Feature Selection in Unsupervised Learning via Evolutionary Search

YongSeog Kim, W. Nick Street, and Filippo Menczer
Management Sciences Department
University of Iowa
Iowa City, IA 52242 USA
{yong-s-kim, nick-street, filippo-menczer}@uiowa.edu

Abstract

Feature subset selection is an important problem in knowledge discovery, not only for the insight gained from determining relevant modeling variables but also for the improved understandability, scalability, and possibly, accuracy of the resulting models. Feature selection has traditionally been studied in supervised learning situations, with some estimate of accuracy used to evaluate candidate subsets. However, we often cannot apply supervised learning for lack of a training signal. For these cases, we propose a new feature selection approach based on clustering. A number of heuristic criteria can be used to estimate the quality of clusters built on the basis of a given feature subset. Rather than attempting to combine such criteria, we use ELSA, an evolutionary local selection algorithm capable of maintaining a diverse population of solutions that approximate the Pareto front in a multi-dimensional objective space. Each evolved solution represents a feature subset and a number of clusters; a standard K-means algorithm is applied to form the given number of clusters based on the selected features. We evaluate the approach with two synthetic data sets, one with ad-hoc distributions of points in a low-dimensional space and one with random distributions in a high-dimensional space. We also apply the algorithm to real data. Our preliminary results show promise in finding Pareto-optimal solutions through which we can identify the significant features and the correct number of clusters.

Paper ID: 364

Contact author: W. Nick Street, Management Sciences Department, S232 Pappajohn Business Building, University of Iowa, Iowa City, IA 52242. Phone (319) 335-1016. Email nick-street@uiowa.edu.

1 Introduction

Feature selection and clustering are active research areas in pattern recognition, machine learning, and data mining. Feature selection is defined as the process of choosing a subset of the original predictive variables by eliminating redundant features and those with little or no predictive information. If we extract as much information as possible from a given data set while using the smallest number of features, we can not only save a great amount of computing time and cost, but often build a model that generalizes better to unseen points. Further, it is often the case that finding the correct subset of predictive features is an important problem in its own right.

We adopt the wrapper model [16] of feature selection which requires two components: a search algorithm that searches through the possible combinations of features, and one or more criterion functions that evaluate the quality of each feature subset. Let D represent the original feature dimension of a given data set. The whole search space is $O(2^D)$, making exhaustive search impractical for data sets with even moderate dimensionality.

Most research on search algorithms has used heuristic search approaches in favor of efficiency rather than optimality. For instance, algorithms such as sequential search [30, 19], branch and bound [26], nonlinear optimization [5], and simulated annealing [27] have been applied. The formulation of feature selection as a combinatorial optimization problem has also lead to the use of genetic algorithms [28, 31]. A recent review of these methods can be found in [8]. Regardless of the search algorithm employed, most previous methods evaluated potential solutions in terms of predictive accuracy. Specifically, the data set could be divided into training and test sets, with the error rate on the test set used to estimate the classifier's true error rate. However, in many situations we don't have information about the class to which each data point belongs, and thus we can not apply supervised learning to estimate subset quality.

Instead, we may wish to find natural grouping of the examples in the feature space. This problem of *clustering* or *unsupervised learning* is another active research area in the knowledge discovery community. The idea is to represent groups of points by a cluster center after determining the inherent number of clusters in the given data set. For example,

clustering can be used very effectively in target marketing. Based on customer behavior such as brand loyalty, price sensitivity, or quality sensitivity, manufacturers can use different marketing strategies. Furthermore, they can save time and expense by restricting their concern to a group of customers who are most likely to buy their goods. One difficulty in clustering is to determine the number of clusters. Once a number of clusters have been formed based on some given features, we must to evaluate how well this model represents the complexity of the data. Clustering may be performed using iterative methods such as K-means [17] or expectation maximization (EM) [12], probability models [7], or optimization models [6]. Recent research has focused on scaling methods like K-means to large data sets [4]. We take the view that an effective way to scale a clustering algorithm is to reduce the dimensionality of the data by using a subset of the points to select a subset of the features.

A number of heuristic criteria can be used to estimate the quality of the clusters. Examples include the compactness of each cluster and the separation among different clusters. Several attempts have been made to combine different heuristic quality measures into some single quantity to be optimized [9]. This is a difficult problem to solve in the general case, since any given data set may have unique characteristics, and any given decision maker will have their own mental model of the tradeoffs among criteria. In such situations we must use *multi-objective* or *Pareto* optimization. Formally, each solution s_i is associated with an evaluation vector $F = (F_1(s_i), \ldots, F_C(s_i))$ where C is the number of quality criteria. One solution s_1 is said to *dominate* another solution s_2 if $\forall c : F_c(s_1) \geq F_c(s_2)$ and $\exists c : F_c(s_1) > F_c(s_2)$, where F_c is the c-th criterion, $c \in \{1 \ldots C\}$. Neither solution dominates the other if $\exists c_1, c_2 : F_{c_1}(s_1) > F_{c_1}(s_2), F_{c_2}(s_2) > F_{c_2}(s_1)$. We define the *Pareto front* as the set of nondominated solutions. The goal is to approximate as best possible the Pareto front, presenting the decision maker with a set of high-quality solutions from which to choose.

To this point, feature selection and clustering have been studied separately. In this study, we solve the two problems simultaneously by proposing an unsupervised algorithm to select a subset of features. As a search algorithm, greedy methods such as sequential floating search are suitable for small- and medium-scale problems [21]. Since we are interested in large-scale problems, we turn to evolutionary algorithms (EAs) to intelligently search the space of possible feature subsets. An EA is a parallel and global search algorithm that works

with a population of solutions to simultaneously evaluate many points in the search space.

Ideally the population of an EA converges to the global optimum. However, standard EAs often converge prematurely to local optima. Furthermore, standard EAs assume a single fitness function to be optimized and thus cannot consider multiple fitness criteria effectively. A number of multi-objective extensions of evolutionary algorithms have been proposed in recent years [29]. Most of them employ computationally expensive selection mechanisms to favor dominating solutions and to maintain diversity, such as Pareto domination tournaments [15] and fitness sharing [13].

Since we wish to search the space in parallel without sacrificing efficiency, we use a new evolutionary algorithm that maintains diversity over multiple objectives by employing a *local* selection scheme. This Evolutionary Local Selection Algorithm (ELSA) works well for Pareto optimization problems [24]. In this framework, each individual solution is allocated to a local environment based on its criteria values and competes with others to consume shared resources only if they are located in the same environment. Eventually, its chance to reproduce is jointly affected by its quality and by the presence of similar solutions sharing its local environmental resources. The more densely populated the local environment, the more competition among individuals for resources, resulting in bias toward different local environments. In this application, the EA automatically maintains diversity among solutions by biasing its search toward uncovered combinations of features.

In order to evaluate the quality of a subset of features, we use the standard K-means algorithm [17] with each solution's selected subset of features. K-means requires the number of clusters, K, as input. In our approach we do not want to commit to some estimate of the number of clusters, nor to search exhaustively or greedily over possible values of K. Therefore we evolve K as part of the genotypic representation of each individual solution, along with the feature subset. If the selected features are sufficient to explain the data, we expect our clustering to be effective.

The remainder of the paper is organized as follows. In Section 2, we discuss our approach in detail, justifying our heuristic clustering quality metrics, illustrating our evolutionary algorithm, and describing how ELSA is combined with K-means. Section 3 presents some experiments with synthetic and real data sets, and discusses the interpretation of the ELSA

output to select a subset of good features. Finally, we conclude the paper by suggesting directions for future work.

2 Feature selection algorithm

2.1 Heuristic metrics for clustering

A number of numerical measurements are available to evaluate clustering quality [14, 9]. Most of them are based on geometric distance metrics and therefore they are not directly applicable because they are biased by the dimensionality of the space, which is variable in feature selection problems. In our study we use four heuristic fitness criteria, described below. Two of the criteria are inspired by statistical metrics and two by Occam's razor [3]. Each objective, after being normalized into the unit interval, is to be maximized by the EA.

F_{within} : This objective is meant to favor dense clusters by measuring cluster cohesiveness. It is inspired by the total within-cluster sum of squares (TWSS) measure. Formally, let $x_i, i = 1, \cdots, n$, be data points and x_{ij} be the value of the j-th feature of x_i. Let d be the dimension of the *selected* feature set, J, and K be the number of clusters. Now, define the cluster membership variables α_{ik} as follows:

$$\alpha_{ik} = \begin{cases} 1 & \text{if } x_i \text{ belongs to cluster } k \\ 0 & \text{otherwise} \end{cases}$$

where $k = 1, \cdots, K$ and $i = 1, \cdots, n$. The centroid of the k-th cluster, γ_k, can be defined by its coordinates:

$$\gamma_{kj} = \frac{\sum_{i=1}^{n} \alpha_{ik} x_{ij}}{\sum_{i=1}^{n} \alpha_{ik}}, \; j \in J.$$

F_{within} can finally be computed as follows:

$$F_{within} = 1 - \frac{1}{Z_{within}} \frac{1}{d} \sum_{k=1}^{K} \sum_{i=1}^{n} \alpha_{ik} \sum_{j \in J} (x_{ij} - \gamma_{kj})^2 \tag{1}$$

where the normalization by the number of selected features, d, is meant to compensate for the dependency of the distance metric on the dimensionality of the feature subspace. Z_{within} is a normalization constant meant to achieve F_{within} values spanning the unit interval. Its value is set empirically for each data set.

$F_{between}$: This objective is meant to favor well-separated clusters by measuring their distance from the global centroid. It is inspired by the total between-cluster sum of squares (TBSS) measure. We compute $F_{between}$ as follows:

$$F_{between} = \frac{1}{Z_{between}} \frac{1}{d} \frac{1}{k-1} \sum_{k=1}^{K} \sum_{i=1}^{n} (1 - \alpha_{ik}) \sum_{j \in J} (x_{ij} - \gamma_{kj})^2 \tag{2}$$

where, as for F_{within}, we normalize by the dimensionality of the selected feature subspace and by the empirically derived constant $Z_{between}$.

$F_{clusters}$: The purpose of this objective is to compensate for the previous metrics' bias towards increasing the number of clusters. For example, $F_{within} = 1$ in the extreme case when we have the same number of clusters as the number of data points, with each point allocated to its own cluster. Clearly such overfitting makes the model more complex and less generalizable than can be justified by the data. Therefore, other things being equal, we want fewer clusters:

$$F_{clusters} = 1 - \frac{K - K_{min}}{K_{max} - K_{min}} \tag{3}$$

where K_{max} (K_{min}) is the maximum (minimum) number of clusters that can be encoded into a candidate solution's representation.

$F_{complexity}$: The final objective is aimed at finding parsimonious solutions by minimizing the number of selected features:

$$F_{complexity} = 1 - \frac{d-1}{D-1}. \tag{4}$$

Note that at least one feature must be used. Other things being equal, we expect that lower complexity will lead to easier interpretability of solutions as well as better

```
initialize population of agents, each with energy θ/2
while there are alive agents and for T iterations
   for each energy source c
      for each v (0 .. 1)
         E^c_envt(v) ← 2vE^c_tot
      endfor
   endfor
   for each agent a
      a' ← mutate(clone(a))
      for each energy source c
         v ← Fitness(a', c)
         ΔE ← min(v, E^c_envt(v))
         E^c_envt(v) ← E^c_envt(v) − ΔE
         E_a ← E_a + ΔE
      endfor
      E_a ← E_a − E_cost
      if (E_a > θ)
         insert a' into population
         E_a' ← E_a/2
         E_a ← E_a − E_a'
      else if (E_a < 0)
         remove a from population
      endif
   endfor
endwhile
```

Figure 1: ELSA pseudo-code. In each iteration, the environment is replenished and then each alive agent executes the main loop. In sequential implementations, the main loop calls agents in random order to prevent spurious sampling effects. Extinction does not occur in the experiments described in this paper, so we stop the algorithm after T iterations. The values of this and other parameters are discussed in Section 3.

generalization.

2.2 Evolutionary local selection algorithm

ELSA springs from algorithms originally motivated by artificial life models of adaptive agents in ecological environments [23]. Modeling reproduction in evolving populations of realistic organisms requires that selection, like any other agent process, be locally mediated by the environment in which the agents are situated. An agent's fitness must result from individual interactions with the environment, which contains other agents as well as finite shared resources.

We now briefly describe the ELSA implementation for the feature selection problem discussed in this paper. A more extensive discussion of the algorithm and its application to Pareto optimization problems can be found elsewhere [24]. Figure 1 outlines the ELSA algorithm at a high level of abstraction.

Each agent (candidate solution) in the population is first initialized with some random

solution and an initial reservoir of *energy*. The representation of an agent consists of $D + K_{max} - 2$ bits. D bits correspond to the selected features (1 if a feature is selected, 0 otherwise). The remaining bits are a unary representation of the number of clusters.[1] The motivation for this representation over a binary one stems from the desire to preserve the regularity of the number of clusters under the mutation operator: mutating any one bit will change K by one. Mutation is the only genetic operator (no crossover operator is used in the experiments described here) and therefore it is the only means of exploring the search space.

In each iteration of the algorithm, an agent explores a candidate solution similar to itself. The agent collects ΔE from the environment and is taxed with E_{cost} for this action. The net energy intake of an agent is determined by its fitness. This is a function of how well the candidate solution performs with respect to the criteria being optimized. But the energy also depends on the state of the environment. The environment corresponds to the set of possible values for each of the criteria being optimized.[2] We imagine an energy source for each criterion, divided into bins corresponding to its values. So, for criterion fitness F_c and bin value v, the environment keeps track of the energy $E^c_{envt}(v)$ corresponding to the value $F_c = v$. Further, the environment keeps a count of the number of agents $P_c(v)$ having $F_c = v$. The energy corresponding to an action (alternative solution) a for criterion F_c is given by

$$Fitness(a, c) = \frac{F_c(a)}{P_c(F_c(a))}. \tag{5}$$

Candidate solutions receive energy only inasmuch as the environment has sufficient resources; if these are depleted, no benefits are available until the environmental resources are replenished. Thus an agent is rewarded with energy for its high fitness values, but also has an interest in finding unpopulated niches in objective space, where more energy is available. The result is a natural bias toward diverse solutions in the population. E_{cost} for any action is a constant ($E_{cost} < \theta$).

In the selection part of the algorithm, each agent compares its current energy level with

[1] The cases of zero or one cluster are meaningless in this application. Therefore we count the number of clusters as $k = \kappa + 2$ where κ is the number of ones and $2 \leq k \leq K_{max}$.

[2] Continuous objective functions are discretized.

```
assign each data point to a randomly chosen cluster
calculate the centroid γ_k of each cluster k
do
   for each point i
      move i to nearest cluster arg min_k distance(i, γ_k)
   endfor
   recalculate the centroids of clusters whose data sets have changed
while at least one point changed cluster assignment
```

Figure 2: K-means clustering algorithm.

a threshold θ. If its energy is higher than θ, the agent reproduces: the mutated clone that was just evaluated becomes part of the population, with half of its parent's energy. When an agent runs out of energy, it is killed. The population size is independent of the reproduction threshold; θ only affects the energy stored by the population at steady-state.

When the environment is replenished with energy, each criterion c is allocated an equal share of energy:

$$E_{tot}^{c} = \frac{p_{max} E_{cost}}{C} \tag{6}$$

where $C = 4$ criteria in this study. This energy is apportioned in linear proportion to the values of each fitness criterion, so as to bias the population toward more promising areas in objective space [11]. Note that the total replenishment energy that enters the system at each iteration is $p_{max} E_{cost}$, which is independent of the population size p but proportional to the parameter p_{max}. This way we can maintain p below p_{max} on average, because in each iteration the total energy that leaves the system, pE_{cost}, cannot be larger than the replenishment energy.

2.3 K-means algorithm

In order for ELSA to assign energy to a solution, it needs to evaluate the fitness criteria corresponding to the solution's feature subset and number of clusters. Therefore it must form the given number of clusters based on the selected features. In the experiments described here, the clusters to be evaluated are constructed using a standard K-means algorithm [17].

K-means is one of the most often used non-hierarchical clustering methods. It iteratively assigns each data point to the cluster whose centroid is located nearest to the given point, and recalculates the centroids based on the new set of assignments. Some variants of K-

means have been suggested in order to improve the efficiency of the algorithm, avoid initial seed value effects, or find the global optimum [20, 1]. However, in our study we use the standard K-means algorithm as summarized in Figure 2.

Each time a new candidate solution is evaluated, the corresponding bit string is parsed to get a feature subset J and a cluster number K. The K-means algorithm receives in input the projection of the data set onto J and uses it to form K clusters. The four fitness criteria F_{within}, $F_{between}$, $F_{clusters}$, and $F_{complexity}$ are then computed and returned to ELSA.

3 Evaluation

By definition it is hard to evaluate the quality of an unsupervised clustering algorithm. Feature selection problems present the added difficulties that the clusters depend on the dimensionality of the selected features and that any given feature subset may have its own clusters, which may well be incompatible with those formed based on different feature subsets.

For these reasons we take a gradual approach to evaluate the proposed approach. First, we use a small-dimensional synthetic data set, in which the points have been generated carefully with well-defined distributions and clusters along each feature dimension. This data set allows us to validate our algorithm by determining whether any given solution evolved by ELSA represents a sensible compromise between the conflicting heuristic quality objectives.

Second, we use a high-dimensional synthetic data set, in which the distributions of the points and the significant features are known, while the appropriate clusters in any given feature subspace are not known. This data set allows us to estimate the performance of the algorithm by observing which portions of the significant features are identified by the evolved solutions.

Finally, we use a real data set for which we have knowledge about the clusters and the relevant features. In this case, we can evaluate the solutions both by examining the selected features and by judging the semantics of the resulting clusters.

Another way to evaluate our approach is by comparison with an alternative algorithm. For this purpose we have implemented a greedy heuristic algorithm known as two-way se-

quential selection [18]. Our implementation of this algorithm for clustering requires a set value of K and uses F_{within} as the only optimization criterion. The algorithm begins by finding the single dimension along which the objective is optimized. This dimension constitutes the initial feature set. At each successive step, the algorithm adds the feature that, when combined with the current set, forms the best clusters. It then checks to see if the least significant feature in the current set can be eliminated to form a new set with superior performance. This iteration is continued until all the features have been added. For comparison purposes, we repeated the algorithm for the same values of K considered by ELSA.

3.1 Experiment 1

The first synthetic data set has $n = 300$ points and $D = 6$ features. It is constructed as follows. One cluster is formed along feature 1 and two clusters are formed randomly along feature 2. Therefore if we plot the data projected onto dimensions 1 and 2 we obtain two clearly separated clusters. Along feature 3, we randomly reassign the points to two independent clusters. We repeat the process for feature 4. Finally, for features 5 and 6, the points are distributed uniformly. All the clusters along each dimension are formed by generating points from a pseudo-Gaussian distribution obtained by averaging the coordinates of some number of uniformly distributed points.[3] Figure 3 illustrates this data set by projecting the points onto some of the feature subspaces with $d = 2$.

The motivation for this data set is to have an understanding of the relationships between the different features, and at the same time a realistic mixture of significant, less significant, and insignificant features. If we consider the subsets of dimensionality $d = 2$, feature 1 taken in conjunction with feature 2, or 3, or 4 creates two correlated clusters. However, if we pick any two of features 2, 3, or 4, the clusters in each dimensions are not correlated and thus there are four clusters. If we considered $d = 3$ and we picked features 2, 3, and 4, we would find $2^3 = 8$ clusters. The last two features are white noise and thus of no significance.

The individuals are represented by strings with 12 bits, 6 for the features and 6 for the number of clusters, so that $K_{max} = 8$. There are 7 energy bins for $F_{clusters}$, 6 for $F_{complexity}$,

[3] The standard deviation of these pseudo-Gaussian distributions is $\sigma \approx 0.06$.

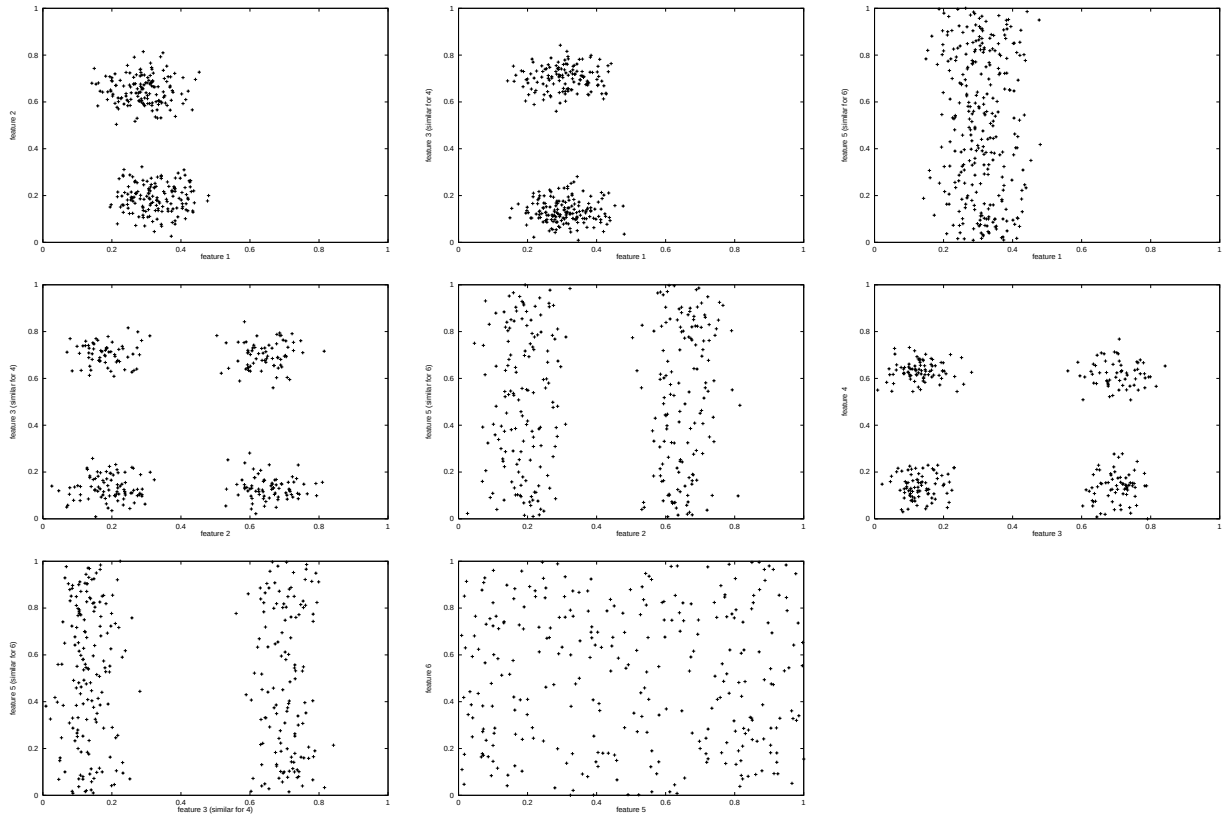

Figure 3: The data set of Experiment 1, plotted in some of the possible 2-dimensional subsets of the features space.

Parameter	Value
$\Pr(mutation)$	0.1
$\Pr(crossover)$	0
p_{max}	100
E_{cost}	0.2
θ	0.3
T	400

Table 1: ELSA parameters values. The probability of mutation refers to a per-bit mutation rate.

and 10 each for F_{within} and $F_{between}$. The values used for the various ELSA parameters are shown in Table 1.

The best solution with four clusters in more than one dimension included features 2 and 3. The best solution with $K = 2$ and more than one dimension included features 1 and 4. As depicted in Figure 3, both of these solutions describe the data very well. The final population was dominated by solutions with one feature, which typically look extremely good along two criteria: complexity, and either F_{within} (many centers inside one true cluster) or $F_{between}$ (well-separated centers along a random dimension).

As expected, the greedy search method performed very well on this simple data set. With $K = 2$, features were added in the order 1, 3, 2, 4, 5, 6; with $K = 4$, the order was 1, 3, 4, 2, 5, 6. As it happens, the two-dimensional clusters along features 1 and 3 are somewhat better (in terms of F_{within} than those along features 1 and 2.

3.2 Experiment 2

With the second data set we intend to test the algorithm on a problem with higher dimensionality, while retaining the "realistic" flavor of the smaller data set. In other words we have some "significant" features (in which points belong to correlated normal clusters), some "Gaussian noise" features (in which values are drawn from single or bimodal normal distributions along each dimension, but the distributions along different features are uncorrelated), and some "white noise" features (in which points are drawn from uniform distributions).

The data set has $n = 500$ points and $D = 30$ features. It is constructed so that the first 10 features are significant, with 5 "true" clusters consistent across these features. The next 10 features are Gaussian noise, with points randomly and independently assigned to 2 normal clusters along each of these dimensions. The remaining 10 features are white noise. The standard deviation of the normal distributions is $\sigma \approx 0.06$ and the means are themselves drawn from uniform distributions in the unit interval, so that the clusters may overlap — the actual number of clusters may be smaller than constructed, along each dimension.

Individuals are represented by 38 bits, 30 for the features and 8 for K ($K_{max} = 10$). There are 9 bins for $F_{clusters}$ and 10 each for $F_{complexity}$, F_{within}, and $F_{between}$. The parameters for ELSA are the same as those used in Experiment 1 (see Table 1), except that T = 8000

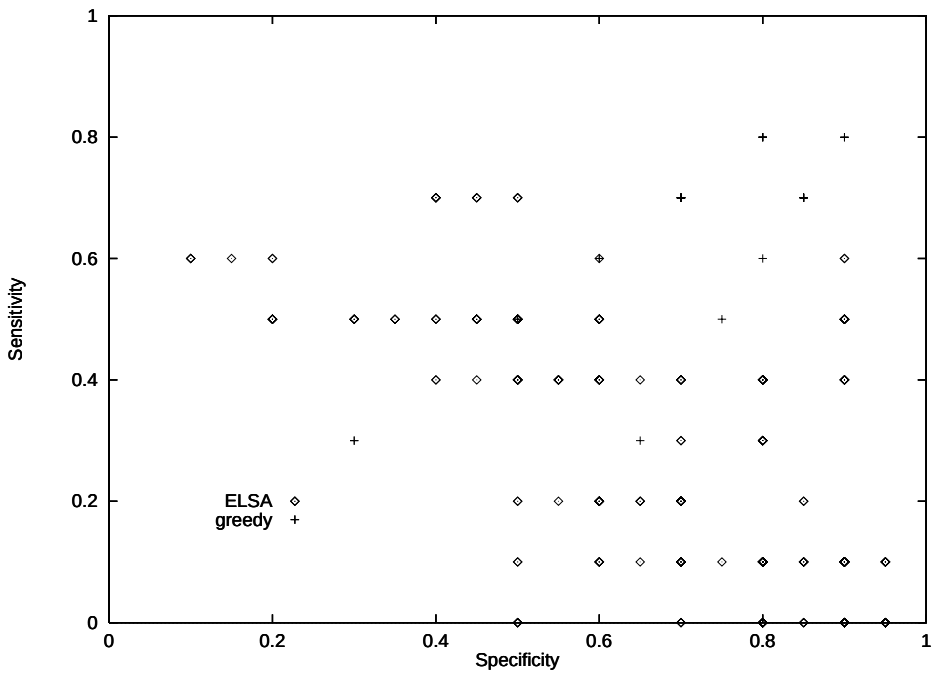

Figure 4: ROC curve showing sensitivity and specificity of the individuals in the final generation for data set 2, along with the solutions generated by the greedy algorithm.

iterations.

Another way to evaluate the performance of our algorithm is by looking at the *sensitivity* and *specificity* of each evolved solution. These measures are defined as follows:

$$Sensitivity = \frac{TP}{TP + FN} \tag{7}$$

$$Specificity = \frac{TN}{TN + FP} \tag{8}$$

where TP (true positive) is the number of selected significant features, FP (false positive) is the number of selected noise features, TN (true negative) is number of discarded noise features, and FN (false negative) is the number of discarded significant features. Ideally, both measurements would be close to 1, as only significant features would be selected. However, as we increase a solution's complexity (decrease $F_{complexity}$), more features are selected and as a result sensitivity goes up while specificity goes down. Therefore, if we plot sensitivity versus specificity (an ROC curve) for the solutions in the final population, we can estimate the algorithm's effectiveness at discriminating between significant and noise features.

Figure 4 shows such an ROC curve for our large synthetic data, along with the values

for the solutions generated by the greedy algorithm for $K = 2, \cdots, 10$ clusters. ELSA can successfully discard noise features, as shown by specificity values up to 95%. The range of the sensitivity among the evolved solutions is smaller, with a peak of 70%. This is consistent with the data set, since the clusters along the 10 significant dimensions are correlated and therefore not all those features are necessary. The trend displayed by the intermediate ELSA solutions attests to the trade-off between sensitivity and specificity in this difficult problem. The greedy algorithm finds a few solutions with both high sensitivity (up to 80%) and high specificity (up to 90%), but it has additional knowledge about the problem — it uses a complexity of 10 features, corresponding to the number of significant dimensions in the data.

3.3 Experiment 3

In addition to the artificial data sets discussed above, we also test our algorithm on a real data set, the Wisconsin Prognostic Breast Cancer (WPBC) data [22, 2]. This data set records 30 numeric features quantifying the nuclear grade of breast cancer patients at the University of Wisconsin Hospital. It also contains traditional prognostic variables tumor size and number of positive lymph nodes, along with a binary variable indicating whether lymph status was recorded. This results in a total of 33 features for each of 227 cases.

Individuals are represented by 37 bits, 33 for the features and 4 for K ($K_{max} = 6$), therefore there are 5 bins for $F_{clusters}$. Other ELSA parameters are the same as those used in Experiment 2 (see Table 1), except that T = 10,000.

We analyze clustering performance on this data set by looking for clinical relevance in the resulting clusters. Specifically, we can observe the actual outcome (time to recurrence, or known disease-free time) of the cases in the various clusters. Figure 5 shows a Kaplan-Meier estimate of the true disease-free survival times for patients in the clusters represented in one solution from our final population. This solution contained three clusters in 7 dimensions. It was chosen by picking the best individual (in terms of $F_{between}$ and F_{within}) with three clusters from the final population.

The figure clearly shows that the clustering solution found three groups with well-separated survival characteristics. The best prognostic group (represented by the top curve)

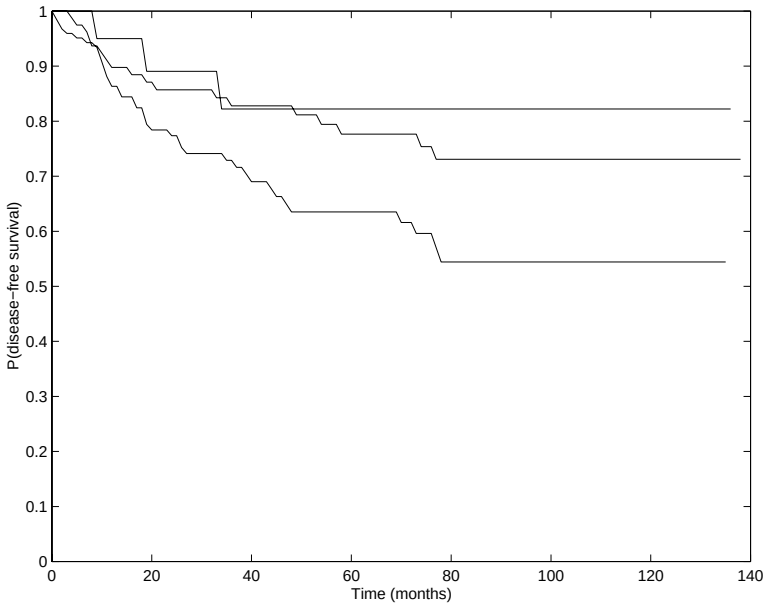

Figure 5: Estimated survival curves for the groups found by the ELSA-based clustering method. Choosing best solution with three clusters created groups corresponding to good, intermediate, and poor prognosis.

was relatively small, containing 22 cases, with only three recurrences. Because of its small size, it was not statistically significantly different from the intermediate group (p = .075). The intermediate group was well-differentiated from the poor group (p ¡ 0.01).

The chosen dimensions included a mix of nuclear morphometric features such as symmetry, concavity and texture, along with lymph status and tumor size. We note that the inclusion of lymph status requires dissection of the ancillary nodes for staging purposes, leaving the patient at risk for painful complications. While we would prefer to make treatment decisions without this feature, the clustering results consistently indicated that it was relevant to the forming of prognostic groups.

4 Conclusions

We presented a novel approach for large-scale feature selection problems using unsupervised learning. ELSA, an evolutionary local selection algorithm, was used successfully in previous work in conjunction with supervised learning [24, 25]. In this paper we used ELSA to search for possible combination of features and numbers of clusters, with the guidance of the

K-means algorithm. While the search biases of ELSA and K-means may not be ideal for this application, the combination of a multi-objective search algorithm with unsupervised learning provides a promising framework for feature selection. We summarize our findings as follows.

- ELSA covers a large space of possible feature combinations well while simultaneously optimizing the multiple criteria.
- The standard K-means algorithm can be used to guide ELSA by evaluating the quality of a subset of features.
- A number of possibly conflicting heuristic metrics can be plugged into the algorithm, while remaining agnostic about their relative worth or their relationships.
- Most importantly, in the proposed framework we can select significant feature subsets without training examples, while at the same time identifying the inherent numbers of clusters.

In future work we would like to compare the performance of ELSA on the unsupervised feature selection task with other multi-objective EAs [10], using each in conjunction with the standard K-means algorithm. We will also consider the use of different clustering algorithms that may be more appropriate in specific situations, such as problems with nominal features or clusters with different shapes.

Another interesting direction is the further analysis of the interactions among our various optimization criteria. For instance, increasing the number of features dramatically affects both of our cluster quality metrics. While we corrected for much of this effect with normalization terms, further study is needed to decorrelate the effects of the various criteria. Further, the $F_{between}$ measure does not necessarily correlate directly with one's intuition about cluster quality. Well-separated, but nearby, clusters are judged harshly by the traditional TBSS measure on which $F_{between}$ is based. We will explore other objectives that implement the idea of forming well-separated clusters.

Although in theory the best thing that an algorithm can do in multi-objective optimization is to approximate the Pareto front, it would be desirable from the standpoint of a

non-expert user to identify one single solution, out of the final population, representing a "best compromise." Once the algorithm has identified a manageable set of candidate solutions (the Pareto front approximation in the final population), we might be able to apply some more expensive statistical or geometric method. For example, we might look along the approximate Pareto front for a point of maximal curvature, by considering tangential hyperplanes in Pareto space.

From a knowledge discovery perspective, our algorithm offers several advantages. Certainly the simplicity bias of Occam's Razor is well-established as a means for improving generalization on real-world data sets. Further, it is often the case that the user can gain insight into the problem domain by finding the set of relevant features; consider, for example, the significant literature on prognostic factors in breast cancer, or the target marketing problem described in Section 1. Finally, a key problem in data mining is the scaling of predictive methods to large data sets. Our algorithm can easily be used as a preprocessing step to determine an appropriate set of features (and number of clusters), allowing the application of iterative algorithms like K-means on much larger problems.

Acknowledgments

We are grateful to Melania Degeratu for developing the original ELSA software and for helpful discussions. This work was supported in part by NSF grant IIS-99-96044.

References

[1] G.P. Babu and M.N. Murty. A near-optimal initial seed value selection in k-means algorithm using a genetic algorithm. *Pattern Recognition Letters*, 14(10):763–769, 1993.

[2] C. L. Blake and C. J. Merz. UCI repository of machine learning databases [http://www.ics.uci.edu/~mlearn/MLRepository.html], 1998. University of California, Irvine, Department of Information and Computer Sciences.

[3] A. Blumer, A. Ehrenfeucht, D. Haussler, and M. K. Warmuth. Occam's razor. *Information Processing Letters*, 24:377–380, 1987.

[4] P. S. Bradley, U. M. Fayyad, and C. Reina. Scaling clustering algorithms to large databases. In R. Agrawal, P. Stolorz, and G. Piatetsky-Shapiro, editors, *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining*, pages 9–15, Menlo Park, CA, 1998. AAAI Press.

[5] P. S. Bradley, O. L. Mangasarian, and W. N. Street. Feature selection via mathematical programming. *INFORMS Journal on Computing*, 10(2):209–217, 1998.

[6] P.S. Bradley, O.L. Mangasarian, and W.N. Street. Clustering via concave minimization. In M. I. Jordan M. C. Mozer and T. Petsche, editors, *Advances in Neural Information Processing Systems 9*, pages 368–374, MA: Cambridge, 1997. MIT Press.

[7] P. Cheeseman, J. Kelly, M. Self, J. Stutz, W. Taylor, and D. Freeman. Autoclass: A bayesian classification system. In *Proceedings of the Fifth International Conference on Machine Learning*, pages 54–64, San Francisco, CA, 1988. Morgan Kaufmann.

[8] M. Dash and H. Liu. Feature selectin for classification. *Intelligent Data Analysis*, 1(3):131–156, 1997.

[9] D.L. Davies and D.W. Bouldin. A cluster separation measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-1(2):224–227, 1979.

[10] K. Deb and J. Horn. Special issue on multi-criterion optimization. *Evolutionary Computation Journal*, 8(2), 2000.

[11] M. Degeratu, 2000. Personal communication.

[12] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38, 1977.

[13] D. E. Goldberg and J. Richardson. Genetic algorithms with sharing for multimodal function optimization. In *Proc. 2nd International Conference on Genetic Algorithms*, 1987.

[14] J.A. Hartigan. *Clustering Algorithms*. Wiley, New York, 1975.

[15] J Horn. Multicriteria decision making and evolutionary computation. In *Handbook of Evolutionary Computation*. Institute of Physics Publishing, London, 1997.

[16] G. H. John, R. Kohavi, and K. Pfleger. Irrelevant features and the subset selection problem. In *Proceedings of the 11th International Conference on Machine Learning*, San Mateo, CA, 1994. Morgan Kaufmann.

[17] R.A. Johnson and D.W. Wichern. *Applied Multivariate Statistical Analysis*. Prentice Hall, New Jersey, 3 edition, 1992.

[18] J. Kittler. Feature selection and extraction. In Y. Fu, editor, *Handbook of Pattern Recognition and Image Processing*, New York, 1978. Academic Press.

[19] J. Kittler. Feature set search algorithms. In C.H. Chen, editor, *Pattern Recognition and Signal Processing*, pages 41–60, Sijthoff and Noordhoff, Alphen aan den Rijn, The Netherlands, 1978.

[20] K. Krishna and M.N. Murty. Genetic k-means algorithm. *IEEE Transactions on Systems, Man, and Cybernetics - PartB: Cybernetics*, 29(3):433–439, 1999.

[21] M. Kudo and J. Sklansky. Comparison of algorithms that select features for pattern classifiers. *Pattern Recognition*, 33:25–41, 2000.

[22] O. L. Mangasarian, W. N. Street, and W. H. Wolberg. Breast cancer diagnosis and prognosis via linear programming. *Operations Research*, 43(4):570–577, July-August 1995.

[23] F. Menczer and R. K. Belew. Latent energy environments. In R. K. Belew and M. Mitchell, editors, *Adaptive Individuals in Evolving Populations: Models and Algorithms*. Addison Wesley, Reading, MA, 1996.

[24] F. Menczer, M. Degeratu, and W.N. Street. Efficient and scalable pareto optimization by evolutionary local selection algorithms. *Evolutionary Computation*, 8(2), 2000.

[25] F. Menczer, W.N. Street, and M. Degeratu. Evolving heterogeneous neural agents by local selection. In V. Honavar, M. Patel, and K. Balakrishnan, editors, *Advances in the Evolutionary Synthesis of Neural Systems*. MIT Press, Cambridge, MA, 2000.

[26] P.M. Narendra and K. Fukunaga. A branch and bound algorithm for feature subset selection. *IEEE Transactions on Computers*, 26(9):917–922, 1977.

[27] W. Siedlecki and J. Sklansky. On automatic feature selection. *International Journal of Pattern Recognition and Artificial Intelligence*, 2(2):197–220, 1988.

[28] W. Siedlecki and J. Sklansky. A note on genetic algorithms for large-scale feature selection. *Pattern Recognition Letters*, 10:335–347, 1989.

[29] D. A. Van Veldhuizen. *Multiobjective Evolutionary Algorithms: Classifications, Analyses, and New Innovations*. PhD thesis, Air Force Institute of Technology, 1999.

[30] A.W. Whitney. A direct method of nonparametric measurement selection. *IEEE Transactions on Computers*, 20:1100–1103, 1971.

[31] J. Yang and V. Honavar. Feature subset selection using a genetic algorithm. In H. Motada and H. Liu, editors, *Feature Extraction, Construction, and Subset Selection: A Data Mining Perspective*. Kluwer, New York, 1998.