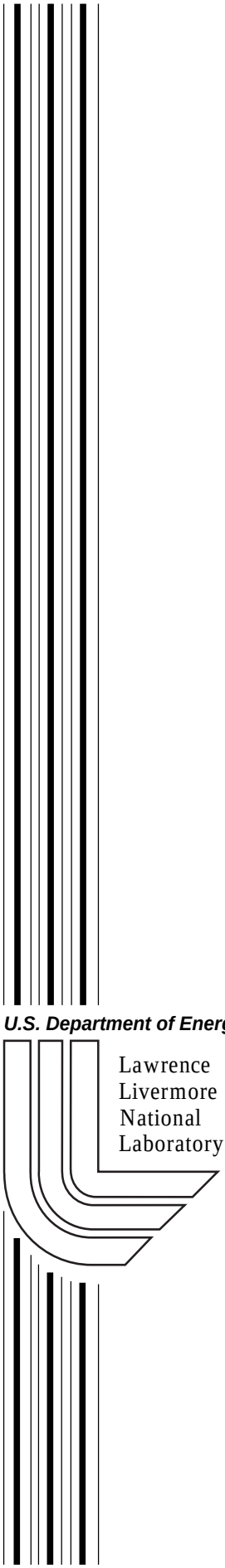
Preprint
UCRL-JC-146851

Feature Subset Selection by Estimation of Distribution Algorithms

E. Cantú-Paz

This article was submitted to Genetic and Evolutionary Computation Conference, New York City, NY, July 9 – 13, 2002

January 17, 2002



This work was performed under the auspices of the United States Department of Energy by the University of California, Lawrence Livermore National Laboratory under contract No. W-7405-Eng-48.

Feature Subset Selection by Estimation of Distribution Algorithms

Erick Cantú-Paz
Center for Applied Scientific Computing
Lawrence Livermore National Laboratory
Livermore, CA 94551
cantupaz@llnl.gov

Abstract

This paper describes the application of four evolutionary algorithms to the identification of feature subsets for classification problems. Besides a simple GA, the paper considers three estimation of distribution algorithms (EDAs): a compact GA, an extended compact GA, and the Bayesian Optimization Algorithm. The objective is to determine if the EDAs present advantages over the simple GA in terms of accuracy or speed in this problem. The experiments used a Naive Bayes classifier and public-domain and artificial data sets. In contrast with previous studies, we did not find evidence to support or reject the use of EDAs for this problem.

1 INTRODUCTION

In machine learning, the problem of supervised classification is concerned with using labeled examples to induce a model that classifies objects into a finite set of known classes. The examples are described by a vector of numeric or nominal features. Some of these features may be irrelevant or redundant. Avoiding irrelevant or redundant features is important because they may have a negative effect on the accuracy of the classifier. In addition, by using fewer features we may obtain savings in the cost of acquiring the data, and improve the comprehensibility of the classification model. Finding feature subsets that result in accurate classifiers can be cast as a search problem, and simple genetic algorithms have been used successfully for this problem in the past.

This paper presents experiments with a simple genetic algorithm (sGA), and three estimation of distribution algorithms (EDAs): a compact GA (cGA), an extended compact GA (ecGA), and the Bayesian Optimization Algorithm (BOA). Instead of the mutation and crossover operations of conventional GAs, EDAs use a statistical model of the individuals that survive selection to generate new individuals. EDAs are an important step toward solving the linkage problem, a fundamental obstacle to the application of simple GAs to problems with unknown relationships among variables. Numerous experimental and theoretical results show that EDAs can solve hard problems reliably and efficiently (Pelikan et al., 1999; Etxeberria & Larrañaga, 1999; Mühlenbein & Mahnig, 1999).

The objective of this study is to determine if EDAs present advantages over simple GAs in terms of accuracy or speed when applied to feature selection problems. The experiments described in this paper use public-domain and artificial data sets. The classifier was a Naive Bayes, a simple classifier that can be induced quickly, and that has been shown to have good accuracy in many problems (Kohavi & John, 1997).

Our target was to maximize the accuracy of classification. The experiments demonstrate that all the feature selection methods tried here resulted in higher accuracies than using all the features. However, in contrast with other studies, we found no evidence to support or reject the use of the advanced EDAs in this problem.

The next section briefly reviews previous applications of EAs to feature subset selection. Section 3 describes the algorithms, data sets, and the fitness evaluation method. The experimental results are presented in section 4. Section 5 concludes this paper with a summary and a discussion of future research directions.

2 FEATURE SELECTION

In a domain where objects are described by d features, there are 2^d possible feature subsets. Obviously, searching exhaustively for the best subset (using any criteria to measure quality) is futile. One approach to deal with this problem is to preprocess the data and select features based on properties that good feature sets are presumed to have, such as

orthogonality and high information content. This is known as the filter approach (John, Kohavi, & Phleger, 1994). Although it can be relatively fast, the filter approach may produce disappointing results, because it ignores completely the induction algorithm.

An alternative to preprocessing the data is the wrapper approach. The key idea is to consider the induction algorithm as a black box that can be used by a heuristic search algorithm to evaluate each candidate feature subset (John, Kohavi, & Phleger, 1994). The feature subset with the higher evaluation is selected as the final set on which to run the inducer, which then should be tested on data not used during the search.

Numerous search algorithms have been used to search for feature subsets (Jain & Zongker, 1997). Genetic algorithms are usually reported to deliver good results, but there are exceptions where simpler (and faster) algorithms result in higher accuracies on particular data sets (Jain & Zongker, 1997)

Applying GAs to the feature selection problem is straightforward: the chromosomes of the individuals contain one bit for each feature, and the value of the bit determines whether the feature will be used in the classification. Using the wrapper approach, the individuals are evaluated by training the classifiers using the feature subset indicated by the chromosome and using the resulting accuracy to calculate the fitness. Siedlecki and Sklansky (1989) were the first to describe the application of GAs in this way.

GAs have been used to search for feature subsets in conjunction with several classification methods such as neural networks (Brill et al., 1990; Brotherton & Simpson, 1995), decision trees (Bala et al., 1996), k-nearest neighbors (Kelly & Davis, 1991; Punch et al., 1993; Raymer et al., 1997), rules (Vafaie & Jong, 1993), and Naive Bayes (Inza et al., 1999).

Besides selecting feature subsets, GAs can extract new features by searching for a vector of numeric coefficients that is used to transform linearly the original features (Kelly & Davis, 1991; Punch et al., 1993). In this case, a value of zero in the transformation vector is equivalent to avoiding the feature. Raymer et al. (1997) and Raymer et al. (2000) combined the linear transformation with explicit feature selection flags in the chromosomes, and reported an advantage over the pure transformation method.

As far as we could tell, only one model-building EA has been used previously to select feature subsets. Inza et al. (1999) and Inza et al. (2001) presented experiments with an algorithm that learns a Bayesian network to model promising solutions. Inza et al. (2001) reported that the model-building algorithm found subsets that result in better accuracies than simple GAs and two sequential feature selection algorithms. Their algorithm is similar to one included in our study, and we use some of the same data sets.

3 METHODS

This section describes the algorithms and the data used in this study as well as the method used to evaluate the fitness.

3.1 ALGORITHMS AND DATA SETS

The simple genetic algorithm in this study uses binary strings, binary (pairwise) tournament selection without replacement, one-point crossover, and bit-wise point mutation. Simple GAs such as this have been used successfully in many applications. However, it has long been recognized that the problem-independent crossover operators used in simple GAs can disrupt groups of related variables and prevent the algorithm from reaching the global optimum, unless exponentially-sized populations are used (Thierens (1999) gives a good description of this problem).

One approach to identify and exploit the relationships among variables is to estimate the joint distribution of the individuals that survive selection and use this distribution to generate new individuals. The complexity of the models has increased over time as the methods of building models from data mature and more powerful computers become available. Interested readers can consult the reviews by Pelikan et al. (1999) and Larrañaga et al. (1999).

The simplest model-building EA that was used in the experiments reported here is the compact GA (Harik, Lobo, & Goldberg, 1998). This algorithm assumes that the variables (bits) that represent the problem are independent, and therefore it models the population with a product of Bernoulli distributions. The compact GA receives its name from the compact way it represents the population: the cGA uses a vector p of length equal to the problem's length, l. Each element of p contains the probability that a sample will take the value 1. If the Bernoulli trial is not successful the sample will be 0. All positions of p are initialized to 0.5 to simulate the usual uniform random initialization of simple GAs. New individuals are obtained by sampling consecutively from each position of p and concatenating the values obtained. The probabilities vector is updated by comparing the fitness of two individuals obtained from it. For each $p_k, k = 1, .., l$, if the fittest individual has a 1 in the k-th position, p_k is increased by $1/n$, where n is the size of the virtual population that the user wants to simulate. Likewise, if the fittest individual has a 0 in the k-th position, p_k is decreased by $1/n$. The cGA iterates until all positions in p_k contain either zero or one.

The PBIL (Baluja, 1994) and the UMDA (Mühlenbein, 1998) are other examples of algorithms that use univariate

models and operate on binary alphabets. They differ from the cGA in the method to update the probabilities vector.

The extended compact GA (Harik, 1999) uses a product of marginal distributions on a partition of the variables. In this model, subsets of variables can be modeled jointly, and the subsets are considered independent of other subsets. Formally, the model is $P = \prod_{i=0}^{m} P_i$, where m is the number of subsets in the partition of variables and P_i represents the distribution of the i-th subset. The distribution of a subset with k members is stored in a table with $2^k - 1$ entries. The problem consists on finding a partition that models the population correctly. Harik (1999) proposed a greedy algorithm that initially supposes that all variables are independent. The model search tries to merge all pairs of subsets and chooses the merger that minimizes a complexity measure based on information theory. The search continues until no further subsets can be merged. In contrast to the cGA, the ecGA has an explicit population that is evaluated and subject to selection at each iteration of the algorithm. The algorithm builds the model considering only those solutions that survive selection. The population is initialized randomly, and new individuals are generated by sampling consecutively from the m subset distributions.

The Bayesian Optimization Algorithm (Pelikan, Goldberg, & Cantú-Paz, 1999) models the selected individuals using a Bayesian network, which can represent dependence relations among arbitrary number of variables. Independently, Etxeberria and Larrañaga (1999) and Mühlenbein and Mahnig (1999) introduced similar algorithms. The BOA uses a greedy search to optimize the Bayesian Dirichlet metric, a measure of how well the network represents the data (the BOA could use other metrics). The user specifies the maximum number of incoming edges to any node of the network. This number corresponds to the highest degree of interaction assumed among the variables of the problem. As the ECGA, the BOA builds the model considering only the solutions that survived selection. New individuals are generated by sampling from the network. The main difference between the ecGA and the BOA is the model that they use to represent the survivors.

Figure 1 illustrates the different models used by the ecGA and the BOA. The ecGA cannot represent individual relationships among the variables in a subset.

The classifier induced in the experiments was a Naive Bayes (NB). This classifier was chosen for its speed and simplicity, but the evolutionary wrapper method is suitable for any other supervised classifiers, as mentioned in the previous section. In the NB, the probabilities for nominal features were estimated from the data using maximum likelihood estimation (their observed frequencies in the data) and applying the Laplace correction. Numeric features were assumed to have a normal distribution. Missing values in the data were skipped.

The experiments used the C++ implementations of the ecGA (Lobo & Harik, 1999) and the BOA version 1.0 (Pelikan, 1999) that are distributed by their authors on the web.[1] The ecGA code has a non-learning mode that emulates the cGA. The sGA and Naive Bayes were developed in C++. All programs were compiled with g++ version 2.96 using -O2 optimizations and executed on a Linux workstation with dual 1.5 GHz Intel Xeon processors (all programs were executed on a single processor). For the ecGA and the BOA codes, we used the random number generators included in their distributions, for everything else we used a Mersenne Twister.

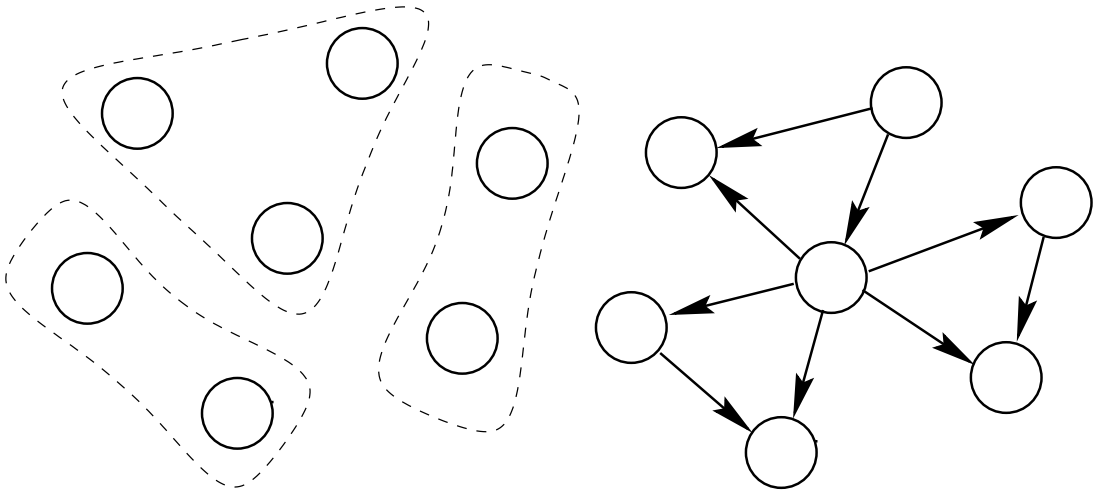
The first four data sets used in the experiments are available in the UCI repository (Blake & Merz, 1998). The data sets are briefly described in table 1. Random21 and Redundant21 are two artificial data sets with 21 features each. The target concept of these two data sets is to define whether the first nine features are closer to (0,0,...,0) or (9,9,...,9) in Euclidean distance. The features were generated uniformly at random in the range [3,6]. All the features in Random21 are random, and the first, fifth, and ninth features are repeated four times each in Redundant21. We took the definition of Redundant21 from the paper by Inza et al. (1999).

3.2 MEASURING FITNESS

Since we are interested in classifiers that generalize well, the fitness calculations must include some estimate of the generalization of the Naive Bayes using the candidate subsets. If enough data are available, the generalization may be estimated by dividing the training data into training and testing sets. The training set is used to find the class conditional probabilities, and the accuracy of the trained classifier on the testing set is used to calculate the fitness.

Unfortunately, the training data sets are small, so the procedure above may not be practical in our case. Instead, we estimate the generalization of the network using crossvalidation. In k-fold crossvalidation, the data D is partitioned randomly into k non-overlapping sets, $D_1, ...D_k$. At each iteration i (from 1 to k), the network is trained with $D \backslash D_i$ and tested on D_i. Since the data are partitioned randomly, it is likely that repeated crossvalidation experiments return different results. Although there are well-known methods to deal with "noisy" fitness evaluations in EAs (Miller & Goldberg, 1996), we chose to limit the uncertainty in the accuracy estimate by repeating 10-fold crossvalidation experiments until the standard deviation of the accuracy estimate drops below 1% (or a maximum of five repetitions). This heuristic was proposed by Kohavi and John (1997) in

[1] Available at `http://www-illigal.ge.uiuc.edu`



(a) ecGA (b) BOA

Figure 1: Representation of the models used in the ecGA and the BOA.

Domain	Instances	Classes	Numeric Feat.	Nominal Feat.	Missing
Ionosphere	351	2	34	–	N
Segmentation	2310	7	19	–	N
Sick Euthyroid	3163	2	7	18	Y
Soybean Large	683	19	–	35	Y
Random21	2500	2	21	–	N
Redundant21	2500	2	21	–	N

Table 1: Description of the data used in the experiments. The last column indicates if the data has missing values.

their study of wrapper methods for feature selection, and was adopted by Inza et al. (1999). We use the accuracy estimate as our fitness function.

Even though crossvalidation is expensive computationally, the cost was not prohibitive in our case, since the data sets were relatively small and the NB classifier is very efficient. If larger data sets or other inducers were used, we would have to deal with the uncertainty in the evaluation by other means, such as increasing slightly the population size (to compensate for the noise in the evaluation) or by sampling the training data. We defer a discussion of possible performance improvements until the final section.

Our fitness measure does not include any term to bias the search toward small feature subsets. However, the algorithms found small subsets, and with some data the algorithms consistently found the smallest subsets that describe the target concepts. This suggests that the data sets contained irrelevant or redundant features that decreased the accuracy of the Naive Bayes.

4 EXPERIMENTS

The simple GA used a population with 100 individuals, one-point crossover with probability 1.0, and mutation with probability $1/l$, where l was the length of the chromosomes that corresponds to the total number of features in each problem. Promising solutions were selected with pairwise binary tournaments without replacement. The experiments were terminated after 50 generations, although we did not observe much improvements after 10–20 generations.

The cGA, ecGA, and the BOA used a population with 1000 individuals. Larger populations were chosen because these algorithms may need large samples to estimate correctly the parameters of their models of promising solutions. These algorithms were terminated after a maximum of 50 generations. The remainder of the parameters used were the defaults provided in their distributions: the cGA and ecGA used tournaments among 16 individuals, and the BOA used truncation selection with a threshold of 50%.

To evaluate the generalization accuracy of the feature selection methods, we used 5 iterations of 2-fold crossvalidation (5x2cv). In each iteration, the data were randomly divided in halves. One half was input to the feature selection algorithms. The final feature subset found in each experiment was used to train a final NB classifier (using the training data), which was then tested on the other half of the data. The accuracy results presented in table 2 are the average and standard deviations of the ten tests.

To determine if the differences among the algorithms are statistically significant, we used a combined F test proposed by Alpaydin (1999). Let $p_i^{(j)}$ denote the difference in the accuracy rates of two classifiers in fold j of the i-th iteration of 5x2 cv, $\bar{p} = (p_i^{(1)} + p_i^{(2)})/2$ denote the mean, and $s_i^2 = (p_i^{(1)} - \bar{p})^2 + (p_i^{(2)} - \bar{p})^2$ the variance, then

$$f = \frac{\sum_{i=1}^{5} \sum_{j=1}^{2} \left(p_i^{(j)}\right)^2}{2 \sum_{i=1}^{5} s_i^2}$$

is approximately F distributed with 10 and 5 degrees of freedom, and we reject the null hypothesis that the two algorithms have the same error rate with 0.95 confidence if $f > 4.74$ (Alpaydin, 1999). Care must be taken to ensure that all the algorithms use the same training and testing data in the two folds of the five crossvalidation experiments.

Table 2 has the average accuracies obtained. The best observed result in the table is highlighted in **bold** type, and those results that according to the combined F test are significantly different from the best are marked with a bullet (•). There are two immediate observations that we can make from the results. First, the feature selection algorithms result in a great improvement in accuracy over using a NB with all the features. However, this difference is not always significant (Soybean Large, Random21). Second, all the feature selection algorithms result in similar accuracy values. There is not a single statistically significant difference in these data sets.

We must be careful not to take the results at face value and conclude incorrectly that the cGA and the ecGA find feature subsets that result in better accuracies than the other EAs, since the differences are small and not significant. For the same reasons, we cannot disqualify the BOA, which did not score highest in any data set, or any other algorithm.

Our results disagree with the conclusions of Inza et al. (1999) and Inza et al. (2001), who found statistically significant differences between the accuracy of their EDA and other genetic and sequential feature selection methods (using the same combined F test). This disagreement may be due to differences in the algorithms or the experimental setup. Their EDA learns a Bayesian network from the selected individuals using a greedy search that adds edges to the graph that maximize the Bayesian Information Criterion; the BOA considers edge additions and deletions and attempts to maximize a different measure of model quality. Another important difference is that they stopped their algorithms after not observing a (significant) improvement over the previous generation, while we stopped after 50 generations. Iterating the algorithm longer could result in overfitting the training data, but preliminary experiments using their stopping criterion do not show any significant differences.

In terms of the size of the final feature subsets, all the algorithms find similarly-sized subsets, which are substantially and significantly smaller than the original set of features

(see table 3). It is interesting to note that all the EAs were able to find subsets with nine relevant features for the Redundant21 data (the sGA found a solution with 10 features once).

The EDAs used here took considerably more time to finish than the simple GA, which was expected since the simple GA used a smaller population size.[2] This observation, along with the experimental results of accuracy and feature subset size, leads us to recommend the simple GA over the EDAs for feature selection problems.

5 CONCLUSIONS

This paper presented experiments with four evolutionary algorithms applied to the feature selection problem. The experiments considered a Naive Bayes classifier and public-domain and artificial data sets. With this data and classifier we did not find evidence to support or reject the use of the sophisticated model-building EAs in this problem. However, if we take into account that the simple GA was much faster than the other algorithms and found feature subsets of similar quality, we are inclined to recommend the sGA over the other algorithms.

There are numerous opportunities to extend this work. The results that suggest that model-building GAs are not advantageous for feature selection should be explored further with additional data sets and other induction algorithms. It is not clear what characteristics of the data or the classifier would require an EDA to find feature subsets that reliably result in high accuracies.

Future work should also explore methods to improve the computational efficiency of the algorithms to deal with much larger data sets. In particular, subsampling the training sets and parallelizing the fitness evaluations seem like promising alternatives. In addition, future work should explore efficient methods to deal with the noisy accuracy estimates, instead of using the expensive multiple crossvalidations that we employed. Previous work (Miller & Goldberg, 1996) indicates that small increases of the population size are sufficient to deal with noise in the fitness evaluation.

[2] We would expect the simple GA to be ten times faster than the other algorithms, since its population was ten times smaller. However, the simple GA was usually more than ten times faster than the rest. Some of this extra time can be explained because the EDAs build a model every generation. There are also random variations in the number of crossvalidations used to estimate the accuracy, which may account for some deviation from our expectations.

Acknowledgments

I thank Martin Pelikan for providing the graphs in figure 1.

UCRL-JC-146851. This work was performed under the auspices of the U.S. Department of Energy by University of California Lawrence Livermore National Laboratory under contract No. W-7405-Eng-48.

References

Alpaydin, E. (1999). Combined 5×2cv F test for comparing supervised classification algorithms. *Neural Computation, 11*, 1885–1892.

Bala, J., De Jong, K., Huang, J., Vafaie, H., & Wechsler, H. (1996). Using learning to facilitate the evolution of features for recognizing visual concepts. *Evolutionary Computation, 4*(3), 297–311.

Baluja, S. (1994). *Population-based incremental learning: A method for integrating genetic search based function optimization and competitive learning* (Tech. Rep. No. CMU-CS-94-163). Pittsburgh, PA: Carnegie Mellon University.

Blake, C., & Merz, C. (1998). UCI repository of machine learning databases.

Brill, F. Z., Brown, D. E., & Martin, W. N. (1990). *Genetic algorithms for feature selection for counterpropogation networks* (Tech. Rep. No. IPC-TR-90-004). Charlottesville: University of Virginia, Institute of Parallel Computation.

Brotherton, T. W., & Simpson, P. K. (1995). Dynamic feature set training of neural nets for classification. In McDonnell, J. R., Reynolds, R. G., & Fogel, D. B. (Eds.), *Evolutionary Programming IV* (pp. 83–94). Cambridge, MA: MIT Press.

Etxeberria, R., & Larrañaga, P. (1999). Global optimization with Bayesian networks. In *II Symposium on Artificial Intelligence (CIMAF99).* (pp. 332–339).

Harik, G. (1999). *Linkage learning via probabilistic modeling in the ECGA* (IlliGAL Report No. 99010). Urbana, IL: University of Illinois at Urbana-Champaign, Illinois Genetic Algorithms Laboratory.

Harik, G. R., Lobo, F. G., & Goldberg, D. E. (1998). The compact genetic algorithm. In *Proceedings of 1998 IEEE International Conference on Evolutionary Computation* (pp. 523–528). Piscataway, NJ: IEEE Service Center.

Inza, I., Larrañaga, P., & Sierra, B. (2001). Feature subset selection by Bayesian networks: a comparison with genetic and sequential algorithms. *International Journal of Approximate Reasoning, 27*(2), 143–164.

Domain	All Features	sGA	cGA	ecGA	BOA
Ionosphere	83.37±2.65•	**91.79±1.67**	91.34±1.48	91.51±1.93	91.67±2.20
Segmentation	79.71±0.94•	89.44±1.39	**90.03±0.42**	89.77±1.32	89.60±0.96
Soybean Large	85.22±5.50	87.81±1.60	86.99±3.69	**88.20±1.74**	88.14±1.82
Sick Euthyroid	79.04±4.23•	95.73±1.07	95.74±0.85	**96.08±1.11**	95.83±0.86
Random21	94.02±0.86	95.23±0.89	**95.61±0.89**	95.01±0.81	95.44±0.85
Redundant21	76.89±1.32•	95.21±0.99	**95.42±0.66**	95.19±0.82	95.36±0.78

Table 2: Mean accuracies found (± standard deviation) in the 5x2cv experiments. The best result is in **bold** and a bullet (•) denotes a result that according to the combined F test is significantly different from the best result with 95% confidence.

Domain	Original	sGA	cGA	ecGA	BOA
Ionosphere	34•	11.9±2.92	12±3.09	**11.2±1.93**	**11.2±3.12**
Segmentation	19•	7.7±0.82	**7.4±0.70**	7.7±0.67	7.9±0.99
Soybean Large	25•	9.8±1.75•	8.2±1.62	7.7±2.05	**7.2±1.87**
Sick Euthyroid	35•	**11.2±3.32**	11.7±2.94	**11.2±4.34**	11.9±1.97
Random21	21•	**10.7±1.49**	11.5±1.51	11.1±1.28	11.1±1.37
Redundant21	21•	9.1±0.32•	**9±0**	**9±0**	**9±0**

Table 3: Mean sizes of final feature subsets (± standard deviation).

Inza, I., Larrañaga, P., Etxeberria, R., & Sierra, B. (1999). Feature subset selection by Bayesian networks based on optimization. *Artificial Intelligence*, *123*(1-2), 157–184.

Jain, A., & Zongker, D. (1997). Feature selection: evaluation, application and small sample performance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *19*(2), 153–158.

John, G., Kohavi, R., & Phleger, K. (1994). Irrelevant features and the feature subset problem. In *Proceedings of the 11th International Conference on Machine Learning* (pp. 121–129). Morgan Kaufmann.

Kelly, J. D., & Davis, L. (1991). Hybridizing the genetic algorithm and the K nearest neighbors classification algorithm. In Belew, R. K., & Booker, L. B. (Eds.), *Proceedings of the Fourth International Conference on Genetic Algorithms* (pp. 377–383). San Mateo, CA: Morgan Kaufmann.

Kohavi, R., & John, G. (1997). Wrappers for feature subset selection. *Artificial Intelligence*, *97*(1-2), 273–324.

Larrañaga, P., Etxeberria, R., Lozano, J. A., & Peña, J. M. (1999). *Optimization by learning and simulation of Bayesian and Gaussian networks* (Tech Report No. EHU-KZAA-IK-4/99). Conostia-San Sebastian, Spain: University of the Basque Country.

Lobo, F. G., & Harik, G. R. (1999). *Extended compact genetic algorithm in C++* (IlliGAL Report No. 99016). Urbana, IL: University of Illinois at Urbana-Champaign, Illinois Genetic Algorithms Laboratory.

Miller, B. L., & Goldberg, D. E. (1996). Genetic algorithms, selection schemes, and the varying effects of noise. *Evolutionary Computation, 4*(2), 113–131.

Mühlenbein, H. (1998). The equation for the response to selection and its use for prediction. *Evolutionary Computation*, *5*(3), 303–346.

Mühlenbein, H., & Mahnig, T. (1999). FDA-A scalable evolutionary algorithm for the optimization of additively decomposed functions. *Evolutionary Computation*, *7*(4), 353–376.

Pelikan, M. (1999). *A simple implementation of the Bayesian optimization algorithm (BOA) in C++ (version 1.0)* (IlliGAL Report No. 99011). Urbana, IL: University of Illinois at Urbana-Champaign, Illinois Genetic Algorithms Laboratory.

Pelikan, M., Goldberg, D. E., & Cantú-Paz, E. (1999). BOA: The Bayesian optimization algorithm. In Banzhaf, W., Daida, J., Eiben, A. E., Garzon, M. H., Honavar, V., Jakiela, M., & Smith, R. E. (Eds.), *Proceedings of the Genetic and Evolutionary Computation Conference 1999: Volume 1* (pp. 525–532). San Francisco, CA: Morgan Kaufmann Publishers.

Pelikan, M., Goldberg, D. E., & Lobo, F. (1999). *A survey of optimization by building and using probabilistic models* (IlliGAL Report No. 99018). Urbana, IL: University of Illinois at Urbana-Champaign, Illinois Genetic Algorithms Laboratory.

Pelikan, M. & Goldberg, D. E. (2000). *Research on the Bayesian optimization algorithm.* (IlliGAL Report No. 2000010). Urbana, IL: University of Illinois

at Urbana-Champaign, Illinois Genetic Algorithms Laboratory.

Punch, W. F., Goodman, E. D., Pei, M., Chia-Shun, L., Hovland, P., & Enbody, R. (1993). Further research on feature selection and classification using genetic algorithms. In Forrest, S. (Ed.), *Proceedings of the Fifth International Conference on Genetic Algorithms* (pp. 557–564). San Mateo, CA: Morgan Kaufmann.

Raymer, M. L., Punch, W. F., Goodman, E. D., Kuhn, L. A., & Jain, A. K. (2000). Dimensionality reduction using genetic algorithms. *IEEE Transactions on Evolutionary Computation*, *4*(2), 164–171.

Raymer, M. L., Punch, W. F., Goodman, E. D., Sanschagrin, P. C., & Kuhn, L. A. (1997). Simultaneous feature scaling and selection using a genetic algorithm. In Bäck, T. (Ed.), *Proceedings of the Seventh International Conference on Genetic Algorithms* (pp. 561–567). San Francisco: Morgan Kaufmann.

Siedlecki, W., & Sklansky, J. (1989). A note on genetic algorithms for large-scale feature selection. *Pattern Recognition Letters*, *10*, 335–347.

Thierens, D. (1999). Scalability problems of simple genetic algorithms. *Evolutionary Computation*, *7*(4), 331–352.

Vafaie, H., & Jong, K. A. D. (1993). Robust feature selection algorithms. In *Proceedings of the International Conference on Tools with Artificial Intelligence* (pp. 356–364). IEEE Computer Society Press.