

# Feature Selection for Support Vector Machines by Means of Genetic Algorithms

Holger Fröhlich

holger.froehlich@informatik.uni-tuebingen.de

Olivier Chapelle

olivier.chapelle@tuebingen.mpg.de

Bernhard Schölkopf

bernhard.schoelkopf@tuebingen.mpg.de

Department Empirical Inference, Max-Planck-Institute of biological Cybernetics, Tübingen, Germany

## Abstract

*The problem of feature selection is a difficult combinatorial task in Machine Learning and of high practical relevance, e.g. in bioinformatics. Genetic Algorithms (GAs) offer a natural way to solve this problem. In this paper we present a special Genetic Algorithm, which especially takes into account the existing bounds on the generalization error for Support Vector Machines (SVMs). This new approach is compared to the traditional method of performing cross-validation and to other existing algorithms for feature selection.*

## 1. Introduction

In many practical pattern classification tasks we are confronted with the problem, that we have a very high dimensional input space and we want to find out the combination of the original input features which contribute most to the classification. Supposed we want to classify cells into cancer or not based upon their gene expressions. Surely on one hand we want to have a combination of genes as small as possible. On the other hand we want to get the best possible performance of the learning machine.

Assumed we have training data  $\mathcal{D} = \{(\mathbf{x}_i, y_i) | \mathbf{x}_i \in \mathcal{X}, y_i \in \mathcal{Y}, i = 1, \dots, n\}$  (drawn i.i.d. from some unknown probability distribution  $P(\mathbf{x}, y)$ ) where  $\mathcal{X}$  is a vector space of dimension  $d$  and  $\mathcal{Y}$  is a finite set of labels. Then the problem of feature selection can be formally addressed in the following two ways [19]:

1. Given  $m \ll d$ , find out the  $m$  features that give the smallest expected generalization error; or
2. Given a maximum allowable generalization error  $\gamma$ , find the smallest number  $m$  of features.

Unlike e.g. Gaussian Processes in regression, Support Vector Machines (SVMs) do not offer the opportunity of an automated internal relevance detection and hence algorithms for feature selection play an important role. In the literature there are known two general approaches to solve the feature selection problem: The filter approach and the wrapper approach [10]: In a filter method feature selection is performed as a preprocessing step to the actual learning algorithm, i.e. before applying the classifier to the selected feature subset. Features are selected with regard to some predefined relevance measure which is independent of the actual generalization performance of the learning algorithm. This can mislead the feature selection algorithm [10]. Wrapper methods, on the other hand, train the classifier system with a given feature subset as an input and return the estimated generalization performance of the learning machine as an evaluation of the feature subset. This step is repeated for each feature subset taken into consideration. Depending on the bias and variance of the used estimator this method induces some overfitting of the wrapper algorithm. Traditionally  $k$ -fold cross-validation is used as an estimator of the generalization error.

In this paper we are taking existing theoretical bounds on the generalization error for SVMs instead of performing cross-validation. This is computationally much faster than  $k$ -fold cross-validation, since we do not have to retrain  $k$  times on each feature subset but just once. Additionally we will see that although in general the error bounds have a higher bias than cross-validation in practical situations they often have a lower variance and can thus reduce the overfitting of the wrapper algorithm. The feature selection process is done in an evolutionary way. The genetic encoding also allows us to optimize different parameters of the SVM, like the regularization parameter  $C$  in parallel. As far as we know, this is completely new approach, since previous work (e.g. [14]) on feature selection by means of Genetic Algorithms (GAs) neither took advantage of the generalization error bounds nor on the possibility of an inter-

nal kernel parameter optimization. This paper is organized as follows: In the next section we will first review different possibilities of estimating the generalization error of a SVM. Afterwards several settings of GAs for feature selection will be presented. Our new approaches will be compared to each other as well as to the traditional approach by performing cross-validation and to three popular existing algorithms for feature selection on two artificial and two real life data sets in section 4. Section 5 is a critical discussion, and section 6 draws a general conclusion.

## 2. Estimating the Generalization Performance of a SVM

As well known, SVM training [4, 15] is done by performing the following optimization procedure:

$$\max_{\alpha} W^2(\alpha) = \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i,j=1}^n \alpha_i \alpha_j y_i y_j k(x_i, x_j) \\ \text{subject to } 0 \leq \alpha_i \text{ for all } i = 1, \dots, n, \text{ and } \sum_{i=1}^n \alpha_i y_i = 0 \quad (1)$$

where  $k$  is the kernel function. One can deal with the non-separable case by e.g. mapping  $k(x_i, x_j) \mapsto k(x_i, x_j) + \frac{1}{C} \delta_{ij}$  with  $C$  being a regularization parameter and  $\delta$  the Kronecker symbol (c.f.[3]). This is commonly referred to as 2-norm C-SVM.

A goal of every classifier  $f$  is to minimize the expected generalization error (or risk) over all possible patterns drawn from the unknown distribution  $\mathcal{P}(\mathbf{x}, y)$  (see e.g. [16])

$$R[f] = \int_{\mathbf{x} \times \mathcal{Y}} \ell(\mathbf{x}, y, f(\mathbf{x})) d\mathcal{P}(\mathbf{x}, y) \quad (2)$$

with  $\ell$  being some loss-function. However, since (2) is not computable (as we don't know  $\mathcal{P}$ ), we are forced to estimate the generalization performance. Besides statistical general methods, like cross-validation, for SVM there exist theoretical bounds on the leave-one-out error ([16, 17, 2]). Three of them are cited here:

**Theorem 1.** (Vapnik [16]) *Let  $\rho$  be the size of the maximal margin and  $\phi(\mathbf{x}_1), \dots, \phi(\mathbf{x}_n)$  the images of the training patterns in feature space which are lying within a sphere of radius  $R$ . Let  $\alpha^*$  be the tuple of Lagrangian multipliers which are obtained by maximizing functional (1). If the images of the training data of size  $n$  belonging to a sphere of radius  $R$  are separable with the corresponding margin  $\rho$ , then the expectation of the test error probability  $P^{n-1}$  has the bound*

$$EP_{err}^{n-1} \leq \frac{1}{n} E \left\{ \frac{R^2}{\rho^2} \right\} = \frac{1}{n} E \{ R^2 W^2(\alpha^*) \} \quad (3)$$

where expectation is taken over samples of size  $n - 1$ .

Following [15] one can calculate  $R^2$  by maximizing:

$$R^2 = \max_{\beta} \sum_i \beta_i k(\mathbf{x}_i, \mathbf{x}_i) - \sum_{i,j} \beta_i \beta_j k(\mathbf{x}_i, \mathbf{x}_j) \\ \text{subject to } \sum_i \beta_i = 1, \beta_i \geq 0, i = 1, \dots, n \quad (4)$$

**Theorem 2.** (Vapnik, Chapelle [17, 2]) *Let  $\alpha^*$  be the tuple of Lagrangian multipliers which are obtained by maximizing functional (1). Supposed we have a SVM without threshold. Under the assumption that the set of support vectors does not change when removing example  $p$  we have*

$$EP_{err}^{n-1} \leq \frac{1}{n} E \left\{ \sum_{p=1}^n \psi \left( \frac{\alpha_p^*}{(\mathbf{K}_{SV}^{-1})_{pp}} - 1 \right) \right\} \quad (5)$$

where  $\psi : \mathbb{R} \rightarrow \{0, 1\}$  is the step function  $\psi(v) = \begin{cases} 0 & : v \leq 0 \\ 1 & : \text{otherwise} \end{cases}$ ,  $\mathbf{K}_{SV}$  is the matrix of dot products between support vectors in feature space,  $P_{err}^{n-1}$  is the probability of test error for the machine trained on a sample of size  $n - 1$  and the expectations are taken over the random choice of the sample.

Note that this theorem assumes a SVM without threshold. Thus in general one has to transform the data in feature space beforehand to ensure this condition, e.g. by  $\phi(\mathbf{x}_i) \mapsto \phi(\mathbf{x}_i) - \frac{1}{n} \sum_{j=1}^n \phi(\mathbf{x}_j)$ .

As the computation of the inverse of the matrix  $\mathbf{K}_{SV}$  is rather expensive, one can upper bound  $(\mathbf{K}_{SV}^{-1})_{pp}$  by  $\frac{1}{k(\mathbf{x}_p, \mathbf{x}_p)}$  [3]. Thus one recovers the Jaakkola-Haussler bound [9]:

$$EP_{err}^{n-1} \leq \frac{1}{n} E \left\{ \sum_{p=1}^n \psi (\alpha_p^* k(\mathbf{x}_p, \mathbf{x}_p) - 1) \right\} \quad (6)$$

Other bounds also exist, but they will not be taken into consideration here. For the computation of all these bounds the SVM has to be trained just once rather than  $k$  times when performing  $k$ -fold cross-validation.

## 3. Genetic Algorithms for Feature Selection using SVMs

### 3.1. Encodings and Fitness Functions

Let us first put our focus on the case where the number of features to be selected is not known beforehand (problem 2 on the preceding page). In this case we have a search space of size  $2^d$ . We can represent this search space by a standard binary encoding where a "1" indicates the selection of the feature at the corresponding position. The fitness of each feature subset is evaluated by either bound (3) or by bound (6) or, in the traditional way, by performing cross-validation. We will call the corresponding algorithms GAR2W2, GAJH respectively GAAcc. To break a tie for smaller feature subsets the generalization error estimate is multiplied by the term  $1 + 0.001 \cdot \frac{\#\{\text{features selected}\}}{d}$ , where the constant 0.001 aims to be lower than the standard deviation of the estimate of the generalization performance for a given feature subset.

The term  $R^2$  in bound (3) is estimated by the expression  $\frac{1}{n} \sum_{i=1}^n k(\mathbf{x}_i, \mathbf{x}_i) - \frac{1}{n^2} \sum_{i,j=1}^n k(\mathbf{x}_i, \mathbf{x}_j)$ , which means that every  $\beta_i = \frac{1}{n}$  in (4). In the multiclass case one can compute one of the bounds pair wise for classes  $c$  and  $c'$  and sum it up.

The GA used here is the CHC algorithm by Eshelman [5], which was reported to perform a more aggressive and faster search strategy than the traditional Simple GA [7]. One of the main ideas of CHC is the so called *population elitist* strategy: The best individuals of the child-generation replace the worst individuals of the parent-generation. Individuals are selected randomly for recombination, but they are only allowed to mate, if their genomes are not too close (with regard to some metric). Then half of the differing genes between parents are exchanged to produce children. If no recombination is possible any more, because the individuals are too similar, the algorithm is restarted. The best individual is left untouched and the others are created by heavy mutation of the best individual.

In the case where the number of features to be selected is known beforehand (problem 1 on page 1) a binary encoding in the prior way due to the much smaller search space of size  $\binom{d}{m}$  would be inefficient. Therefore in this case it is reasonable to switch to a decimal encoding  $(c_1, \dots, c_m)$  where  $c_i \in \{1, \dots, d\}$  ( $i = 1, \dots, m$ ) indicates the number of the feature which is selected. Of course one has to make sure that each  $c_i$  is unique in the code. Because the genomes are much shorter with decimal index encoding than with binary encoding, the probability to mutate was set from 35% in the standard CHC to 50% for each gene. In analogy to the binary case we will call the GAs in dependence of their used fitness function  $m$ -GAR2W2,  $m$ -GAJH respectively  $m$ -GAAcc.

GAAcc, GAR2W and GAJH were stopped, if the best solution didn't change for the last 200 generations at least 10,000 generations had been computed. For the  $m$ -GAR2W2,  $m$ -GAJH,  $m$ -GAAcc at least 1,000 generations were necessary.

### 3.2. Optimizing Kernel Parameters with Genetic Algorithms

The GAs described above can also be used for the optimization of the regularization parameter  $C$ . If we have a binary genome  $(b_1, \dots, b_d)$   $b_i \in \{0, 1\}$  we can simply concatenate a binary representation of the parameter  $C$  to our existing chromosome and run GAAcc, GAR2W2 respectively GAJH on this new genome. That means we are trying to select an optimal feature subset and an optimal  $C$  at the same time. This is reasonable, because the choice of the parameter  $C$  is influenced by the feature subset taken into account and vice versa. Usually it is not necessary to consider any

arbitrary value of  $C$ , but only certain discrete values, e.g. 0.001, 0.01, ..., 1000 respectively  $10^{-3}$ , ...,  $10^3$ . With just 3 bits we can code the numbers 0, ..., 7, or, if we shift this representation by -3, the numbers -3, ..., 4 which can be interpreted as powers of 10. In a similar manner one could use GAs to optimize e.g. the width of an RBF kernel.

## 4. Experiments

### 4.1. Algorithms and Tests

We will compare the GAs described above to each other and to the Fisher Criterion Score (e.g. [18]), Relief-F [11] and Recursive Feature Elimination (RFE) [8] algorithm. Fisher Criterion Score and Relief-F are two filter methods and RFE is a wrapper method, which was especially designed for SVMs. Fisher Criterion Score, Relief-F, and RFE can only select a given number of features and hence one has to try different numbers  $m$  of features systematically. If we want to compare these algorithms to GAAcc, GAR2W2 and GAJH, we will consider the situation where the least test error was reached. To test whether certain observed differences between test errors are statistically significant, the corrected resampled  $t$ -test recently proposed by C. Nadeau and Y. Bengio is used [12]. This test behaves very conservatively, which means the test might state a certain observed difference of results is not significant although it actually is, more often than one might expect by the prescribed significance level of 5%.

### 4.2. Data Sets

**4.2.1. Toy Data** The artificial data set was created as described in [19]: Six dimensions of 202 are relevant. The probability of the two classes  $y = 1$  and  $y = -1$  was equal. The first three features  $\{X_1, X_2, X_3\}$  are drawn as  $X_i = y\mathcal{N}(i, 1)$  and the second three features  $\{X_4, X_5, X_6\}$  were drawn as  $X_i = \mathcal{N}(0, 1)$  with a probability of 0.7, otherwise the first three were drawn as  $X_i = \mathcal{N}(0, 1)$  and the second three as  $X_i = y\mathcal{N}(i - 3, 1)$ . The remaining features are noise  $X_i = \mathcal{N}(0, 20)$   $i = 7, \dots, 202$ , and the first six features still have redundancy (for more details see [6]).

### 4.2.2. Real Life Data

**Colon Cancer** The Colon cancer problem is described e.g. in [1]: 62 tissue samples probed by DNA microarrays contain 22 normal and 40 colon cancer examples. these two classes have to be discriminated by the expression of 2,000 genes.

**Yeast Data Set** The Yeast microarray data set (Brown Yeast data set, see e.g. [13]) consists of 208 genes that have to be discriminated into 5 classes based on 79 gene expres-

sion values corresponding to different experimental conditions.

### 4.3. Results

**4.3.1. Toy Data** 30 instances of 100 training points corresponding 30 trials were created. On each trial the solution returned by an algorithm was tested on 10,000 independent test points. All data was scaled to the interval  $[-1, 1]$ . For the GAAcc algorithm the fitness of each individual was determined by means of 4-fold cross-validation. In the experiment where the number of features to be selected is fixed,  $m$  was set to 2, 4. A 2-norm C-SVM with a linear kernel was used. The regularization parameter  $C$  which induced the best average test error was picked from the discrete values  $10^{-3}, \dots, 10^3$ . Results are shown in tables 1 and 4. The difference between filter methods and wrapper methods for  $m = 2$  is significant. This shows the principal problems of filter selection methods.

### 4.3.2. Real Life Data

**Colon Cancer** The data was split into a training set of 50 and a test set of 12 for 50 times, and results were averaged over these 50 trials. The data was normalized to mean 0 and standard deviation 1 for each feature. For the GAAcc algorithm the fitness of each individual was determined by means of 10-fold cross-validation. In the experiment where the number of features to be selected is fixed,  $m$  was set to 20, 50, 100, 250, 500, 1000. A linear kernel with a 2-norm C-SVM was used. The parameter  $C$  in the cost function for the C-SVM was set by the same procedure as described above. The results are shown in tables 2 and 5.

Form  $m$  fixed  $m$ -GAAcc is the overall worst performing algorithm, especially if  $m$  is small. For  $m = 20$  the difference to the other methods is significant. This can be interpreted as an overfitting problem due to a higher variance of the cross-validation estimate of the generalization error. As the number of subsets to be evaluated is higher the smaller  $m$  is, the overfitting problem becomes less serious with increasing  $m$ .

If  $m$  is not fixed, GAR2W2 and GAJH are both better than GAAcc again. For  $C = 0.01$  GAAcc performs significantly worse than the best Relief-F algorithm. In the case where  $C$  is not fixed, the difference between GAAcc and GAR2W2/GAJH is significant. Both GAR2W2 and GAJH seem to win something if  $C$  is optimized by the GA, whereas GAAcc does not. GAR2W2 and GAJH reach approximately the same test error as the best RFE in this case. They are only slightly worse than the best Relief-F algorithm.

**Yeast Data Set** 8-fold cross-validation was performed on this data for all methods. The data was normalized to mean 0 and standard deviation 1 for each feature. For the GAAcc

algorithm the fitness of each individual was determined by means of 7-fold cross-validation. In the experiment where the number of features to be selected is fixed,  $m$  was set to 10, 20, 40. A linear kernel with a 2-norm C-SVM was used which discriminates classes pairwise (one-versus-one method). The parameter  $C$  in the cost function for the C-SVM was set by the same procedure as described above. The results are shown in tables 3 and 6.

For  $m = 10$  Relief-F is significantly worse than 10-GAR2W2. Like in prior experiments  $m$ -GAR2W2 and  $m$ -GAJH are always better than  $m$ -GAAcc. Again the GAs using bounds seem to win something, if  $C$  is optimized during the search process, while this is not the case for the GAAcc algorithm.

## 5. Discussion

As a general conclusion from the experiments shown above one can state the following:

- GAs using cross-validation to evaluate a given feature subset show in some cases a significant overfitting problem.
- Using leave-one-out error bounds instead is an alternative. It leads to a better generalization performance in most cases, but if the number of features to select is not fixed beforehand, a higher number of features is selected than with cross-validation.
- Optimizing kernel parameters within the GA is useful, especially if leave-one-out error bounds are used.
- GAs using the  $R^2W^2$  bound show a comparable generalization performance to RFE. The number of selected features is in most cases a bit higher. If the number of features is not fixed beforehand and kernel parameters are optimized within the Genetic Algorithm, one can in fact save time by using such an algorithm instead of running RFE multiple times to determine kernel parameters and an appropriate number of features. E.g. one run of GAR2W2 with parameter optimization on the Colon data set needs about 30 minutes on a Pentium IV with 2GHz. One run of RFE takes about 2 minutes, but to determine the regularization parameter (0.001, 0.01, 0.1, 1, 10, 100, 1000) and the appropriate number of features (20, 50, 100, 250, 500, 1000) one would need all in all  $7 \cdot 6 = 42$  runs. This takes  $42 \cdot 2 = 84$  minutes, which is almost 3 times more than one run of GAR2W2. If the number of features to select is known beforehand, however, GAs do not offer these advantages any more.

## 6. Conclusion

In this paper we dealt with the problem of feature selection for SVMs by means of GAs. In contrast to the traditional way of performing cross-validation to estimate the generalization error induced by a given feature subset we proposed to use the theoretical bounds on the generalization error for SVMs, which is computationally attractive. If the number of features to be selected is fixed and hence the search space is much smaller than with a variable number of selected features, we proposed a decimal encoding, which is much more efficiently than a binary encoding. If the number of features to be selected is not fixed beforehand, the usual binary encoding was taken. Additionally to the selection of a feature subset, one can optimize kernel parameters such as the regularization parameter  $C$  of the SVM by means of GAs. This is reasonable, because the choice of the feature subset has an influence on the appropriate kernel parameters and vice versa.

Existing algorithms such as Fisher Criterion Score, Relief-F and Recursive Feature Elimination were compared to GAs using cross-validation and to GAs using two different error bounds on two toy problems and two DNA micro array data sets. Hereby Recursive Feature Elimination is a heuristic wrapper algorithm which was especially designed for SVMs, and Fisher Criterion Score and Relief-F are two filter algorithms. As a conclusion of the experiments one can state that GAs using the  $R^2W^2$  bound and optimizing various kernel parameters are a recommendable alternative, if the number of features to select is not known beforehand. It reduces overfitting in comparison with  $k$ -fold cross-validation in most of our experiments, because of a lower variance of the generalization error estimate. Additionally, in comparison with running RFE multiple times to determine the kernel parameters and an appropriate feature subset, one in fact saves time.

## References

- [1] U. Alon, N. Barkai, D. Notterman, K. Gish, S. Ybarra, D. Mack, and A. Levine. Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon cancer tissues probed by oligonucleotide arrays. *Cell Biology*, 96:6745 – 6750, 1999.
- [2] O. Chapelle and V. Vapnik. Model selection for Support Vector Machines. In S. Solla, T. Leen, and K.-R. Müller, editors, *Advances in Neural Information Processing Systems 12*, Cambridge, MA, 2000. MIT Press.
- [3] O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee. Choosing Multiple Parameters for Support Vector Machines. *Machine Learning*, 46(1):131 – 159, 2002.
- [4] C. Cortes and V. Vapnik. Support vector networks. *Machine Learning*, 20:273 – 297, 1995.
- [5] L. Eshelman. *The CHC Adaptive Search Algorithm, How to Have Safe Search When Engaging in Non-traditional Genetic Recombination*. Morgan Kaufman, 1991.
- [6] H. Fröhlich. Feature Selection for Support Vector Machines by Means of Genetic Algorithms. Master's thesis, University of Marburg, 2002. <http://www-ra.informatik.uni-tuebingen.de/mitarb/froehlich>.
- [7] D. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison Wesley, Reading, 1998.
- [8] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene Selection for Cancer Classification using Support Vector Machines. *Machine Learning*, 46:389 – 422, 2002.
- [9] T. S. Jaakkola and D. Haussler. Probabilistic kernel regression models. In *Proceedings of the 1999 Conference on AI and Statistics*, 1999.
- [10] R. Kohavi and G. John. Wrappers for Feature Subset Selection. *Artificial Intelligence*, 97(12):273 – 324, 1997.
- [11] I. Kononenko and S. J. Hong. Attribute Selection for Modeling. *Future Generation Computer Systems*, 13(2 - 3):181 – 195, 1997.
- [12] C. Nadeau and Y. Bengio. Inference for the Generalization Error. In S. Solla, T. Leen, and K.-R. Müller, editors, *Advances in Neural Information Processing Systems 12*, Cambridge, MA, 2000. MIT Press.
- [13] P. Pavlidis, J. Weston, J. Cai, and W. Grundy. Gene functional classification from heterogeneous data. In *Proceedings of the fifth International Conference on Computational Molecular Biology*, pages 242 – 248, 2001.
- [14] S. Salcedo-Sanz, M. Prado-Cumplido, F. Perez-Cruz, and C. Bousono-Calzon. Feature Selection via Genetic Optimization. In *Proc. ICANN 2002*, pages 547 – 552, 2002.
- [15] B. Schölkopf, C. Burges, and V. Vapnik. Extracting support data for a given task. In U. N. Fayyad and R. Uthurusamy, editors, *First International Conference for Knowledge Discovery and Data Mining*, Menlo Park, 1995. AAAI Press.
- [16] V. Vapnik. *Statistical Learning Theory*. John Wiley and Sons, New York, 1998.
- [17] V. Vapnik and O. Chapelle. Bounds on error expectation for Support Vector Machines. *Neural Computation*, 12(9), 2000.
- [18] J. Weston, A. Elisseeff, B. Schölkopf, and M. Tipping. Use of the zero-norm with linear models and kernel methods. *JMLR special Issue on Variable and Feature Selection*, 3:1439 – 1461, 2002.
- [19] J. Weston, S. Mukherjee, O. Chapelle, M. Pontil, T. Poggio, and V. Vapnik. Feature selection for SVMs. In S. Solla, T. Leen, and K.-R. Müller, editors, *Advances in Neural Information Processing Systems 13*, Cambridge, MA, 2001. MIT Press.

Exp. type		Method (average % test error $\pm$ standard deviation)						
m	C	GAAcc	GAR2W2	GAJH	Fisher	Relief-F	RFE	no sel.
2	1	3.4 $\pm$ 3	2.8 $\pm$ 1.6	2.8 $\pm$ 1.6	15 $\pm$ 0.4	15 $\pm$ 0.4	<b>2.8 <math>\pm</math> 1.6</b>	9.3 $\pm$ 5
4	1	2.6 $\pm$ 1.3	1.3 $\pm$ 1.3	1.2 $\pm$ 0.8	1.8 $\pm$ 2	1.8 $\pm$ 2	<b>1.1 <math>\pm</math> 0.7</b>	9.3 $\pm$ 5
n. f. <sup>a</sup>	1	<b>1.6 <math>\pm</math> 1.2</b>	2.5 $\pm$ 1.5	3.4 $\pm$ 3.8				9.3 $\pm$ 5
	n. f.	<b>1.5 <math>\pm</math> 1</b>	3.2 $\pm$ 2	3.5 $\pm$ 2.1				

<sup>a</sup> n. f. = not fixed

**Table 1. Toy problem: For  $m = 2$  all wrapper methods find the best separating features (c.f. [6]) 3 and 6 in most cases (2-GAR2W2, 2-GAJH, RFE 29/30, 2-GAAcc 27/30), whereas Fisher Criterion Score and Relief-F select features 2 and 3 in all respectively in 29/30 cases. For  $m = 4$  RFE performs best by finding the best separating features 2, 3, 5, 6 in 27/30 cases. 4-GAR2W2 and 4-GAJH select these features in 25/30 cases, and 4-GAAcc only in 5/30 cases. The filter algorithms use them in only 4/30 cases and select features 1, 2, 3, 6 in 20 (Fisher Criterion Score) respectively 24 cases (Relief-F) instead.**

Exp. type		Method (average % test error $\pm$ standard deviation)						
m	C	GAAcc	GAR2W2	GAJH	Fisher	Relief-F	RFE	no sel.
20	0.01	22.8 $\pm$ 10.5	15.8 $\pm$ 9.3	16.3 $\pm$ 10.4	15 $\pm$ 9.4	<b>14.7 <math>\pm</math> 8.8</b>	16.3 $\pm$ 9.5	16.2 $\pm$ 9.4
50	0.01	20.3 $\pm$ 10.3	16 $\pm$ 9	16.7 $\pm$ 9.4	14.8 $\pm$ 9.1	14.3 $\pm$ 9.8	16.2 $\pm$ 9.1	16.2 $\pm$ 9.4
100	0.01	19.7 $\pm$ 10.6	15.7 $\pm$ 9.5	17.2 $\pm$ 9.1	15.2 $\pm$ 9.7	<b>14.3 <math>\pm</math> 9.5</b>	16.3 $\pm$ 8.7	16.2 $\pm$ 9.4
250	0.01	17.2 $\pm$ 11	16.2 $\pm$ 8.8	16.7 $\pm$ 8.9	<b>15.7 <math>\pm</math> 9</b>	16.5 $\pm$ 10	15.8 $\pm$ 8.3	16.2 $\pm$ 9.4
500	0.01	16.3 $\pm$ 8.7	<b>15.3 <math>\pm</math> 9.1</b>	16.5 $\pm$ 9	17.2 $\pm$ 9	15.5 $\pm$ 9.4	16 $\pm$ 9.4	16.2 $\pm$ 9.4
1000	0.01	17.2 $\pm$ 8.5	16.7 $\pm$ 9.7	16.7 $\pm$ 9.4	17.8 $\pm$ 8.1	<b>16 <math>\pm</math> 9.3</b>	16.3 $\pm$ 9.4	16.2 $\pm$ 9.4
n. f. <sup>a</sup>	0.01	21.3 $\pm$ 12	<b>16.2 <math>\pm</math> 9</b>	17.5 $\pm$ 8.5				16.2 $\pm$ 9.4
	n. f.	21.7 $\pm$ 10.8	<b>15.5 <math>\pm</math> 8.7</b>	15.8 $\pm$ 9.3				

<sup>a</sup> n. f. = not fixed

**Table 2. Colon data set:  $m$ -GAJH gives the best results with just 20 features, Fisher Criterion Score with 50, Relief-F with 100, RFE with 250, and  $m$ -GAAcc and  $m$ -GAR2W2 need 500 features. The result obtained by 500-GAR2W2 is comparable to the RFE with 250 features and slightly worse than the Relief-F with 100 features, while the result of 500-GAAcc is almost the same as 20-GAJH.**

Exp. type		Method (average % test error $\pm$ standard deviation)						
m	C	GAAcc	GAR2W2	GAJH	Fisher	Relief-F	RFE	no sel.
10	100	5.8 $\pm$ 5	<b>4.3 <math>\pm</math> 4.8</b>	5.3 $\pm$ 4.1	8.7 $\pm$ 8.4	11.5 $\pm$ 6.2	5.8 $\pm$ 4.1	3.8 $\pm$ 3.6
20	100	6.7 $\pm$ 5.7	<b>5.3 <math>\pm</math> 5</b>	6.7 $\pm$ 5.7	5.8 $\pm$ 5.8	<b>5.3 <math>\pm</math> 3.5</b>	5.8 $\pm$ 5.8	3.8 $\pm$ 3.6
40	100	6.3 $\pm$ 4.6	3.8 $\pm$ 4.1	3.8 $\pm$ 4.1	3.9 $\pm$ 4.1	5.8 $\pm$ 4.6	<b>3.4 <math>\pm</math> 3.8</b>	3.8 $\pm$ 3.6
n. f. <sup>a</sup>	100	<b>4.8 <math>\pm</math> 4</b>	<b>4.8 <math>\pm</math> 4.8</b>	5.8 $\pm$ 5				3.8 $\pm$ 3.6
	n. f.	5.3 $\pm$ 5	<b>3.4 <math>\pm</math> 3.2</b>	4.3 $\pm$ 4.3				

<sup>a</sup> n. f. = not fixed

**Table 3. Yeast data set:  $m$ -GAAcc gives the best results with only 10 features, while Relief-F needs 20 and the other algorithms 40 features. If  $m$  is not fixed and  $C = 100$ , GAAcc performs the same as GAR2W2 and a bit worse than the best RFE and the best Fisher Criterion Score. But if  $C$  is determined by the GA GAR2W2 is better than GAAcc and GAJH.**

$C$	number of selected features (average number $\pm$ standard deviation)		
	<b>GAAcc</b>	<b>GAR2W2</b>	<b>GAJH</b>
1	<b>4 <math>\pm</math> 1</b>	13 $\pm$ 3	12 $\pm$ 4
not fixed	<b>4 <math>\pm</math> 1</b>	16 $\pm$ 4	18 $\pm$ 4

**Table 4. Toy problem – number of selected features: GAAcc selects 4 features on average for both  $C = 1$  and  $C$  not fixed, which is the same as for the best Fisher Criterion Score/Relief-F/RFE algorithm. In contrast to this GAR2W2 selects 13 respectively 16, and GAJH 12 respectively 18 features. The features selected by GAAcc where among the first 6 in 29/30 cases.**

$C$	number of selected features (average number $\pm$ standard deviation)		
	<b>GAAcc</b>	<b>GAR2W2</b>	<b>GAJH</b>
100	<b>42 <math>\pm</math> 20</b>	221 $\pm$ 14	236 $\pm$ 17
not fixed	<b>49 <math>\pm</math> 30</b>	382 $\pm$ 35	388 $\pm$ 35

**Table 5. Colon data set – number of selected features: GAR2W2 and GAJH select more features (around 200 respectively 400) than GAAcc (around 40 respectively 50). This difference corresponds to the results of the other data sets. It should be compared to the 250 features selected by the best RFE, 100 features selected by the best Relief-F and 50 features selected by the best Fisher Criterion Score algorithm.**

$C$	number of selected features (average number $\pm$ standard deviation)		
	<b>GAAcc</b>	<b>GAR2W2</b>	<b>GAJH</b>
100	<b>9 <math>\pm</math> 1</b>	23 $\pm$ 3	26 $\pm$ 3
not fixed	<b>8 <math>\pm</math> 1</b>	23 $\pm$ 2	23 $\pm$ 2

**Table 6. Yeast data set – number of selected features: GAAcc selects the fewest features (9  $\pm$  1/8  $\pm$  1 GAR2W2 23  $\pm$  3/23  $\pm$  2, GAJH 26  $\pm$  3/23  $\pm$  2 for  $C = 100/C$  not fixed) which should be compared to 40 features taken by the best Fisher Criterion Score and RFE respectively 20 taken by the best Relief-F.**