] N. Sha, M. Vannucci, P.J. Brown, M.K. Trower, G Amplett, and F. Falciani. Gene selection in arthritis classification with large-scale microarray expression profiles. *Comparative and functional genomics*, 4, 171-181, 2003.
- [38] B. Scholkopf, I. Guyon, and J. Weston. Statistical learning and kernel methods in bioinformatics. In proceedings *NATO Advanced Studies Inst. on Artificial Intelligence and Heuristics Methods for Bioinformatics*, San Miniato, Italy October 1-11, 2001. <http://www.clopinet.com/isabelle/Papers/kerbioinfo.pdf>
- [39] R. Solomonoff. A formal theory of inductive inference, part 1 and part 2. *Information and Control* 7, 1-22, 224-254, 1964.

- [40] T. Söderström and P. Stoica. *System identification*. Prentice Hall, 1989.
- [41] M. Stone. Cross-validatory choice and assessment of statistical prediction. *Journal of the Royal Statistical Society*, Vol. 36, No. 1, 111-147, 1974.
- [42] M. Stone. Asymptotics for and against cross-validation. *Biometrika*, 64:29–35, 1977.
- [43] I. Tabus and J. Astola. On the Use of MDL Principle in Gene Expression Prediction. *Journal of Applied Signal Processing*, Volume 2001, No. 4, pp. 297-303, December 2001.
- [44] I. Tabus, J. Rissanen, and J. Astola. Classification and feature gene selection using the normalized maximum likelihood model for discrete regression. *Signal Processing, Special issue on Genomic Signal Processing*, Vol. 83, No.4, pp. 713-727, April, 2003.
- [45] I. Tabus, C. Mircean, W. Zhang, I. Shmulevich, and J. Astola. Transcriptome-based glioma classification using informative gene set. In *Genomic and molecular neuro-oncology* (W. Zhang and G. Fuller, eds), Jones and Bartlett Publishers, pp. 205-220, 2003.
- [46] I. Tabus, J. Rissanen, and J. Astola. Normalized maximum likelihood models for Boolean regression with application to prediction and classification in genomics. In *Computational And Statistical Approaches To Genomics* (W. Zhang and I. Shmulevich, eds.), Kluwer Academic Publishers, pp. 173-196, 2002.
- [47] K. Torkkola. Feature extraction by non-parametric mutual information maximization. *Journal Machine Learning Research*, Special Issue on Variable and Feature Selection, 3(Mar):1415-1438, 2003.
- [48] J. Weston, S. Mukherjee, O. Chapelle, M. Pontil, T. Poggio, and V. Vapnik. Feature selection for SVMs. In *Advances in Neural Information Processing Systems*, vol. 13, MIT Press, Cambridge, MA, pp. 668-674, 2000.
- [49] J. Weston, A. Elisseeff, B. Schölkopf, and M. Tipping. Use of the zero-norm with linear models and kernel methods. *Journal Machine Learning Research*, Special Issue on Variable and Feature Selection, 3(Mar):1439-1461, 2003.
- [50] E.P. Xing, M. I. Jordan, and R. M. Karp. Feature selection for high-dimensional genomic microarray data. *Machine Learning: Proceedings of the Eighteenth International Conference*, San Mateo, CA: Morgan Kaufmann, 2001.
- [51] M. Xiong, W. Li, J. Zhao, L. Jin, and E. Boerwinkle. Feature (gene) selection in gene expression-based tumor classification. *Mol. Genet. Metab.*, 73, 239247, 2001.
- [52] X. Zhang, W.H. Wong. Recursive sample classification and gene selection based on SVM: method and software description. Technical Report, Department of Biostatistics, Harvard School of Public Health, 2001.

- [53] X. Zhou, X. Wang, and E.R. Dougherty. Gene prediction using multinomial probit regression with Bayesian gene selection. *EURASIP Journal Applied Signal Processing*, No.1 pp. 115–124, 2004.
- [54] X. Zhou, X. Wang, E.R. Dougherty. Construction of genomic networks using mutual-information clustering and reversible -jump Markov-chain-Monte-Carlo predictor design. *Signal Processing, Special issue on Genomic Signal Processing*, Vol. 83, No.4, pp. 745–761, 2003.

Figure 1: The estimation of the classification accuracy of a given feature set  $X_{i_1}, \dots, X_{i_m}$  by a three-folded cross-validation experiment: the training set  $\mathcal{D} = \{(X_1, \dots, X_N, Y)\}$  is split into three equally sized subsets  $\mathcal{D}_A, \mathcal{D}_B, \mathcal{D}_C$ . The classifier parameters are fitted to the data provided by two of the subsets of  $\mathcal{D}$ , and the accuracy is estimated over the remaining subset, and the process is repeated for all ways of taking two out of three subsets. The computed average classification error,  $\hat{\varepsilon}|_{\{i_1, \dots, i_m\}}$  is used to direct the search for the most accurate feature set,  $X_{i_1^*}, \dots, X_{i_{m^*}^*}$  and its size  $m^*$ .

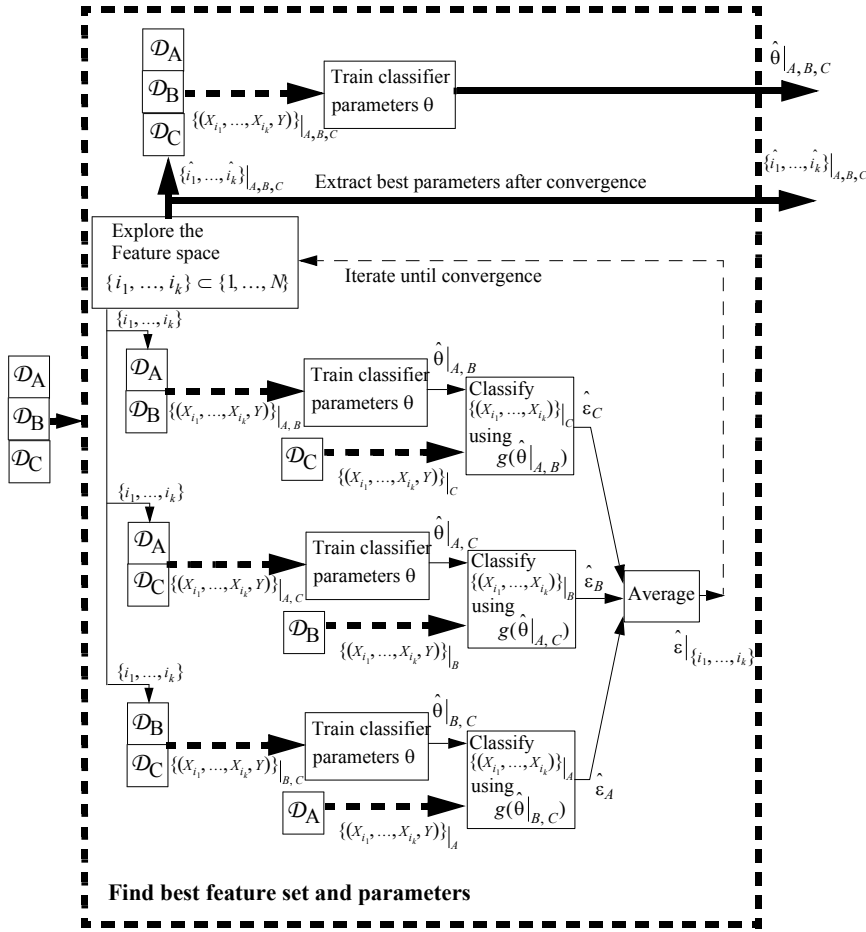


Figure 2: Overall estimation of the classification accuracy within a four-folded cross-validation experiment. The blocks “Find the best feature set and parameters” operate on three of the folds (as depicted in Fig. 1), while the classification accuracy is estimated over the fourth fold, to avoid over-fitting.

