# Feature Subset Selection, Class Separability, and Genetic Algorithms

Erick Cantú-Paz

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

**Abstract.** The performance of classification algorithms in machine learning is affected by the features used to describe the labeled examples presented to the inducers. Therefore, the problem of feature subset selection has received considerable attention. Genetic approaches to this problem usually follow the wrapper approach: treat the inducer as a black box that is used to evaluate candidate feature subsets. The evaluations might take a considerable time and the traditional approach might be impractical for large data sets. This paper describes a hybrid of a simple genetic algorithm and a method based on class separability applied to the selection of feature subsets for classification problems. The proposed hybrid was compared against each of its components and two other feature selection wrappers that are used widely. The objective of this paper is to determine if the proposed hybrid presents advantages over the other methods in terms of accuracy or speed in this problem. The experiments used a Naive Bayes classifier and public-domain and artificial data sets. The experiments suggest that the hybrid usually finds compact feature subsets that give the most accurate results, while beating the execution time of the other wrappers.

## 1 Introduction

The problem of classification in machine learning consists of using labeled examples to induce a model that classifies objects into a set of known classes. The objects are described by a vector of features, some of which may be irrelevant or redundant and may have a negative effect on the accuracy of the classifier. There are two basic approaches to feature subset selection: wrapper and filter methods [1]. Wrappers treat the induction algorithm as a black box that is used by the search algorithm to evaluate each candidate feature subset. While giving good results in terms of the accuracy of the final classifier, wrapper approaches are computationally expensive and may be impractical for large data sets. Filter methods are independent of the classifier and select features based on properties that good feature sets are presumed to have, such as class separability or high correlation with the target. Although filter methods are much faster than wrappers, filters may produce disappointing results, because they completely ignore the induction algorithm.

K. Deb et al. (Eds.): GECCO 2004, LNCS 3102, pp. 959–970, 2004.

This paper presents a hybrid algorithm that combines the strengths of filters and wrappers and attempts to avoids their weaknesses. The hybrid consists of a simple genetic algorithm (sGA) used in its traditional role as a wrapper, but initialized with the output of a filter method based on a class separability metric. The objective of this study is to determine if the hybrid method presents advantages over simple GAs and conventional feature selection algorithms 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 [2].

Our target was to maximize the accuracy of classification. The experiments demonstrate that, in most cases, the proposed hybrid algorithm finds subsets that result in the best accuracy (or in an accuracy not significantly different from the best), while finding compact feature subsets, and performing faster than the wrapper methods.

The next section briefly reviews previous applications of EAs to feature subset selection. Section 3 describes the class separability filter and its hybridization with a GA. Section 4 describes the algorithms, data sets, and the fitness evaluation method used in the experiments reported in section 5. Section 6 concludes this paper with a summary and a discussion of future research directions.

## 2 Feature Selection

Reducing the dimensionality of the vectors of features that describe each object presents several advantages. As mentioned above, irrelevant or redundant features may affect negatively the accuracy of classification algorithms. In addition, reducing the number of features may help decrease the cost of acquiring data and might make the classification models easier to understand.

There are numerous techniques for dimensionality reduction. Some common methods seek transformations of the original variables to lower dimensional spaces. For example, principal components analysis reduces the dimensions of the data by finding orthogonal linear combinations with the largest variance. In the mean square error sense, principal components analysis yields the optimal linear reduction of dimensionality. However, it is not necessarily true that the principal components that capture most of the variance are useful to discriminate among objects of different classes. Moreover, the linear combinations of variables make it difficult to interpret the effect of the original variables on class discrimination. For these reasons, in the remainder of this paper we ignore methods that transform the features and we focus on techniques that select subsets of the original variables.

Among the feature subset algorithms, wrapper methods have received considerable attention. Wrappers are attractive because they seek to optimize the accuracy of a classifier, tailoring their solutions to a specific inducer and a domain. They search for a good feature subset using the induction algorithm to evaluate the merit of candidate subsets. Numerous search algorithms have been used to search for feature subsets [3]. Genetic algorithms are usually reported

to deliver good results, but exceptions have been reported where simpler (and faster) algorithms result in higher accuracies on particular data sets [3].

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 [4] 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 [5,6], decision trees [7], k-nearest neighbors [8,9,10,11], rules [12], and Naive Bayes [13, 14].

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 [8,9]. In this case, a value of zero in the transformation vector is equivalent to avoiding the feature. Raymer et al. [10,15] combined the linear transformation with explicit feature selection flags in the chromosomes, and reported an advantage over the pure transformation method.

More sophisticated Distribution Estimation Algorithms (DEAs) have also been used to search for optimal feature subsets. DEAs explicitly identify the relationships among the variables of a problem by building a model of selected individuals and using the model to generate new solutions. In this way, DEAs avoid the disruption of groups of related variables that might prevent a simple GA from reaching the global optimum. However, in terms of accuracy, the DEAs do not seem to outperform simple GAs when searching for feature subsets [13, 14,16,17]. For this reason, we limit this study to simple GAs.

The wrappers' evaluation of candidate feature subsets can be computationally expensive on large data sets. Filter methods are computationally efficient and offer an alternative to wrappers. Genetic algorithms have been used as filters in regression problems to optimize a cost function derived from the correlation matrix between the features and the target value [18]. GAs have also been used as a filter in classification problems minimizing the inconsistencies present in subsets of the features [19]. An inconsistency between two examples occurs if the examples match with respect to the feature subset considered, but their class labels disagree. Lanzi demonstrated that this filter method efficiently identifies feature subsets that were at least as predictive as the original set of features (the results were never significantly worse). However, the accuracy on the reduced subset is not much different (better or worse) than with all the features. In this study we show that the proposed method can reduce the dimensionality of the data and increase the predictive accuracy considerably.

## 3 Class Separability

The idea of using a measure of class separability to select features has been used in machine learning and computer vision [20,21]. The class separability filter that we propose calculates the class separability of each feature using the Kullback-Leibler (KL) distance between histograms of feature values. For each

feature, there is one histogram for each class. Numeric features are discretized using $\sqrt{|D|}/2$ equally-spaced bins, where $|D|$ is the size of the training data. The histograms are normalized dividing each bin count by the total number of elements to estimate the probability that the $j$-th feature takes a value in the $i$-th bin of the histogram given a class $n$, $p_j(d = i|c = n)$. For each feature $j$, we calculate the class separability as

$$\Delta_j = \sum_{m=1}^{c} \sum_{n=1}^{c} \delta_j(m, n), \tag{1}$$

where $c$ is the number of classes and $\delta_j(m, n)$ is the KL distance between histograms corresponding to classes $m$ and $n$:

$$\delta_j(m, n) = \sum_{i=1}^{b} p_j(d = i|c = m) \log \left( \frac{p_j(d = i|c = m)}{p_j(d = i|c = n)} \right), \tag{2}$$

where $b$ is the number of bins in the histograms. Of course, other distribution distance metrics could be used instead of KL distance.

The features are then sorted in descending order of the distances $\Delta_j$ (larger distances mean better separability). Heuristically, we consider that two features are redundant if their distances differ by less than 0.0001, and we eliminate the feature with the smallest distance. We eliminate irrelevant non-discriminative features with $\Delta_j$ distances less than 0.001.

The heuristics used to eliminate redundant and irrelevant features were calibrated using artificial data sets that are described later. We recognize that these heuristics may fail in some cases if the thresholds chosen are not adequate to a particular classification problem. However, perhaps the major disadvantage of the method is that it ignores pairwise (or higher) interactions among variables. It is possible that features that appear irrelevant (not discriminative) when considered alone are relevant when considered in conjunction with other variables. For example, consider the two-class data displayed in figure 3. Each of the features alone does not have discriminative power, but taken together the two features perfectly discriminate the two classes.

To explore combinations of features we decided to use a genetic algorithm. After running the filter algorithm, we have some knowledge about the relative importance of each feature considered individually. This knowledge is incorporated into the GA by using the relative distances to initialize the GA. The distances $\Delta_j$ are linearly normalized between 0.1 and 0.9 to obtain the probability $p_j$ that the $j$-th bit in the chromosomes is initialized to 1 (and thus that the corresponding feature is selected). By making the lower and upper limits of $p_j$ different from 0 and 1, we are able to explore combinations that include features that the filter had eliminated as redundant or irrelevant. It also allows a chance to delete features that the filter identified as important.

After the GA is initialized with the output of the filter, the GA runs as a wrapper feature selection algorithm. The GA manipulates a population of candidate feature subsets using conventional GA operators. Each candidate solution is evaluated using an estimate of the accuracy of a classifier on the feature subset indicated in the chromosome and the best solution is reported to the user.

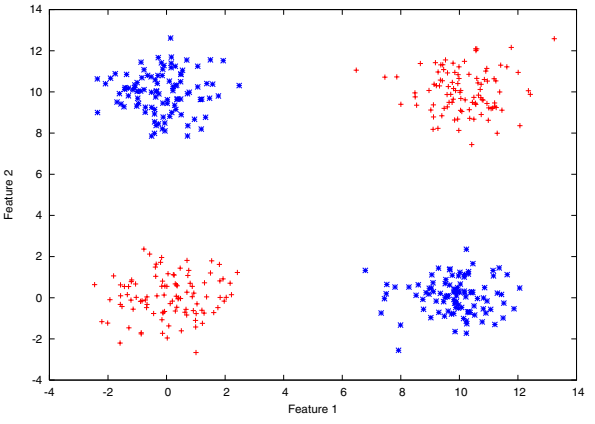


**Fig. 1.** Example of a data set where each feature considered alone does not discriminate between the two classes, but the two features taken together discriminate the data perfectly.

## 4 Methods

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

### 4.1 Algorithms and Data Sets

The GA used uniform crossover with probability 1.0, and mutation with probability $1/l$, where $l$ was the length of the chromosomes and corresponds to the total number of features in each problem. The population size was set to $\lfloor 3\sqrt{l} \rfloor$, following the Gambler's ruin model for population sizing that asserts that the population size required to reach a solution of a particular quality is $O(\sqrt{l})$ [22]. Promising solutions were selected with pairwise binary tournaments without replacement. The algorithms were terminated after observing no improvement of the best individual over consecutive generations. Inza et al. [13] and Cantú-Paz [14] used similar algorithms and termination criterion.

We compare the results of the class separability filter and the GAs with two traditional greedy feature selection algorithms. Greedy feature selection algorithms that add or delete a single feature from the candidate feature subset are common. There are two basic variants: sequential forward selection (SFS) and sequential backward elimination (SBE). Forward selection starts with an empty set of features. In the first iteration, the algorithm considers all feature subsets with only one feature. The feature subset with the highest accuracy is used as the basis for the next iteration. In each iteration, the algorithm tentatively adds to the basis each feature not previously selected and retains the feature subset that results in the highest estimated performance. The search terminates after

**Table 1.** Description of the data used in the experiments.

| Domain | Instances | Classes | Numeric Feat. | Nominal Feat. | Missing |
|---|---|---|---|---|---|
| Anneal | 898 | 6 | 9 | 29 | Y |
| Arrhythmia | 452 | 16 | 206 | 73 | Y |
| Euthyroid | 3163 | 2 | 7 | 18 | Y |
| Ionosphere | 351 | 2 | 34 | – | N |
| Pima | 768 | 2 | 8 | – | N |
| Segmentation | 2310 | 7 | 19 | – | N |
| Soybean Large | 683 | 19 | – | 35 | Y |
| Random21 | 2500 | 2 | 21 | – | N |
| Redundant21 | 2500 | 2 | 21 | – | N |

the accuracy of the current subset cannot be improved by adding any other feature. Backward elimination works in an analogous way, starting from the full set of features and tentatively deleting each feature not deleted previously.

The classifier used in the experiments was a Naive Bayes (NB). This classifier was chosen for its speed and simplicity, but the proposed hybrid method can be used with any other supervised classifiers. 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 algorithms were developed in C++ and compiled with g++ version 2.96 using -O2 optimizations. The experiments were executed on a single processor of a Linux (Red Had 7.3) workstation with dual 2.4 GHz Intel Xeon processors and 512 Mb of memory. A Mersenne Twister random number generator [23] was used in the GA and the data partitioning.

The data sets used in the experiments are described in table 1. With the exception of Random21 and Redundant21, the data sets are available in the UCI repository [24]. Random21 and Redundant21 are two artificial data sets with 21 features each and were proposed originally by Inza [13]. The target concept of these two data sets is 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.

### 4.2 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. 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 classifier 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. Alt-

hough there are well-known methods to deal with "noisy" fitness evaluations in EAs [25], 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 [2] in their study of wrapper methods for feature selection, and was adopted by Inza et al. [13]. 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.

## 5 Experiments

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 entire training data), which was then tested on the other half of the data. The accuracy results presented in table 2 are the mean and standard deviations of the ten tests.

To determine if the differences among the algorithms were statistically significant, we used a combined F test proposed by Alpaydin [26]. 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

$$f = \frac{\sum_{i=1}^{5} \sum_{j=1}^{2} (p_{i,j})^2}{2 \sum_{i=1}^{5} s_i^2}$$

is approximately F distributed with 10 and 5 degrees of freedom. We rejected the null hypothesis that the two algorithms have the same error rate at a 0.95 significance level if $f > 4.74$ [26]. Care was taken to ensure that all the algorithms used the same training and testing data in the two folds of the five crossvalidation experiments.

Table 2 has the mean accuracies obtained with each method. The best observed result in the table is highlighted in **bold** type as well as those results that according to the combined F test are not significantly different from the best at a 0.95 significance level. There are two immediate observations that we can make from the results. First, the feature selection algorithms result in an improvement

**Table 2.** Means and standard deviations of the accuracies found in the 5x2cv experiments. The best result and those not significantly different from the best are displayed in **bold**.

| Domain | Naive | Filter | FilterGA | sGA | SFS | SBE |
|---|---|---|---|---|---|---|
| Anneal | 89.93 2.72 | **93.43** 1.44 | **93.07** 2.89 | **92.47** 1.69 | **90.36** 2.37 | **93.47** 2.71 |
| Arrhythmia | 56.95 3.18 | **62.08** 2.52 | **64.16** 2.13 | 59.78 3.51 | 58.67 3.25 | 59.73 2.33 |
| Euthyroid | 87.33 3.23 | 89.06 0.41 | **94.20** 2.02 | **94.92** 0.74 | **94.57** 0.54 | **94.48** 0.42 |
| Ionosphere | 83.02 2.04 | **89.57** 1.29 | **90.54** 0.83 | **88.95** 2.14 | 85.23 2.76 | **89.17** 1.73 |
| Random21 | 93.89 0.81 | 82.24 2.32 | **95.41** 1.06 | **92.45** 3.96 | 82.12 1.70 | 80.61 2.13 |
| Pima | **74.87** 2.55 | **74.45** 2.23 | **75.49** 2.49 | **75.29** 2.57 | **73.46** 1.77 | **74.45** 1.71 |
| Redundant | 77.12 0.33 | 80.29 1.09 | **83.68** 2.94 | **86.70** 2.73 | 79.74 2.54 | 80.32 1.03 |
| Segment | 79.92 0.73 | 85.40 1.11 | **87.97** 1.12 | 84.73 2.37 | **90.85** 1.02 | **91.28** 0.93 |
| Soybean | **84.28** 4.72 | **86.01** 4.89 | **81.23** 5.73 | **81.79** 6.12 | 78.63 3.23 | **86.27** 5.00 |

**Table 3.** Means and standard deviations of the sizes of final feature subsets. The best result and those not significantly different from the best are in **bold**.

| Domain | Original | Filter | FilterGA | sGA | SFS | SBE |
|---|---|---|---|---|---|---|
| Anneal | 38 | 23.8 3.97 | 12.8 2.04 | 22.1 3.81 | **5.4** 0.92 | 16.4 9.54 |
| Arrhythmia | 279 | 212.5 16.30 | 86.2 6.42 | 138.9 4.99 | **3.9** 1.76 | 261.1 28.2 |
| Euthyroid | 25 | **1.0** 0.00 | 6.3 1.68 | 13.7 1.55 | **1.3** 0.64 | 1.2 0.40 |
| Ionosphere | 34 | 33.0 0.00 | 11.2 2.04 | 16.0 1.95 | **4.4** 1.56 | 30.9 1.76 |
| Pima | 8 | 4.3 2.87 | **2.9** 0.83 | 4.9 0.70 | **1.6** 0.66 | 5.3 1.00 |
| Random21 | 21 | **10.2** 3.60 | **10.3** 1.10 | 13.6 2.06 | **9.3** 0.90 | 12.6 4.48 |
| Redundant | 21 | **8.8** 0.40 | **8.1** 1.70 | 10.6 1.43 | **8.6** 0.92 | 9.1 0.70 |
| Segmentation | 19 | 11.0 0.00 | 9.9 1.51 | 9.6 1.69 | **4.0** 0.63 | 7.7 2.79 |
| Soybean Large | 35 | 32.9 1.51 | 19.50 2.11 | 21.7 2.15 | **10.6** 2.01 | 30.7 2.28 |

of accuracy over using a NB with all the features. However, this difference is not always significant (Soybean Large, Pima). Second, the proposed hybrid always reaches the highest accuracy or accuracies that are not significantly different from the highest. The simple GA with random initialization also performs very well, reaching results that are not significantly different from the best for all but two data sets.

In terms of the size of the final feature subsets (table 3), forward sequential selection consistently found the smallest subsets. This was expected, since this algorithm is heavily biased toward small subsets (because it starts from an empty set and adds features only when they show improvements in accuracy). However, in many cases SFS resulted in significantly worse accuracies than the proposed GA hybrid. The proposed hybrid found significantly—and substantially—smaller feature subsets than the filter alone or the sGA.

Table 4 shows the mean number of feature subsets examined by each algorithm. In most cases, the GAs examine fewer subsets than SFS and SBE, and the FilterGA examined fewer subsets than the GA initialized at random. This suggests that the search of the FilterGA was highly biased toward good solutions.

**Table 4.** Means and standard deviations of the number of feature subsets examined by each algorithm. The best result and those not significantly different from the best are in **bold**.

| Domain | FilterGA | | sGA | | SFS | | SBE | |
|---|---|---|---|---|---|---|---|---|
| Anneal | **38.84** | 19.31 | 48.08 | 32.24 | 225.50 | 29.46 | 569.20 | 185.49 |
| Arrhythmia | **105.23** | 26.98 | 120.26 | 40.09 | 1356.0 | 480.76 | 4706.9 | 6395.16 |
| Euthyroid | **36.00** | 28.62 | 37.50 | 18.06 | 55.8 | 14.46 | 324.8 | 0.40 |
| Ionosphere | **38.48** | 21.85 | **41.98** | 23.73 | 170.5 | 45.73 | 131.5 | 53.18 |
| Pima | **12.73** | 6.84 | 20.36 | 6.79 | **18.5** | 3.77 | 24.1 | 4.83 |
| Random21 | **35.74** | 20.58 | 64.61 | 34.81 | 168.0 | 10.68 | 147.9 | 59.35 |
| Redundant21 | **32.99** | 23.17 | 42.62 | 46.19 | 159.9 | 11.85 | 193.9 | 6.43 |
| Segmentation | **37.92** | 32.27 | **30.08** | 23.43 | 84.8 | 9.17 | 160.3 | 21.35 |
| Soybean Large | **42.60** | 25.35 | **42.60** | 22.73 | 342.5 | 47.39 | 171.5 | 67.46 |

**Table 5.** Execution time (in CPU seconds) of the 5x2cv experiments with each algorithm. The Filter method is always the fastest algorithm (denoted with **bold** type). The results in *italics* type correspond to the second fastest algorithm.

| Domain | Filter | FilterGA | sGA | SFS | SBE |
|---|---|---|---|---|---|
| Anneal | **0.28** | 44.2 | 66.4 | *26.1* | 190 |
| Arrhythmia | **4.37** | 926.0 | 1322.9 | *775* | 32497 |
| Euthyroid | **0.31** | 62.4 | 91.9 | *21.2* | 290.3 |
| Ionosphere | **0.12** | *9.9* | 12.8 | 10.4 | 22.1 |
| Pima | **0.03** | 2.1 | 2.8 | *0.9* | 2.3 |
| Random21 | **0.46** | *44.8* | 80.6 | 71.9 | 119.6 |
| Redundant21 | **0.45** | *44.0* | 54.6 | 67.1 | 148.6 |
| Segmentation | **0.64** | 77.3 | 65.5 | *31.6* | 138.6 |
| Soybean Large | **1.81** | *94.5* | 99.7 | 137.2 | 293.4 |

The number of examined subsets can be used as a coarse surrogate for the execution time, but the actual times depend on the number of features present in each candidate subset and may vary considerably from what we might expect. The execution times (user time in CPU seconds) for the entire 5x2cv experiments are reported in table 5. For the filter method, the time reported includes the time to compute and sort class separabilities and the time to evaluate the naive Bayes on the feature subset found by the filter method. The proposed filter method is by far the fastest algorithm, beating its closest competitor by two orders of magnitude. However, the filter found significantly less accurate results for four of the nine datasets. Among the wrapper methods, SFS and the hybrid of the filter and the GA are the fastest.

Figure 5 summarizes the tradeoff between accuracy (table 2) and execution time (table 5) for six of the data sets. The other data sets were omitted to reduce clutter. The graph clearly shows that the filter is two orders of magnitude faster than the other methods, but the wrappers usually result in accuracy improvements.

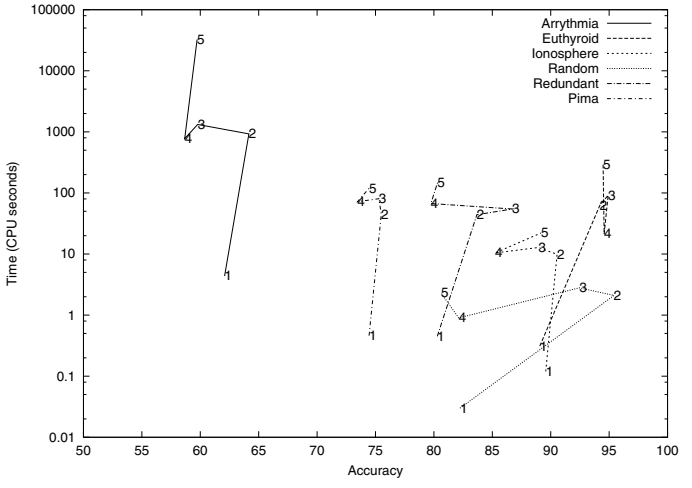


**Fig. 2.** Plot of accuracies vs. execution time for five data sets. The algorithms are identified by labels as follows: (1) Filter, (2) FilterGA, (3) sGA, (4) SFS, (5) SBE.

## 6 Conclusions

This paper presented experiments with a proposed GA-Filter hybrid for feature selection in classification problems. The results were compared against a simple GA, two traditional sequential methods, and a filter method based on a simple class separability metric. The experiments considered a Naive Bayes classifier and public-domain and artificial data sets. In the data sets we tried, the proposed method always found the most accurate solutions or solutions that were not significantly different from the best. The proposed method usually found the second smallest feature subsets (behind SFS) and performed faster than simple GAs, SFS, and SBE methods.

This work can be extended with experiments with other evolutionary algorithms, classification methods, additional data sets, and alternative class distance metrics. In particular, it would be interesting to explore methods that consider more than one feature at a time to calculate class separabilities.

There are numerous opportunities 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. Note that SFS and SBE are inherently serial methods and cannot benefit from parallelism as much as GAs. In addition, future work should explore efficient methods to deal with the noisy accuracy estimates, instead of using the relatively expensive multiple crossvalidations that we employed.

**Acknowledgments.** I would like to thank Martin Pelikan and Chandrika Kamath for useful discussions on this topic.

UCRL-CONF-202041. 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

1. John, G., Kohavi, R., Phleger, K.: Irrelevant features and the feature subset problem. In: Proceedings of the 11th International Conference on Machine Learning, Morgan Kaufmann (1994) 121–129
2. Kohavi, R., John, G.: Wrappers for feature subset selection. Artificial Intelligence **97** (1997) 273–324
3. Jain, A., Zongker, D.: Feature selection: evaluation, application and small sample performance. IEEE Transactions on Pattern Analysis and Machine Intelligence **19** (1997) 153–158
4. Siedlecki, W., Sklansky, J.: A note on genetic algorithms for large-scale feature selection. Pattern Recognition Letters **10** (1989) 335–347
5. Brill, F.Z., Brown, D.E., Martin, W.N.: Genetic algorithms for feature selection for counterpropagation networks. Tech. Rep. No. IPC-TR-90-004, University of Virginia, Institute of Parallel Computation, Charlottesville (1990)
6. Brotherton, T.W., Simpson, P.K.: Dynamic feature set training of neural nets for classification. In McDonnell, J.R., Reynolds, R.G., Fogel, D.B., eds.: Evolutionary Programming IV, Cambridge, MA, MIT Press (1995) 83–94
7. Bala, J., De Jong, K., Huang, J., Vafaie, H., Wechsler, H.: Using learning to facilitate the evolution of features for recognizing visual concepts. Evolutionary Computation **4** (1996) 297–311
8. Kelly, J.D., Davis, L.: 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, San Mateo, CA, Morgan Kaufmann (1991) 377–383
9. Punch, W.F., Goodman, E.D., Pei, M., Chia-Shun, L., Hovland, P., Enbody, R.: Further research on feature selection and classification using genetic algorithms. In Forrest, S., ed.: Proceedings of the Fifth International Conference on Genetic Algorithms, San Mateo, CA, Morgan Kaufmann (1993) 557–564
10. Raymer, M.L., Punch, W.F., Goodman, E.D., Sanschagrin, P.C., Kuhn, L.A.: Simultaneous feature scaling and selection using a genetic algorithm. In Bäck, T., ed.: Proceedings of the Seventh International Conference on Genetic Algorithms, San Francisco, Morgan Kaufmann (1997) 561–567
11. Kudo, M., Sklansky, K.: Comparison of algorithms that select features for pattern classifiers. Pattern Recognition **33** (2000) 25–41
12. Vafaie, H., De Jong, K.A.: Robust feature selection algorithms. In: Proceedings of the International Conference on Tools with Artificial Intelligence, IEEE Computer Society Press (1993) 356–364
13. Inza, I., Larrañaga, P., Etxeberria, R., Sierra, B.: Feature subset selection by Bayesian networks based optimization. Artificial Intelligence **123** (1999) 157–184
14. Cantú-Paz, E.: Feature subset selection by estimation of distribution algorithms. In Langdon, W.B., Cantú-Paz, E., Mathias, K., Roy, R., Davis, D., Poli, R., Balakrishnan, K., Honavar, V., Rudolph, G., Wegener, J., Bull, L., Potter, M.A., Schultz, A.C., Miller, J.F., Burke, E., Jonoska, N., eds.: GECCO 2002: Proceedings of the Genetic and Evolutionary Computation Conference, San Francisco, CA, Morgan Kaufmann Publishers (2002) 303–310

15. Raymer, M.L., Punch, W.F., Goodman, E.D., Kuhn, L.A., Jain, A.K.: Dimensionality reduction using genetic algorithms. IEEE Transactions on Evolutionary Computation **4** (2000) 164–171
16. Inza, I., Larrañaga, P., Sierra, B.: Feature subset selection by Bayesian networks: a comparison with genetic and sequential algorithms. International Journal of Approximate Reasoning **27** (2001) 143–164
17. Inza, I., Larrañaga, P., Sierra, B.: Feature subset selection by estimation of distribution algorithms. In Larrañaga, P., Lozano, J.A., eds.: Estimation of Distribution Algorithms: A new tool for Evolutionary Computation. Kluwer Academic Publishers (2001)
18. Ozdemir, M., Embrechts, M.J., Arciniegas, F., Breneman, C.M., Lockwood, L., Bennett, K.P.: Feature selection for in-silico drug design using genetic algorithms and neural networks. In: IEEE Mountain Workshop on Soft Computing in Industrial Applications, IEEE Press (2001) 53–57
19. Lanzi, P.: Fast feature selection with genetic algorithms: a wrapper approach. In: IEEE International Conference on Evolutionary Computation, IEEE Press (1997) 537–540
20. Guyon, I., Elisseeff, A.: An introduction to variable and feature selection. Journal of Machine Learning Research **3** (2003) 1157–1182
21. Oh, I.S., Lee, J.S., Suen, C.: Analysis of class separation and combination of class-dependent features for handwritting recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence **21** (1999) 1089–1094
22. Harik, G., Cantú-Paz, E., Goldberg, D.E., Miller, B.L.: The gambler's ruin problem, genetic algorithms, and the sizing of populations. Evolutionary Computation **7** (1999) 231–253
23. Matsumoto, M., Nishimura, T.: Mersenne twister: A 623-dimensionally equidistributed uniform pseudorandom number generator. ACM Transactions on Modeling and Computer Simulation **8** (1998) 3–30
24. Blake, C., Merz, C.: UCI repository of machine learning databases (1998)
25. Miller, B.L., Goldberg, D.E.: Genetic algorithms, selection schemes, and the varying effects of noise. Evolutionary Computation **4** (1996) 113–131
26. Alpaydin, E.: Combined $5 \times 2$cv F test for comparing supervised classification algorithms. Neural Computation **11** (1999) 1885–1892