ELSEVIER International Journal of Approximate Reasoning 27 (2001) 143–164

INTERNATIONAL JOURNAL OF APPROXIMATE REASONING

www.elsevier.com/locate/ijar

Feature subset selection by Bayesian networks: a comparison with genetic and sequential algorithms

Iñaki Inza *, Pedro Larrañaga, Basilio Sierra

Department of Computer Science and Artificial Intelligence, University of the Basque Country, P.O. Box 649, E-20080 Donostia – San Sebastián, Basque Country, Spain

Received 1 September 2000; accepted 1 March 2001

Abstract

In this paper we perform a comparison among FSS–EBNA, a randomized, population-based and evolutionary algorithm, and two genetic and other two sequential search approaches in the well-known feature subset selection (FSS) problem. In FSS–EBNA, the FSS problem, stated as a search problem, uses the estimation of Bayesian network algorithm (EBNA) search engine, an algorithm within the estimation of distribution algorithm (EDA) approach. The EDA paradigm is born from the roots of the genetic algorithm (GA) community in order to explicitly discover the relationships among the features of the problem and not disrupt them by genetic recombination operators. The EDA paradigm avoids the use of recombination operators and it guarantees the evolution of the population of solutions and the discovery of these relationships by the factorization of the probability distribution of best individuals in each generation of the search. In EBNA, this factorization is carried out by a Bayesian network induced by a cheap local search mechanism. FSS–EBNA can be seen as a hybrid Soft Computing system, a synergistic combination of probabilistic and evolutionary computing to solve the FSS task. Promising results on a set of real Data Mining domains are achieved by FSS–EBNA in the comparison respect to well-known genetic and sequential search algorithms.

Keywords: Feature subset selection; Estimation of distribution algorithm; Soft computing; Estimation of Bayesian network algorithm; Bayesian network; Predictive accuracy

* Corresponding author. Tel.: +34-943018000 ext. 5026; fax: +34-943219306.
E-mail address: ccbincai@si.ehu.es (I. Inza).

0888-613X/01/$ - see front matter
PII: S0888-613X(01)00038-X

1. Motivation

The basic problem of supervised Classification in Data Mining is concerned with the induction of a model that classifies a given object into one of the several known classes. In order to induce the classification model, each object is described by a pattern of d features. With advanced computer technologies, big data archives are usually formed and many features are used to describe the objects. Here, Data Mining and Machine Learning communities usually formulate the following question: *Are all of these d descriptive features useful for learning the classification model?* On trying to respond to this question, we come up with the feature subset selection (FSS) approach, which can be reformulated as follows: *given a set of candidate features, select the best subset in a classification task*.

The dimensionality reduction made by an FSS process can carry out several advantages for a classification system in a specific task:

- a reduction in the cost of acquisition of the data;
- improvement of the comprehensibility of the final classification model;
- a faster induction of the final classification model;
- an improvement in classification accuracy.

The attainment of higher classification accuracies, coupled with a notable dimensionality reduction, is the common objective of Machine Learning and Data Mining processes.

It has long been proved that the classification accuracy of supervised classifiers is not monotonic with respect to the addition of features. Irrelevant or redundant features, depending on the specific characteristics of the supervised classifier, may degrade the predictive accuracy of the classification model.

FSS can be viewed as a search problem, with each state in the search space specifying a subset of the possible features of the task. Exhaustive evaluation of possible feature subsets is usually unfeasible in practice because of the large amount of computational effort required. Due to its randomized, evolutionary and population-based nature, genetic algorithms (GAs) have been the most commonly used search engine in the FSS process [32,54,58]. Most of the theory of GAs deals with the so-called building blocks (BBs) [18]; simply said, by BBs, partial solutions of a problem are meant, formed by groups of related variables. GAs reproduce BBs by an implicit manipulation of a large number of them by mechanisms of selection and recombination. A crucial factor of the GA success resides in the proper growth and mixing of the optimal BBs of the problem. Problem-independent recombination operators often break these BBs and do not mix them efficiently; thus, this could delay the discovery of the global optima or produce a convergence to a local optima.

Linkage learning (LL) [20] is the identification of the BBs to be conserved under recombination. Recently, various approaches to solve the LL problem have been proposed. Several proposed methods are based on the manipulation

of the representation of solutions during the optimization to make the interacting components of partial solutions less likely to be broken. For this purpose, various reordering and mapping operators are used. One of these approaches is the well-known Messy genetic algorithm (MGA) [18].

Instead of extending the GA, in later years, a new approach has strongly emerged under the estimation of ditribution algorithm (EDA) [37,43]. EDA approach explicitly learns the probabilistic structure of the problem and uses this information to ensure a proper mixing and growth of BBs that do not disrupt them. The further exploration of the search space is guided, instead of crossover and mutation operators as in GAs, by the probabilistic modelling of promising solutions.

Bearing these difficulties in mind, in this paper we perform a comparison in the FSS task among FSS–EBNA [25], a novel algorithm inspired on the EDA approach, and other classic genetic and sequential search algorithms. FSS–EBNA (Feature Subset Selection by Estimation of Bayesian Network Algorithm) is based on the probabilistic modelling by Bayesian networks (BNs) of promising solutions (feature subsets, in our case) to guide further exploration of the space of features. This BN is sampled to obtain the next generation of feature subsets, therefore, guaranteeing the evolution of the search for the optimal subset of features. Using a wrapper approach to evaluate the goodness of each candidate feature subset, we search for the feature subset which maximizes the predictive accuracy of the Naive–Bayes supervised classifier on presented tasks. However, this paper presents the first comparison among FSS–EBNA and other classic search algorithms in the FSS task.

FSS–EBNA can be seen as a hybrid Soft Computing [7] system, a synergistic combination of probabilistic and evolutionary computing to solve the FSS task: the use of a classic probabilistic paradigm as BNs in conjunction with the evolutionary strategy scheme of the EDA paradigm, enables us to construct an application which tackles the FSS problem. The remainder of the paper is organized as follows: Section 2 introduces the FSS problem and its basic components. Section 3 introduces the EDA approach, BNs and the combination of both, the EBNA search engine. Section 4 presents the application of EBNA to the FSS problem, resulting in the FSS–EBNA algorithm. In Section 5, we compare it against four well-known FSS methods, specially focusing our attention on the comparison against two genetic approaches. The last section picks up the summary of our work and some ideas about future lines of research.

2. Feature subset selection as a search problem

Even if we locate our work as a Machine Learning or Data Mining approach, the FSS literature includes plenty of works in other fields such as

Pattern Recognition [27,29], Statistic [40,46] or Text-Learning [41]. In the BN community we have the example of the work of Provan and Singh [51], who, using a BN as a classifier, build it selecting in a greedy manner the variables that should maximize the predictive accuracy of the network. Thus, different communities have exchanged and shared ideas among them to deal with the FSS problem.

As reported by Aha and Bankert [2], the objective of FSS in Machine Learning and Data Mining is to *reduce the number of features used to characterize a dataset so as to improve a classifier's performance on a given task*. In our framework we will appreciate a tradeoff between a high-predictive-accuracy and a low cardinality (selection of few features) in the finally chosen feature subset. The FSS task can be exposed as a search problem, each state in the search space identifying a subset of possible features.

Thus, any FSS algorithm must determine four basic issues that define the nature of the search process (see [39] for a review of FSS algorithms):

1. *The starting point in the space*. It determines the direction of the search. One might start with no features and successively add them, or one might start with all features and successively remove them. One might also select an initial state somewhere in the middle of the search space.

2. *The organization of the search*. It determines the strategy of the search. Roughly speaking, the search strategies can be *complete* or *heuristic*. The basis of the *complete search* is the systematic examination of every possible feature subset. Three classic complete search implementations are depth-first, breadth-first and branch and bound search [46]. On the other hand, among *heuristic* algorithms, there are *deterministic heuristic* algorithms and *non-deterministic heuristic* ones. Classic deterministic heuristic FSS algorithms are sequential forward selection (SFS) and sequential backward elimination (SBE) [29], floating selection methods [52] and best-first search [30]. They are deterministic in the sense that all the runs always obtain the same solution. *Non-deterministic heuristic* search is used to escape from local maximums. Randomness is used for this purpose and this implies that one should not expect the same solution from different runs. Two classic implementations of non-deterministic search engines are the frequently applied GAs [54] and Simulated Annealing [15].

3. *Evaluation strategy of feature subsets*. By the calculation of the goodness of each proposed feature subset the evaluation function identifies the promising areas of the search space. The objective of FSS algorithm is its maximization. The search algorithm uses the value returned by the evaluation function to guide the search. Some evaluation functions carry out this objective looking only at the intrinsic characteristics of the data and measuring the power of a feature subset to discriminate among the classes of the problem: this type of evaluation functions is grouped below the *filter* strategy. These evaluation functions are usually monotonic and increase with the addition of features that afterwards can hurt the predictive accuracy of the final classifier. However,

Kohavi and John [30] reported that when the goal of FSS is the maximization of the accuracy, the features selected should depend not only on the features and the target concept to be learned, but also on the special characteristics of the supervised classifier. Thus, they proposed the *wrapper* concept: this implies that the FSS algorithm conducts the search for a good subset using the classifier itself as a part of the evaluation function, the same classifier that will be used to induce the final classification model. Once the classification algorithm is fixed, the idea is to train it with the feature subset found by the search algorithm, estimating the predictive accuracy on the training set and assigning it as the value of the evaluation function of the feature subset.

4. *Criterion for halting the search*. An intuitive approach for stopping the search will be the non-improvement of the evaluation function value of alternative subsets. Another classic criterion will be to fix an amount of possible solutions to be visited along the search.

3. EDA paradigm, Bayesian Networks and EBNA approach

In this section, EDA paradigm and BNs will be explained. Bearing these two concepts in mind, EBNA, the search engine used in our FSS algorithm will be presented. EDA paradigm is the general formula of the EBNA algorithm; on the other hand, BNs can be seen as the most important basis of EBNA.

3.1. The need of the EDA paradigm

Many combinatorial optimization algorithms have no mechanism for capturing the relationships among the variables of the problem. The related literature has many papers proposing different heuristics in order to implicitly capture these relationships. GAs implicitly capture these relationships by concentrating samples on combinations of high-performance members of the current population through the use of the crossover operator. Crossover combines the information contained within pairs of selected 'parent' solutions by placing random subsets of each parent's bits into their respective positions in a new 'child' solution. In GAs no explicit information is kept about which groups of variables jointly contribute to the quality of candidate solutions. The crossover operation is randomized and could disrupt many of these relationships among the variables; therefore, most of the crossover operations yield unproductive results and the discovery of the global optima could be delayed.

On the other hand, GAs are also criticized in the literature for three aspects [36]:

- the large number of parameters and their associated referred optimal selection or tuning process;

1. $D_0 \leftarrow$ Generate N individuals (the initial population) randomly.
2. Repeat for $l = 1, 2, \ldots$ until a stop criterion is met:
2.1. $D^s_{l-1} \leftarrow$ Select $S \leq N$ individuals from D_{l-1} according to a selection method.
2.2. $p_l(\mathbf{x}) = p(\mathbf{x}|D^s_{l-1}) \leftarrow$ Estimate the joint probability distribution of an individual being among the selected inviduals.
2.3. $D_l \leftarrow$ Sample N individuals (the new population) from $p_l(\mathbf{x})$.

Fig. 1. Main scheme of the EDA paradigm.

- the extremely difficult prediction of the movements of the populations in the search space;
- their incapacity to solve the well-known deceptive problems [18].

Linkage Learning is the identification of groups of variables (or BBs) that are related. Instead of extending the GA, the idea of the explicit discovery of these relationships during the optimization process itself has emerged from the roots of the GA community. One way to discover these relationships is to estimate the joint distribution of promising solutions and use this estimate in order to generate new individuals. [1] A general scheme of the algorithms based on this principle is called the estimation of distribution algorithm (EDA) [43]. In EDA (see Fig. 1), there are no crossover nor mutation operators, and the new population is sampled from a probability distribution which is estimated from the selected individuals. However, estimating the distribution is a critical task in EDA (for a more complete review see [36]).

The simplest way to estimate the distribution of good solutions assumes the independence between the features [2] of the problem. New candidate solutions are sampled by only regarding the proportions of the values of the variables independently to the remaining ones. Population-based incremental learning (PBIL) [5], compact genetic algorithm (cGA) [20], univariate marginal distribution algorithm (UMDA) [42] and bit-based simulated crossover (BSC) Syswerda [57] are four algorithms of this type. They work well in problems with no significant interactions (or relationships) among features and so, the need for covering higher-order interactions among the variables is seen for more complex or real tasks.

Efforts covering pairwise interactions among the features of the problem have generated algorithms such as: MIMIC uses simple chain distributions [14], the so-called dependency trees [6] and the bivariate marginal distribution algorithm (BMDA) [50]. However, Pelikan and Müehlenbein [50] have demonstrated that only covering pairwise dependencies is not enough with problems which have higher-order interactions: in this type of problems the

[1] In a search algorithm the terms 'individual' and 'solution' can be used indistinctly.

[2] In the Evolutionary Computation community, the term 'variable' is normally used instead of 'feature'. We use both terms indistinctly.

discovery of the global optimum is considerably delayed when lower-order-based probability estimating approaches are used.

In this way, the factorized distribution algorithm (FDA) [44] covers higher-order interactions. This is done using a previously fixed factorization of the joint probability distribution. However, FDA needs prior information on the decomposition and factorization of the problem which is often not available. Without the need of this extra information about the decomposition and factorization of the problem, other probabilistic models, able to cover higher-order interactions, appear in the literature. For this purpose, BNs, graphical representations able to cover these higher-order interactions, can be used. Thus, EBNA [16] and BOA [49] are algorithms which use different implementations of BNs for estimating the joint distribution of promising solutions. In this way, multivariate interactions among variables can be covered. In [56] a BN with a polytree structure is proposed: the proposed algorithm is called polytree approximation of distribution algorithms (PADA) and it is based on a method proposed by Acid and de Campos [1] for detecting relationships among variables. Ochoa et al. [47] propose an initial algorithm to learn (from conditional independence tests) the junction tree from a database. Harik presents an algorithm extend compact genetic algorithm (EcGA) [21], whose basic idea consists of factorizing the joint probability distribution in a product of marginal distributions of variable size: these marginal distributions of variable size are related to the variables that are contained in the same group and to the probability distributions associated to them.

Many of these works notify a faster discovery of the global optimum by EDA algorithms than GAs for certain combinatorial optimization problems. Harik [21] and Pelikan and Müehlenbein [50] show several intuitive and well-known problems where the GAs, when the crossover operator is applied, frequently disrupt optimum relationships among features: in this way, these optimum relationships that appear in the parent solutions will disappear in the children solutions and the discovery of the global optimum will be delayed. Thus, these authors note that EDA approaches are able to first discover and then maintain these relationships during the entire optimization process, producing a faster discovery of the global optimum than GAs.

In order to avoid the evaluation of larger amounts of possible solutions (and its associated CPU time), a faster discovery of well-fitted areas in the search space is highly desired in the FSS problem. Bearing this purpose in mind, we use an EDA-inspired approach for FSS. Based on the EBNA work of Etxeberria and Larrañaga [16], we propose the use of BNs as the models for representing the probability distribution of the set of candidate solutions for the FSS problem, using the application of automatic learning methods to induce the right distribution model in each generation in an efficient way.

3.2. Bayesian networks

A Bayesian network (BN) [9,38,48] encodes the relationships contained in the modelled data. It can be used to describe the data as well as to generate new instances of the variables with similar properties as those of given data. A BN encodes the probability distribution $p(\mathbf{x})$, where $\mathbf{X} = (X_1, \ldots, X_d)$ is a vector of variables, and it can be seen as a pair $(M, \boldsymbol{\theta})$. M is a directed acyclic graph (DAG) where the nodes correspond to the variables and the arcs represent the conditional (in)dependencies [3] [13] among the variables. By X_i, both the variable and the node corresponding to this variable is denoted. M will give the factorization of $p(\mathbf{x})$:

$$p(\mathbf{x}) = \prod_{i=1}^{d} p(x_i \,|\, \pi_i),$$

where Π_i is the set of parent variables (nodes) that X_i has in M and π_i its possible instantations. The number of states of Π_i will be denoted as $|\Pi_i| = q_i$ and the number of different values of X_i as $|X_i| = r_i$. $\boldsymbol{\theta} = \{\theta_{ijk}\}$ are the required conditional probability values to completely define the joint probability distribution of $\mathbf{X}$. θ_{ijk} will represent the probability of X_i being in its kth state while Π_i is in its jth instantation. This factorization of the joint distribution can be used to generate new instances using the conditional probabilities in a modelled dataset.

Informally, an arc between two nodes relates these nodes so that the value of the variable corresponding to the ending node of the arc depends on the value of the variable corresponding to the starting node. Every probability diztribution can be defined by a BN [48]. As a result, BNs are widely used in problems where uncertainty is handled using probabilities.

Unfortunately, finding the optimal $\hat{M}$ requires searching through all possible structures, which has been proven to be NP-hard [11]. Although promising results have been obtained using global search techniques [12,34,35,59] their computation cost makes them unfeasible for our problem. We need to find $\hat{M}$ as fast as possible, so a simple algorithm which returns a good structure, even if it is not optimal, will be preferred.

In our implementation Algorithm B [8] is used for learning BNs from data. Algorithm B is a greedy search heuristic. The algorithm starts with an arc-less structure in each generation of the search and at each step, it adds the arc with the maximum increase in the BIC approximation (penalized maximum likelihood of the proposed structure [55]). The algorithm stops when adding an arc does not increase the utilized measure. In the BIC metric, the penalized function is given by the Jeffreys–Schwarz criterion, resulting in the Bayesian

[3] Let $\mathbf{Y}, \mathbf{Z}$ and $\mathbf{W}$ be three disjoint sets of variables, then $\mathbf{Y}$ is said to be conditionally independent of $\mathbf{Z}$ given $\mathbf{W}$, if and only if $p(\mathbf{y} \,|\, \mathbf{z}, \mathbf{w}) = p(\mathbf{y} \,|\, \mathbf{w})$, for all possible values $\mathbf{y}, \mathbf{z}$ and $\mathbf{w}$ of $\mathbf{Y}, \mathbf{Z}$ and $\mathbf{W}$.

1. $BN_0 \leftarrow (M_0, \boldsymbol{\theta_0})$.
2. $D_0 \leftarrow$ Sample N individuals from BN_0.
3. For $l = 1, 2, \ldots$ until a stop criterion is met:
3.1. $D^s_{l-1} \leftarrow$ Select S individuals from D_{l-1}.
3.2. $\hat{M}_l \leftarrow$ Find the structure which maximizes $BIC(M_l, D^s_{l-1})$.
3.3. $\boldsymbol{\theta_l} \leftarrow$ Calculate $\{\theta^l_{ijk} = \frac{N^{l-1}_{ijk}+1}{N^{l-1}_{ij}+r_i}\}$ using D^s_{l-1} as the data set.
3.4. $BN_l \leftarrow (\hat{M}_l, \boldsymbol{\theta_0})$.
3.5. $D_l \leftarrow$ Sample N individuals from BN_l using PLS.

Fig. 2. EBNA basic scheme.

information criterion (BIC) [55]. The BIC score of a BN structure M, from a database D and containing N cases (characterized by d variables) is denoted as follows:

$$\mathrm{BIC}(M, D) = \sum_{i=1}^{d} \sum_{j=1}^{q_i} \sum_{k=1}^{r_i} N_{ijk} \log \frac{N_{ijk}}{N_{ij}} - \frac{\log N}{2} \sum_{i=1}^{d} (r_i - 1) q_i.$$

N_{ijk} denotes the number of cases in D in which the variable X_i is instantiated to its kth value and its parents in the network structure are instantiated to their jth value. Let $N_{ij} = \sum_{k=1}^{r_i} N_{ijk}$, where r_i and q_i represent the number of possible states of X_i and its parents, respectively.

3.3. Estimation of Bayesian network algorithm: EBNA

The general procedure of EBNA appears in Fig. 2. To understand the steps of the algorithm, the following concepts must be clarified.

M_0 is the DAG with no arcs at all and $\boldsymbol{\theta}_0 = \{\forall i : p(X_i = x_i) = 1/r_i\}$, which means that the initial Bayesian network BN_0, assigns the same probability to all individuals. N is the number of individuals in the population. S is the number of individuals selected from the population. Although S can be any value, the most common value in the EDA literature is $S = N/2$. If S is close to N then the populations will not evolve very much from generation to generation. On the other hand, a low S value will lead to low diversity resulting in early convergence.

For individual selection range-based selection is proposed, i.e., selecting the best $N/2$ individuals from the N individuals of the population. However, any selection method could be used. Probabilistic logic sampling algorithm (PLS) [22] is used to sample new individuals from the BN.

Finally, the way in which the new population is created must be pointed out. In the given procedure, all individuals from the previous population are discarded and the new population is composed of all newly created individuals. [4]

[4] Newly created individuals are also known as 'offspring'.

This has the problem of losing the best individuals that have been previously generated, therefore, the following minor change has been made: instead of discarding all the individuals, we maintain the best individual of the previous generation and create $N-1$ new individuals.

An elitist approach is used to form iterative populations. Instead of directly discarding the $N-1$ individuals from the previous generation replacing them with $N-1$ newly generated ones, the $2N-1$ individuals are put together and the best $N-1$ taken among them. These best $N-1$ individuals form the new population together with the best individual of the previous generation. In this way, the populations converge faster to the best individuals found; however, this also implies a risk of losing diversity within the population.

The stopping criteria will be explained in the next section.

4. Feature subset selection by estimation of BN algorithm: FSS–EBNA. Instantation of FSS components

Once the FSS problem and EBNA algorithm are presented, we will use the search engine provided by EBNA to solve the FSS problem, resulting in the FSS–EBNA algorithm. In the next lines we will briefly describe its basic components (see [25] for more details).

Parametrizing the FSS task as an optimization problem, it consists on selecting, for a training set characterized by d features, the subset of features which maximizes the accuracy of a certain classifier on further test instances. Thus, the cardinality of the search space is 2^d. Using an intuitive notation to represent each individual (there are d bits in each individual, each bit indicating whether a feature is present (1) or absent (0)), we can see in Fig. 3 an overview of the application of the EBNA search engine to the FSS problem. In each generation of the search, the induced BN will factorize the probability distribution of selected individuals. The BN will be formed by d nodes, each one representing one feature of the domain.

The determination of a minimum population size to reliably estimate the parameters of a BN is not an easy task [17]. This difficulty is higher for real world problems where the true real probability distribution is not known. In this way, taking the dimensionality of our problems into account, we consider that a population size of 1000 individuals is enough to reliably estimate the BN parameters.

A wrapper approach will be used to calculate the evaluation function value of each proposed individual or feature subset. The value of the evaluation function of a feature subset found by the EBNA search technique, once the supervised classifier is fixed, will be calculated by an accuracy estimation on training data.

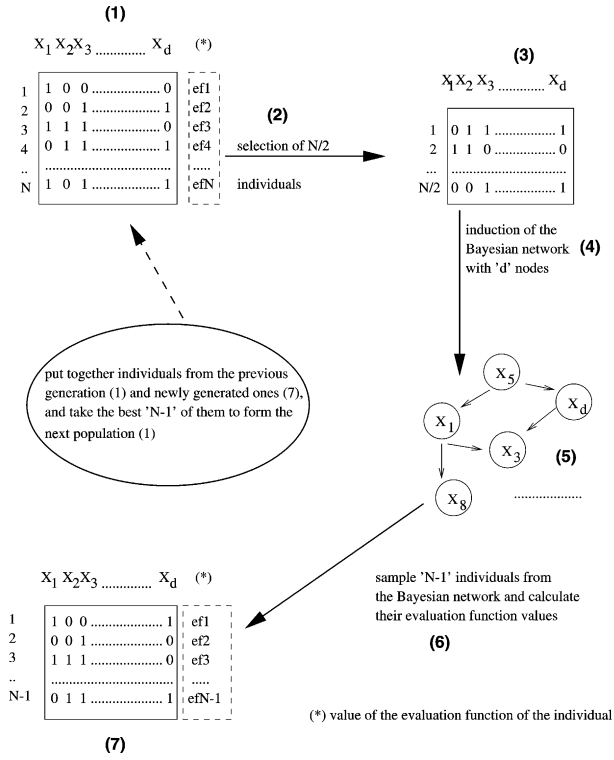

Fig. 3. FSS–EBNA method.

The accuracy estimation, seen as a random variable, has an intrinsic uncertainty. A 10-fold cross-validation multiple times, combined with a heuristic proposed by Kohavi and John [30], will be used to control the intrinsic uncertainty of the evaluation function. The heuristic works as follows:

- if the standard deviation of the accuracy estimate is above 1%, another 10-fold cross-validation is executed;
- this is repeated until the standard deviation drops below 1%, a maximum of five times.

In this way, small datasets will be cross-validated many times. However, larger ones possibly once. Although FSS–EBNA is independent from the specific supervised classifier used within its wrapper approach, in our set of experiments we will use the well-known Naive–Bayes (NB), [10] supervised classifier. It is a simple and fast classifier which uses the Bayes rule to predict the class for each test instance, assuming that features are independent to each other given the class. Due to its simplicity and fast induction, it is commonly used on Data Mining tasks of high dimensionality [30,41]. The probability for discrete features is estimated from data using maximum likelihood estimation and

applying the Laplace correction. A normal distribution is assumed to estimate the class conditional probabilities for continuous attributes. Unknown values in the test instance are skipped. Although its simplicity and its independence assumption among variables, the literature shows that the NB classifier gives remarkably high accuracies in many domains [33], specially in medical ones. Despite its good scaling with irrelevant features, NB can improve its accuracy level discarding correlated or redundant features. NB, based on the independence assumption of features to predict the class, is hurt by correlated features which violate this independence assumption. Thus, FSS can also play a 'normalization' role to discard these groups of correlated features, ideally selecting one of them in the final model. Although Langley and Sage [33] propose a sequential forward feature selection for detecting these correlations (starting from an empty subset of features and sequentially selecting features until no improvement is achieved), Kohavi and John [30] prefer a sequential backward elimination (starting from the full set of features and sequentially deleting features until no improvement is achieved). In our work we propose the use of FSS–EBNA to detect these redundancies among features to improve NB's predictive accuracy.

To stop the search algorithm, we have adopted an intuitive stopping criteria which takes the number of instances of the training set into account. In this way, we try to avoid the 'overfitting' risk [28,30]:

- for datasets with more than 2000 instances, the search is stopped when in a sampled new generation no feature subset appears with an evaluation function value improving the best subset found in the previous generation. Thus, the best subset of the search, found in the previous generation, is returned as the FSS–EBNA's solution;
- for smaller datasets the search is stopped when in a sampled new generation no feature subset appears with an evaluation function value improving, at least with a p-value smaller than 0.1, [5] the value of the evaluation function of the best feature subset of the previous generation. Thus, the best subset of the previous generation is returned as the FSS–EBNA's solution.

For larger datasets the 'overfitting' phenomenom has a lesser impact and we hypothesize that an improvement in the accuracy estimation over the training set will be coupled with an improvement in generalization accuracy on unseen instances. Otherwise, for smaller datasets, in order to avoid the 'overfitting' risk, the continuation of the search is only allowed when a significant improvement in the accuracy estimation of best individuals of consecutive generations appears. We hypothesize that when this significant improvement appears, the 'overfitting' risk decays and there is a basis for further generalization accuracy improvement over unseen instances.

[5] Using a 10-fold cross-validated paired t-test between the folds of both estimations, taking only the first run into account when 10-fold cross-validation is repeated multiple times.

Another concept to understand this stopping criteria is the wrapper nature of the proposed evaluation function. As we will see in the next section, the evaluation function value of each visited solution (the accuracy estimation of the NB classifier on training set by 10-fold cross-validation multiple times, using only the features proposed by the solution) needs several seconds to be calculated (never more than 4 s for used datasets). As the generation of a new generation of individuals implies the evaluation of 1000 new individuals, we only allow the continuation of the search when it demonstrates that it is able to escape from the local optimas, being able to discover new 'best' solutions in each generation. When the wrapper approach is used, the CPU time must be also controlled: we hypothesize that when the continuation of the search is allowed by our stopping criteria, the CPU times to evaluate a new generation of solutions are justified. For a larger study about this stopping criterion, the work [25] can be consulted.

5. Empirical comparison

We will test the power of FSS–EBNA in five real datasets. Table 1 reflects the principal characteristics of the datasets. All datasets come from the *UCI repository* [45] and they have been frequently used in the FSS literature. Due to their large dimensionality, an exhaustive search for the best feature subset is not computationally feasible, so a heuristic search must be performed.

In real datasets we do not know the optimal subset of features respect to the NB classifier. To test the power of FSS–EBNA, a comparison with the following well-known FSS algorithms is carried out:

- SFS is a classic hill-climbing search algorithm [29] which starts from an empty subset of features and sequentially selects features until no improvement is achieved in the evaluation function value. It performs the major part of its search near the empty feature set.
- SBE is another classic hill-climbing algorithm [29] which starts from the full set of features and sequentially deletes features until no improvement is

Table 1
Details of experimental domains

Domain	Number of instances	Number of classes	Number of features and their types
Ionosphere	351	2	34 (34-C)
Horse-colic	368	2	22 (15-D, 7-C)
Soybean-large	683	19	35 (35-C)
Anneal	898	6	38 (6-D, 32-C)
Image	2310	7	19 (19-C)

C: continuous, D: discrete.

achieved in the evaluation function value. It performs the major part of its search near the full feature set. Instead of working with a population of solutions, SFS and SBE try to optimize a single feature subset.
- GA with one-point crossover (GA-o).
- GA with uniform crossover (GA-u).
- FSS–EBNA.

For all the FSS algorithms, the wrapper evaluation function explained in the previous section is used. Although SFS and SBE algorithms stop deterministically, GA algorithms apply the same stopping criteria as FSS–EBNA.

Although the optimal selection of parameters is still an open problem on GAs [19], for both GA algorithms, guided by the recommendations of Bäck [4], the probability of crossover is set to 1.0 and the mutation probability to $1/d$, d being the number of variables of the domain (these values are so common in the literature). Fitness-proportionate selection [18] is used to select the individuals for crossover. In order to avoid any bias in the comparison, the remaining GA parameters are the same as FSS–EBNA's: the population size is set to 1000 and the new population formed by the best members from both old population and offspring.

Due to the non-deterministic nature of FSS–EBNA and GAs (two executions could not give the same result), 5 replications of 2-fold cross-validation (5×2cv) are applied to assess the predictive generalization accuracy of compared FSS algorithms. In each replication, the available data are randomly partitioned into two equal-sized sets S_1 and S_2. The FSS algorithm is trained on each set and tested on the other set. In this way, the reported accuracies are the mean of ten accuracies. The standard deviation of the mean is also reported. We lengthen the comparison by running the NB classifier without feature selection. Table 2 shows the accuracy results on tested datasets. Apart from a high accuracy level, we will also focus our attention on the achievement of a reduced number of features: a good tradeoff between a high accuracy and a low cardinality of the selected feature subset is desired.

A deeper analysis of the accuracy results is carried out by means of statistical tests. The 5×2cv F [3] test is performed to determine the significance degree of accuracy differences among each approach and FSS–EBNA. Thus, in Table 2 the symbol '†' denotes statistically significant difference over FSS–EBNA at the 0.05 confidence level; '$*$', significance at the 0.1 confidence level. Table 3 shows the average (and its standard deviation) number of features selected by different approaches. Experiments are executed in an SGI-Origin 200 computer using the implementation of the NB algorithm that appears in the *MLC++* [31] software.

Regarding these tables, the following observations can be made:
- All FSS algorithms help NB to reduce the number of features needed to induce the final models. This dimensionality reduction is coupled with considerable accuracy improvements in all datasets except *Anneal*, where

Table 2
Accuracy percentages of the NB classifier on real datasets without feature selection and using exposed five FSS methods

Domain	Without FSS	SFS	SBE	GA-o	GA-u	FSS–EBNA
Ionosphere	$84.84 \pm 3.12^{\dagger}$	$90.25 \pm 1.58^{*}$	91.39 ± 2.68	91.17 ± 3.19	$90.97 \pm 2.56^{*}$	92.40 ± 2.04
Horse-colic	$78.97 \pm 2.98^{\dagger}$	83.31 ± 1.98	$82.12 \pm 2.41^{*}$	83.43 ± 2.82	83.51 ± 1.47	83.93 ± 1.58
Soybean-large	$81.96 \pm 3.46^{\dagger}$	$86.38 \pm 3.30^{*}$	$87.78 \pm 3.90^{*}$	$85.64 \pm 4.06^{\dagger}$	$86.09 \pm 4.37^{\dagger}$	88.64 ± 1.70
Anneal	$93.01 \pm 3.13^{*}$	$86.72 \pm 2.09^{\dagger}$	$92.49 \pm 2.94^{*}$	$92.95 \pm 2.67^{*}$	93.13 ± 2.56	94.10 ± 3.00
Image	$79.95 \pm 1.52^{\dagger}$	88.65 ± 1.21	88.82 ± 1.74	88.67 ± 2.48	89.12 ± 1.56	88.98 ± 0.98
Average	83.74	87.06	88.52	88.37	88.56	89.53

The last row shows the average accuracy percentages for six domains.

Table 3
Cardinalities of finally selected features subsets for the NB classifier on real datasets without feature selection and using exposed five FSS methods

Domain	Without FSS	SFS	SBE	GA-o	GA-u	FSS–EBNA
Ionosphere	34	6.00 ± 1.41	21.30 ± 3.80	15.00 ± 2.36	12.66 ± 1.03	13.40 ± 2.11
Horse-colic	22	6.00 ± 2.74	11.20 ± 2.65	5.00 ± 2.82	4.60 ± 1.75	6.10 ± 1.85
Soybean-large	35	12.70 ± 2.71	23.50 ± 2.75	19.00 ± 2.09	19.16 ± 2.31	18.90 ± 2.76
Anneal	38	5.50 ± 2.32	33.60 ± 2.91	21.66 ± 2.66	19.50 ± 2.25	20.50 ± 3.13
Image	19	5.60 ± 1.57	9.40 ± 1.95	8.00 ± 1.41	8.00 ± 1.09	8.00 ± 0.66

It must be taken into account that when no FSS is applied to NB, it uses all the features.

FSS–EBNA is the only algorithm able to significantly improve the accuracy of the NB model without feature selection.

- FSS–EBNA has the best average accuracy among compared algorithms. It has the best estimated value in all except one dataset (*Image* domain, where differences are not significant respect to the best algorithm, GA-u).
- By average SBE, GA-o and GA-u achieve similar accuracy results (their average accuracy results are also very similar). Their accuracies show significant differences respect to FSS–EBNA in *Soybean-large* (SBE, GA-o and GA-u), *Anneal* (SBE and GA-o), *Ionosphere* (GA-u) and *Horse-colic* (SBE).
- Three randomized approaches (GA-o, GA-u, FSS–EBNA) carry out similar dimensionality reductions. We also note that in major part, they select the same features: thus, their accuracy differences in several datasets arise from their disjoint features.
- Although SFS returns the subsets with the fewest features, its accuracy values are in all except one dataset significantly inferior to FSS–EBNA. As SFS starts from the empty feature set, it is biased to perform the major part of its search near the empty feature set [15] and as we may expect it to return subsets with few features. However, it seems that it gets trapped on local optimas and it cannot escape to better fitted areas.
- Apart from the exposed accuracy differences between SBE and FSS–EBNA in several datasets, the major disadvantage of SBE is the slight reduction in the number of features it performs. In all domains, SBE is the algorithm with the lowest feature reduction, being nearly insignificant in *Anneal* dataset. As SBE starts from the full feature subset, it is biased to perform the major part of its search near the full feature set [15] and as we may expect it to return subsets with many features. However, it seems that it is not able to jump to better (or equally) fitted areas with fewer features.
- FSS–EBNA and GA approaches are not initially biased to specific areas of the search space and they are able to discover these well-fitted areas of non-specific dimensionality. Vafaie and De Jong [58] demonstrate on a set of artificial and real problems how greedy-like sequential searches as SFS and

SBE have a tendency to get trapped on local peaks of the search space, and they defend GAs as a more robust search engine in the FSS problem.

Although we have seen that two GA approaches and FSS–EBNA obtain similar accuracy results in many datasets, further studies show that GA approaches need more generations than FSS–EBNA to arrive to similar (or lower) accuracy levels. Table 4 reflects in which generation GA approaches and FSS–EBNA stop using the explained stopping criteria. The 5×2cv F-test is performed to determine the significance degree of the differences in the stop-generation among each GA approach and FSS–EBNA. Thus, in Table 4 the symbol '†' denotes a statistically significant difference over FSS–EBNA at the 0.05 confidence level; '$*$', significance at the 0.1 confidence level.

The results show that FSS–EBNA arrives faster to similar or better predictive accuracies (see also Table 2) than both GA approaches. GA-o and GA-u show significant differences in the stop-generation respect to FSS–EBNA in all except one dataset. FSS–EBNA, by the use of BNs, captures the underlying structure of the problem faster than GAs, avoiding the disruption, as GAs, of the relationships among dependent variables. Only in the *Image* dataset, the domain of lowest dimensionality, the use of FSS–EBNA does not return an advantage respect to other algorithms: observing the accuracy and stop-generation results in this domain, all FSS algorithms demonstrate a similar behaviour, finding similarly fitted subsets and population-based approaches needing nearly the same number of generations to stop.

This faster discovery of optimal solutions by EDA approaches than GAs was also noted by Harik [21] and Pelikan and Müehlenbein [50] in a set of classical optimization problems. As in our case, when a wrapper approach is used to calculate the evaluation function value of each found subset, the faster discovery of similar or better accuracies is a critical task. Although NB is a fast supervised classifier, it needs several seconds to estimate the predictive accuracy (by the exposed 10-fold cross-validation multiple times) of a feature subset on the training set: depending on the number of features selected by the subset, near 1 CPU second is needed in *Ionosphere* (the domain with fewest instances), while near 3 are needed in *Image* (the domain with most instances). As the

Table 4
Mean stop-generation for GAs and FSS–EBNA

Domain	GA-o	GA-u	FSS–EBNA
Ionosphere	$3.50 \pm 0.84^{\dagger}$	$3.10 \pm 0.56^{\dagger}$	1.80 ± 0.42
Horse-colic	$3.20 \pm 1.13^{*}$	$3.40 \pm 0.51^{\dagger}$	2.40 ± 0.69
Soybean-large	$3.30 \pm 0.82^{*}$	$3.60 \pm 0.51^{\dagger}$	2.50 ± 0.70
Anneal	$3.80 \pm 0.42^{\dagger}$	$3.20 \pm 0.44^{\dagger}$	1.80 ± 0.42
Image	3.60 ± 0.84	3.70 ± 0.48	3.50 ± 0.42

The standard deviation of the mean is also reported. The initial generation is considered as the zero generation.

generation of a new population of solutions implies the evaluation of new 1000 individuals, an earlier stop without a damage to the accuracy level is highly desired in order to save CPU time. On the other hand, the times for the induction of the BNs over the best individuals are insignificant in all domains: 3 CPU seconds are needed by average in *Image* domain (the domain with fewer features) and 14 CPU seconds in *Anneal* (the domain with more features).

In a subsequent analysis between GA-o and GA-u predictive accuracies, we note that the differences are not statistically significant in any dataset. So, we should focus our attention on their stop-generation numbers to extract differences between the behaviour of GA-o and GA-u. Thus, observing Table 4, the one-point crossover is better suited for *Horse-colic* and *Soybean-large* datasets than the uniform crossover; otherwise, we see the opposite behaviour in *Ionosphere* and *Anneal*. By the use of FSS–EBNA, we also avoid this tunning among different crossover operators for each dataset.

6. Summary and future work

In this work, we have performed a comparison among FSS–EBNA, a randomized and population-based evolutionary algorithm, and two genetic and other two sequential classic search algorithms in the FSS task. In FSS–EBNA, the FSS problem, stated as a search problem, uses the EBNA search engine, a variant of the EDA approach. The EDA paradigm is born from the roots of the GA community in order to explicitly discover the relationships among the features of the problem and not disrupt them by genetic recombination operators. The EDA paradigm avoids the use of recombination operators and it guarantees the evolution of solutions and the discovery of these relationships by the factorization of the probability distribution of best individuals in each generation of the search. In EBNA, this factorization is carried out by a BN induced by a cheap local search mechanism. As another attractive effect, with the use of FSS–EBNA we avoid the tuning of genetic parameters, still an open problem on GA literature.

Using a wrapper approach, we have searched for the subset of features that optimizes the predictive accuracy of the NB supervised classifier. We have carried out a comparison in a set of tasks among two well-known sequential search engines algorithms (SFS and SBE) and GAs with one-point and uniform crossover operators. FSS–EBNA has demonstrated a robust behaviour, obtaining the best average accuracy respect to all the tasks, the best in all except one dataset and a large reduction in the number of selected features respect the full feature set. We have noted the tendency of SFS to get trapped on local accuracy minima and the incapacity of SBE to perform an acceptable feature reduction. We have also noted the capacity of FSS–EBNA to arrive to similar or better fitted solutions faster than both GA approaches. BNs, which

factorize the probability distribution of best solutions in each generation, are able to capture the underlying structure of the problems faster than GA approaches. On the other hand, the GA's crossover operator frequently disrupts the dependencies among related features: this is the cause of the slower discovery by GAs of solutions of the same level as FSS–EBNA.

As future work, we consider lengthening the works already done using the BNs for the Feature Weighting problem in Nearest Neighbor Algorithm [23] and for the representation of the behaviour of Data Mining classifiers [24]. Continuing the work within the EDA approach for FSS, in order to deal with domains with much larger numbers of features (>50), future work should address the use of simpler probability models to factorize the probability distribution of best individuals, models which assume fewer or no dependencies between the variables of the problem (see [26] for a medical application). Other interesting possibility is the use of parallel algorithms to induce BNs in these tasks of high dimensionality [53,60]. Another way of researching will be the employment of a metric which fixes for each domain the number of individuals needed to reliably learn [17] the parameters of the BN.

Acknowledgements

This work is partially supported by the University of the Basque Country and by the Department of Education, University and Research of the Basque Government under grants 9/UPV/EHU/ 00140.226-12084/2000 and PI 1999-40 respectively.

References

[1] S. Acid, L.M. de Campos, Approximations of causal networks by polytrees: an empirical study, in: B. Bouchon-Meunier, R.R. Yager, L.A. Zadeh (Eds.), Advances in Intelligent Computing, vol. 945, Lecture Notes in Computer Science, Springer, Berlin, 1995, pp. 149–158.
[2] D.W. Aha, R.L. Bankert, Feature selection for case-based classification of cloud types: An empirical comparison, in: Proceedings of the AAAI'94 Workshop on Case-Based Reasoning, 1994, pp. 106–112.
[3] Alpaydin, E., Combined 5×2cv F test for comparing supervised classification learning algorithms, Neural Comput. 11, 1885–1982.
[4] T. Bäck, Evolutionary Algorithms is Theory and Practice, Oxford University Press, Oxford, 1996.
[5] S. Baluja, Population-based incremental learning: A method for integrating genetic search based function optimization and competitive learning, Technical Report CMU-CS-94-163, Carnegie Mellon University, Pittsburgh, PA, 1994.
[6] S. Baluja, S. Davies, Using optimal dependency-trees for combinatorial optimization: Learning the structure of the search space, in: Proceedings of the Fourteenth International Conference on Machine Learning, 1997, pp. 30–38.

[7] P.P. Bonissone, Y.-T. Chen, K. Goebel, P.S. Khedkar, Hybrid soft computing systems: Industrial and commercial applications, Proc. IEEE 87 (9) (1999) 1641–1667.
[8] W. Buntine, Theory refinement in Bayesian networks, in: Proceedings of the Seventh Conference on Uncertainty in Artificial Intelligence, 1991, pp. 52–60.
[9] E. Castillo, J.M. Gutiérrez, A.S. Hadi, Expert Systems and Probabilistic Network Models, Springer, Berlin, Germany, 1997.
[10] B. Cestnik, Estimating probabilities: a crucial task in machine learning, in: Proceedings of the European Conference on Artificial Intelligence, 1990, 147–149.
[11] M. Chickering, D. Geiger, D. Heckerman, Learning Bayesian networks is NP-hard, Technical Report MSR-TR-94-17, Microsoft Research, Redmond, WA, 1994.
[12] D.M. Chickering, D. Geiger, D. Heckerman, Learning Bayesian networks: Search methods and experimental results, Preliminary Papers of the 5th International Workshop on Artificial Intelligence and Statistics, 1995, pp. 112–128.
[13] A.P. Dawid, Conditional independence in statistical theory, J. R. Stat. Soc. B 41 (1979) 1–31.
[14] J.S. De Bonet, C.L. Isbell, P. Viola, MIMIC: Finding optima by estimating probability densities, in: Advances in Neural Information Processing Systems, vol. 9, MIT Press, Cambridge, MA, 1997.
[15] J. Doak, An evaluation of feature selection methods and their application to computer security, Technical Report CSE-92-18, University of California, Davis, CA, 1992.
[16] R. Etxeberria, P. Larrañaga, Global optimization with Bayesian networks, in: Proceedings of the Second Symposium on Artificial Intelligence, La Habana, Cuba, 1999, pp. 332–339.
[17] N. Friedman, Z. Yakhini, On the sample complexity of learning Bayesian networks, in: Proceedings of the Twelfth Conference on Uncertainty in Artificial Intelligence, Portland, OR, 1996, pp. 274–282.
[18] D.E. Goldberg, Genetic algorithms in search, optimization, and machine learning, Addison-Wesley, Reading, MA, 1989.
[19] J.J. Grefenstatte, Optimization of control parameters for genetic algorithms, IEEE Trans. Syst., Man Cybern. 1 (1986) 122–128.
[20] G.R. Harik, F.G. Lobo, D.E. Goldberg, The compact genetic algorithm, IlliGAL Report 97006, Illinois Genetic Algorithms Laboratory, University of Illinois, Urbana–Champaign, 1997.
[21] G. Harik, Linkage learning via probabilistic modelling in the ECGA, IlliGAL Report 99010, Illinois Genetic Algorithms Laboratory, University of Illinois, Urbana–Champaign, 1999.
[22] Henrion, M., Propagating uncertainty in Bayesian networks by probabilistic logic sampling, in: Uncertainty in Artificial Intelligence 2, Elsevier, Amsterdam, 1988, pp. 149–163,.
[23] I. Inza, Feature weighting for nearest neighbor algorithm by Bayesian networks based combinatorial optimization, in: Proceedings of the Student Session of Advanced Course on Artificial Intelligence, 1999, pp. 33–35.
[24] I. Inza, P. Larrañaga, B. Sierra, R. Etxeberria, J.A. Lozano, J.M. Peña, Representing the behaviour of supervised classification learning algorithms by Bayesian networks, Pattern Recogn. Lett. 20 (11–13) (1999) 1201–1210.
[25] I. Inza, P. Larrañaga, R. Etxeberria, B. Sierra, Feature subset selection by Bayesian networks based optimization, Artif. Intell. 123 (1–2) (2000) 157–184.
[26] I. Inza, M. Merino, P. Larrañaga, J. Quiroga, B. Sierra, M. Girala, Feature subset selection by genetic algorithms and estimation of distribution algorithms. A case study in the survival of cirrhotic patients treated with TIPS, Artif. Intell. Med., in press.
[27] A.K. Jain, R. Chandrasekaran, Dimensionality and sample size considerations in pattern recognition practice, in: Handbook of statistics – II, North-Holland, Amsterdam, The Netherlands, 1982, pp. 835–855.
[28] A. Jain, D. Zongker, Feature selection: Evaluation, application, and small sample performance, IEEE Trans. Pattern Anal. 19 (2) (1997) 153–158.

[29] J. Kittler, Feature Set Search Algorithms, in: Pattern Recognition and Signal Processing, Sithoff and Noordhoff, Alphen aan den Rijn, The Netherlands, 1978, pp. 41–60.
[30] R. Kohavi, G. John, Wrappers for feature subset selection, Artif. Intell. 97 (1–2) (1997) 273–324.
[31] R. Kohavi, D. Sommerfield, J. Dougherty, Data mining using MLC++, a Machine Learning Library in C++, International Journal of Artificial Intelligence Tools 6 (1997) 537–566.
[32] M. Kudo, J. Sklansky, Comparison of algorithms that select features for pattern classifiers, Pattern Recogn. 33 (2000) 25–41.
[33] P. Langley, S. Sage, Induction of selective Bayesian classifiers, in: Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence, 1994, pp. 399 –406.
[34] P. Larrañaga, C.M.H. Kuijpers, R.H. Murga, Y. Yurramendi, Learning Bayesian network structures by searching for the best ordering with genetic algorithms, IEEE Trans. Syst. Man Cybern. A 26 (4) (1996) 487–493.
[35] P. Larrañaga, M. Poza, Y. Yurramendi, R.H. Murga, C.M.H. Kuijpers, Structure Learning of Bayesian networks by genetic algorithms: A performance analysis of control parameters, IEEE Trans. Pattern Anal. 18 (9) (1996) 912–926.
[36] P. Larrañaga, R. Etxeberria, J.A. Lozano, J.M. Peña, Combinatorial optimization by learning and simulation of Bayesian networks, in Proceedings of the Conference in Uncertainty in Artificial Intelligence, 2000, pp. 343–352.
[37] P. Larrañaga, J.A. Lozano, Estimation of Distribution Algorithms. A New Tool for Evolutionary Computation, Kluwer Academic Publishers, Norwell, MA, 2001.
[38] S.L. Lauritzen, Graphical Models, Oxford University Press, Oxford, England, 1996.
[39] H. Liu, H. Motoda, Feature Selection for Knowledge Discovery and Data Mining, Kluwer Academic Publishers, Norwell, MA, 1998.
[40] A.J. Miller, Subset Selection in Regression, Chapman & Hall, Washington, DC, 1990.
[41] D. Mladenić, Feature subset selection in text-learning, in: Proceedings of the Tenth European Conference on Machine Learning, 1988, pp. 95–100.
[42] H. Müehlenbein, The equation for response to selection and its use for prediction, Evol. Comput. 5 (3) (1997) 303–346.
[43] H. Müehlenbein, G. Paaß, From recombination of genes to the estimation of distributions, in: Binary Parameters, Lecture Notes in Computer Science 1411: Parallel Problem Solving from Nature – PPSN IV, 1996, pp. 178–187.
[44] H. Müehlenbein, T. Mahnig, FDA – A scalable evolutionary algorithm for the optimization of additively decomposed functions, Evol. Comput. 7 (4) (1999) 353–376.
[45] P. Murphy, UCI Repository of Machine Learning Databases, Department of Information and Computer Science, University of California, Irvine, CA, 1995.
[46] P. Narendra, K. Fukunaga, A branch and bound algorithm for feature subset selection, IEEE Trans. Comput. C-26 (9) (1977) 917–922.
[47] A. Ochoa, M. Soto, R. Santana, J. Madera, N. Jorge, The factorized distribution algorithm and the junction tree: A learning perspective, in: Proceedings of the Second Symposium on Artificial Intelligence, 1999, pp. 368–377.
[48] J. Pearl, Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Morgan Kaufmann, Morgan Kaufmann, Palo Alto, CA, 1988.
[49] M. Pelikan, D.E. Goldberg, E. Cantú-Paz, BOA: The Bayesian Optimization Algorithm, IlliGAL Report 99003, Illinois Genetic Algorithms Laboratory, University of Illinois, Urbana–Champaign, 1999.
[50] M. Pelikan, H. Müehlenbein, The bivariate marginal distribution algorithm, in: Advances in Soft Computing-Engineering Design and Manufacturing, Springer, London, 1999, pp. 521–535.
[51] G.M. Provan, M. Singh, Learning Bayesian networks using feature selection, Preliminary Papers of the Fifth International Workshop on Artificial Intelligence and Statistics, 1995, pp. 450–456.

[52] P. Pudil, J. Novovicova, J. Kittler, Floating search methods in feature selection, Pattern Recogn. Lett. 15 (1) (1994) 1119–1125.
[53] R. Sangüesa, U. Cortés, A. Gisolfi, A parallel algorithm for building possibilistic causal networks, Int. J. Approx. Reason. 18 (3–4) (1998) 251–270.
[54] W. Siedelecky, J. Sklansky, On automatic feature selection, Int. J. Pattern Recogn. 2 (1998) 197–220.
[55] G. Schwarz, Estimating the dimension of a model, Ann. Stat. 7 (1978) 461–464.
[56] M. Soto, A. Ochoa, S. Acid, L.M. de Campos, Introducing polytree approximation of distribution algorithm, in: Proceedings of the Second Symposium on Artificial Intelligence, 1999, pp. 360–367.
[57] G. Syswerda, Uniform crossover in genetic algorithms, in: Proceedings of the International Conference on Genetic Algorithms, vol. 3, 1989, pp. 2–9.
[58] H. Vafaie, K. De Jong, Robust feature selection algorithms, in: Proceedings of the Fifth International Conference on Tools with Artificial Intelligence, 1993, 356–363.
[59] M.L. Wong, W. Lam, K.S. Leung, Using Evolutionary programming and minimum description length principle for data mining of Bayesian networks, IEEE Trans. Pattern Anal. 21 (2) (1999) 174–178.
[60] Y. Xiang, T. Chu, Parallel learning of belief networks in large and difficult domains, Data Min. Knowl. Disc. 3 (3) (1999) 315–338.