FEATURE SELECTION IN SUPERVISED AND UNSUPERVISED LEARNING VIA EVOLUTIONARY SEARCH

by

YongSeog Kim

An Abstract

Of a thesis submitted in partial fulfillment
of the requirements for the Doctor of
Philosophy degree in Business Administration
in the Graduate College of
The University of Iowa

December 2001

Thesis supervisor: Assistant Professor W. Nick Street

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.

The purpose of this study is to provide a comprehensive analysis of feature selection via evolutionary search in supervised and unsupervised learning. To achieve this purpose, it first discusses a general framework for feature selection based on a new search algorithm, Evolutionary Local Selection Algorithm (ELSA). ELSA searches for a good set of feature subset in a multi-dimensional objective space and can be combined with any supervised and unsupervised learning algorithms.

In supervised learning, we train an induction algorithm to maximize the classification accuracy for unseen data while minimizing the size of feature subsets. Using Artificial Neural Networks (ANNs) as an induction algorithm, we apply an ELSA/ANN model to a real business problem, customer targeting, and address the issues related with knowledge discovery.

Focusing on feature selection for creating diversity in a classification ensemble, we provide a new two-level evolutionary algorithm, Meta-Evolutionary Ensemble (MEE) to create an optimal ensemble. In this algorithm, the various ensembles com-

pete with one another, being judged on their estimated predictive performance. In addition, the underlying classifiers also compete with each other, being rewarded for correctly predicting the training examples. In this way we aim to optimize ensembles rather than form ensembles of individually-optimized classifiers.

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 and discuss its significance to the real-world problems in marketing and finance. Further, we show that our new approach can be used to form the given number of clusters based on the selected features.

Abstract approved: _____
Thesis supervisor

_____
Title and department

_____
Date

FEATURE SELECTION IN SUPERVISED AND UNSUPERVISED LEARNING VIA EVOLUTIONARY SEARCH

by

YongSeog Kim

A thesis submitted in partial fulfillment
of the requirements for the Doctor of
Philosophy degree in Business Administration
in the Graduate College of
The University of Iowa

December 2001

Thesis supervisor: Assistant Professor W. Nick Street

Graduate College
The University of Iowa
Iowa City, Iowa

CERTIFICATE OF APPROVAL

PH.D. THESIS

This is to certify that the Ph.D. thesis of

YongSeog Kim

has been approved by the Examining Committee for the thesis requirement for the Doctor of Philosophy degree in Business Administration at the December 2001 graduation.

Thesis committee: _____
Thesis supervisor

_____
Member

_____
Member

_____
Member

_____
Member

To my family, especially to my parents

ACKNOWLEDGEMENTS

I owe many thanks to great professors and friends I have known in the University of Iowa.

My first thanks go to the members of my thesis committee. In particular, I would like to thank my thesis adviser, Dr. W. Nick Street, for his encouragement, appropriate professional guide, and financial support. I have enjoyed very much discussing research topics with him. He has greatly expanded my understanding to various topics on machine learning and data mining. He has even refined my English and programming skills.

Dr. Filippo Menczer introduced an evolutionary algorithm, ELSA that soon became the fundamental part of my research. His careful reviews and comments were always helpful. Dr. Alberto M. Segre has maintained CASTOR Laboratories free of major problems so that I could finish this thesis successfully. He also enlightened the possible new research direction, feature selection in molecular biology. Dr. Padmini Srinivasan introduced a different kind of feature selection problem, feature selection in information retrieval. Her critical comments refined and broadened my knowledge. The insightful comments and careful reviews from Dr. David Eichmann greatly improved the quality of this thesis.

In addition to committee members, many others have helped me. Dr. Warren

Boe was a great academic adviser during the first two years in my Ph.D. program and always greeted me at his office. Even after I decided to work with Dr. Nick Street, he helped me deal with many administrative problems based on his rich experience. I also would like to thank my friends at the CASTOR Laboratories: Karl Ardnt for his personal mentorship on American culture and insightful suggestions; Qiaomin Han, Jiawei Zhang, and Gautam Pant for their friendships and rich discussions.

Finally, I would like to thank my family members who have so much suffered through this long journey. Their true love and moral support has been the greatest source of energy that I need to complete thesis successfully. My special thanks and love go to my wife, MeeYeon Ahn, and two lovely sons, David and Daniel. I love you all.

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.

The purpose of this study is to provide a comprehensive analysis of feature selection via evolutionary search in supervised and unsupervised learning. To achieve this purpose, it first discusses a general framework for feature selection based on a new search algorithm, Evolutionary Local Selection Algorithm (ELSA). ELSA searches for a good set of feature subset in a multi-dimensional objective space and can be combined with any supervised and unsupervised learning algorithms.

In supervised learning, we train an induction algorithm to maximize the classification accuracy for unseen data while minimizing the size of feature subsets. Using Artificial Neural Networks (ANNs) as an induction algorithm, we apply an ELSA/ANN model to a real business problem, customer targeting, and address the issues related with knowledge discovery.

Focusing on feature selection for creating diversity in a classification ensemble, we provide a new two-level evolutionary algorithm, Meta-Evolutionary Ensemble (MEE) to create an optimal ensemble. In this algorithm, the various ensembles com-

pete with one another, being judged on their estimated predictive performance. In addition, the underlying classifiers also compete with each other, being rewarded for correctly predicting the training examples. In this way we aim to optimize ensembles rather than form ensembles of individually-optimized classifiers.

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 and discuss its significance to the real-world problems in marketing and finance. Further, we show that our new approach can be used to form the given number of clusters based on the selected features.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

This chapter discusses research motivation for feature selection and reviews the fundamental concepts related to it. First, we present research motivation for feature selection in supervised and unsupervised learning in Section 1.1. In Section 1.2, we describe two representative algorithms for learning, decision trees and neural networks, and clustering algorithms such as K-means and Expectation Maximization (EM) for unsupervised learning. Section 1.3 provides the overall structure of thesis.

1.1 Feature selection in general

Feature selection has been an active research area in pattern recognition [162, 97, 123], machine learning [9, 115, 108, 31], statistics [62, 59, 143, 152], and data mining communities [99, 114, 113, 106, 107]. The main idea of feature selection is to choose a subset of the original 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 for future analyses, but often build a model that generalizes better to unseen points. By identifying insignificant information, feature selection also can save cost occurred data collection, communication, and maintenance. Feature selection may also significantly improve the comprehensibility

of the resulting classifier models and clarify the relationships among features. Further, it is often the case that finding the correct subset of predictive features is an important problem in its own right. For example, a physician may make a decision based on the selected features whether a potentially dangerous surgery is necessary for treatment or not.

One of the simplest algorithm for feature selection, *backward sequential selection* (BSS) was introduced in [137]. This starts with the full set of features, and greedily removes the one that most improves performance, or degrades performance slightly. Another simple algorithm, *forward sequential selection* (FSS) [208], starts with the empty set of features, and greedily adds features. However, both algorithms cannot consider the interaction among features and thus produce suboptimal solutions [198].

Feature selection is also performed by classifiers in the decision tree family, such as ID3 (1983), CART (1984), and C4.5 (1993) . These algorithms include a feature selection routine as a subroutine and heuristically search the space of feature subsets along the tree structures. Other approaches to feature selection can be done either without using any learning algorithms – the filter approach – or with learning algorithms – the wrapper approach – to evaluate selected features [99]. In the filter approach, the inherent properties of features such as the correlation with the target function are regarded as important and learning algorithms are used only for evaluating final classifier with the selected features. However, the wrapper approach uses

a feature selection algorithm as a wrapper around the learning algorithms. Thus the feature selection algorithm searches for a good subset using the learning algorithms to evaluate feature subsets. The same learning algorithm is used for evaluating the final classifier on holdout data to get the estimated accuracy.

We adopt the wrapper model of feature selection mainly because the wrapper model has been shown to return higher predictive accuracy in supervised learning [99, 44]. The wrapper approach typically 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 $\Theta(2^D)$, making exhaustive search impractical for data sets with even moderate dimensionality. As a search algorithm for feature space, greedy methods such as sequential floating search are suitable for small- and medium-scale problems [123]. Since we are interested in large-scale problems, we use Evolutionary Algorithms (EAs) to intelligently search the space of possible feature subsets.

In order to select a good feature subset, we evaluate each feature subset in terms of multiple objectives: the number of selected features (which should be minimized) along with the accuracy in supervised learning (which should be maximized) or the clustering quality in unsupervised learning (which should be maximized). 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 can 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.

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 [202]. Most of them employ computationally expensive selection mechanisms to favor dominating solutions and to maintain diversity, such as Pareto domination tournaments [90] and fitness sharing [77]. We propose a new algorithm, Evolutionary Local Selection Algorithms (ELSA), which works well for Pareto optimization problems [141, 106]. In ELSA, 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. The more densely populated the local environment, the more competition among individuals for resources, resulting in bias toward different local environments.

Feature selection can have different implications in supervised and unsupervised learning. In supervised learning, an induction algorithm is given a set of training examples and is required to learn generalization rules to fit the training examples. Each example in the training set guides or supervises a learner with a list of feature values and a corresponding class label.

Feature selection in supervised learning aims to find a feature subset that produces a higher classification accuracy given an induction algorithm. This is obtained by eliminating noisy features that make it difficult for a learning algorithm to separate useful signal from noise. Further, feature selection can eliminate *deceptive* features that interact with the learning algorithm's bias counter-productively and deteriorate overall performance on unseen data. For example, consider a customer targeting problem whose main goal is to identify potential customers who are most likely to buy new products or services. An induction algorithm combined with a feature selection routine searches the space of feature subsets in order to maximize predictive accuracy while minimizing the number of features used. Our feature selection model in supervised learning, ELSA/ANNs, uses ELSA to search through the possible combinations of features and artificial neural networks (ANNs) to evaluate the quality of each feature subset in terms of classification accuracy. Estimated predictive accuracy is used to guide the search direction of ELSA.

In *clustering* or *unsupervised learning*, each example in a given data set consists of an only unlabeled instance, a list of feature values. Thus, learning algorithms

are designed to find inherent structures in the data, finding natural grouping of the examples in the feature space. The idea is to represent groups of points by a cluster prototype after determining the inherent number of clusters in the given data set. Clustering is especially useful when there is no or little prior information about data or when it is necessary to minimize assumptions about the data. Clustering may be performed using iterative methods such as K-means [100] or Expectation Maximization (EM) [57, 28], probability models [47], or optimization models [30]. Recently a set of novel clustering algorithms have been proposed in the database community [214, 81]. For instance, Agrawal *et al.* [1] present an order-independent clustering algorithm, CLIQUE, that forms clusters in large data sets.

In this thesis, the standard K-means algorithm [63] and the Expection Maximization (EM) algorithm [57] are used to do clustering. These algorithms group unlabeled patterns into K clusters in such a way that patterns within the same cluster are more similar to each other than other patterns in different clusters. Clustering procedures consist of four steps: pattern representation, clustering, data abstraction and cluster validity analysis [96]. Once the number of clusters K and proximity measurements are defined in pattern representation, actual clustering is performed. For further analysis, cluster prototypes or representative patterns are extracted to describe each cluster. Final assessment of a clustering output is done using heuristic criteria such as the compactness of each cluster and the separation among different clusters.

Thus feature selection in unsupervised learning aims to find a good subset of features that forms high quality of clusters for a given number of clusters. This is harder problem because the interdependency between the clustering quality and the number of clusters. Some researchers [1, 194] have studied feature selection and clustering together. In particular, Devaney and Ram [58] combined a sequential forward and backward search algorithm with two concept learning algorithms, COBWEB [70] and AICC, an improved variant of COBWEB. The category utility score was employed as a quality measurement of feature subsets and the number of clusters was one of the factors used for the computation of utility score. In [201], a Bayesian framework with a unified objective function considering both the number of clusters and the feature subset was applied to the problem of document clustering. Recently, Dy and Brodley [65] proposed a wrapper approach that uses an EM algorithm to form clusters. Feature subsets are evaluated in terms of clustering quality based on either scatter separability or maximum likelihood.

1.2 Machine learning algorithms

In this section, we first review two representative machine learning algorithms – decision tree and neural networks – that have been successfully adopted for feature selection in supervised learning. We then review the clustering algorithms for use in feature selection in unsupervised learning.

1.2.1 Decision trees

After the original work by [93], decision tree induction has become one of the most popular algorithms in the machine learning community. Decision trees such as ID3 [164], CART [37], Assistant 86 [45], and C4.5 [166] have been successfully used for various classification systems. In particular, these algorithms enable human experts to understand the models more easily by representing the learned pattern as a decision tree or a corresponding set of decision rules. In this section, we limit our focus to C4.5 exclusively because it is one of the most frequently used tree algorithms.

Typically, a decision tree is represented by nodes and edges. At each node, each feature is evaluated to determine how well it classifies the associated data points and the best feature is selected. Note that some decision tree algorithms choose a subset or all features at each node. Various measurements such as information gain [164, 144], gain ratio [166], gini index [37], and chi square [83] are used for evaluating features. An empirical comparison among different metrics can be found in [144], which claimed that random feature selection with pruning can achieve comparable predictive accuracy to more elegant feature selection methods. However, a comparative study on the same data sets in [40] refuted this claim and suggested possible reasons for different conclusions from two experiments. Another comparative study also supported the claim that decision trees constructed with elegant feature selection showed higher predictive accuracy than those constructed with random feature selection [134].

Once the best feature is determined, edges descending from the current node are constructed for each of the possible values (or intervals of continuous features) of the selected feature. The training data is mutually exclusively partitioned along those edges. Thus each intermediate node has its own training points and determines the best feature using only the associated points. The process of selecting a new best feature and partitioning the training points is repeated until it reaches a terminal node in which all points belong to the same class or when all features are evaluated. Decision rules can be obtained by following each path from the root node to a terminal node.

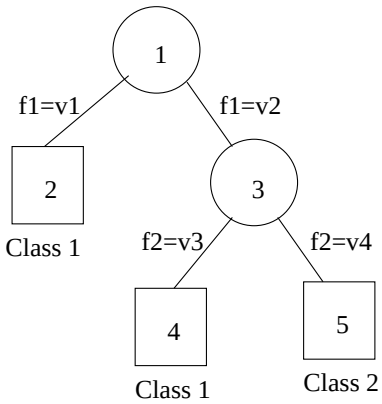

Figure 1.1: The structure of a binary decision tree.

A simple example is shown in Figure 1.1. Node splitting starts from the root node, node 1. After all features (f_1 and f_2) are evaluated, f_1 is selected as the best feature, which has only two possible values, v_1 and v_2 in this case. All points whose

value of f_1 is v_1 take a path to node 2 which will be classified as class 1. However, other points lead to node 3 which will be further split into node 4 and node 5 based on the value of f_2. Based on the class distribution at node 4 and node 5, tree induction algorithms determine them as class 1 and class 2 respectively. The corresponding sets of If-then rules are: If $f_1 = v_1$ or ($f_1 = v_2$ and $f_2 = v_3$), then class 1. If $f_1 = v_2$ and $f_2 = v_4$, then class 2.

Features can appear at most once along any path through the decision tree constructed by C4.5. All paths are mutually exclusive, meaning that any new example will always satisfy one and only one path. For feature selection purposes, decision tree algorithms can be used either solely, or can be combined with search algorithms as in the wrapper approach. When a decision tree algorithm is used solely for feature selection, the set of selected features at intermediate nodes are returned as a solution. However, in the wrapper approach, a new decision tree is built using only selected features by a search algorithm and its predictive accuracy is returned to a search algorithm to guide the search direction. Practical issues in learning decision trees include

- determining how deeply to grow the decision tree,
- discretizing continuous features,
- choosing an appropriate measure to evaluate and select attributes,
- handling training data with missing attribute values,

- handling attributes with different costs,
- and improving computational efficiency.

Decision tree algorithms can avoid overfitting either by stopping growing the tree before it reaches the point where it perfectly classifies the training data, or by allowing the tree to overfit the data and then post-pruning the tree. Compared to neural networks algorithms, decision tree algorithms are faster and have fewer parameters with comparable classification accuracy. Further, graphical denotations and corresponding sets of If-then rules are in particular easier to understand than outputs from neural networks. More comparative experiments with decision tree algorithms and other classification algorithms can be found in [61, 165, 184].

However, the solutions from decision tree algorithms might not be optimal in the sense that the decision about the best feature at the current node is not backtracked. Once the current node is split based on the selected feature, that particular decision is not reconsidered later. Further, it is not clear how to determine the appropriate size of and number of intervals for splitting continuous features, though this problem is well-addressed in [167]. It is also known that there could be many different trees that are consistent with given training data.

1.2.2 Neural networks

The development of the artificial neural networks (ANNs) [172, 177] has been inspired in part by the fact that the most advanced learning system, human brains,

consists of millions of interconnected neurons. Contrasted with decision tree algorithms, ANNs can approximate well both real-valued and discrete-valued target functions. In particular, ANNs have proven successful in many practical problems. For example, ANNs has been applied in management sciences, finance, and marketing for stock market prediction [176, 159], bankruptcy prediction [209], customer clustering [74, 5] and market segmentation [92, 14] with some success.

It was proved in [54] that any function can be approximated to arbitrary accuracy by a network with three layers of units when the output layer uses linear units, and the hidden layer uses sigmoid units. The backpropagation algorithm [175] is the most common network learning algorithm. We briefly review its structure and learning process. Typically, the neural network model consists of a number of neurons (nodes) which are connected by weighted links. We show a representative model with three layers, an input layer, a hidden layer, and an output layer in Figure 1.2.

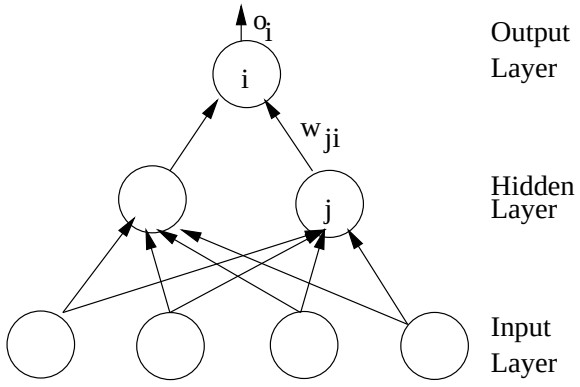

Figure 1.2: The structure of a neural network model. It consists of three layers; input layer, hidden layer, and output layer. There is only one output node for two-class concept learning.

Each node in the input layer receives a corresponding feature from the input data, and passes it via weighted connections to the units in the hidden layer. Each unit i computes its activation level o_i using an activation function, typically the sigmoid logistic function

$$o_i = f(net_i) = \frac{1}{1 + e^{-net_i}} \tag{1.1}$$

where net_i is defined as follows:

$$net_i = \sum_{j \in pred(i)} o_j w_{ji}. \tag{1.2}$$

In the above equation, $pred(i)$ and w_{ji} denotes the set of predecessors of unit i and the connection weight from unit j to unit i respectively.

Learning in neural networks is done by adjusting network weights in order to map input to output through examples in the training data set, $x_n, n = 1, \cdots, N$. Each example x_n consists of feature values, $\overline{x}_n$, and its corresponding class label t_n. When an example with $\overline{x}_n$ is presented to the network, the distance between the target t_n and the actual output vector o_n is measured as follows:

$$E = \frac{1}{2} \sum_{n \in N} (t_d - o_d)^2. \tag{1.3}$$

Fulfilling the learning goal now is equivalent to finding a minimum of E. The weights in the network are changed along a search direction $\delta(t)$, driving the weights in the direction of the estimated minimum:

$$w(t+1) = w(t) + \eta * \delta(t) \tag{1.4}$$

where the learning rate η determines the step size of weight changes and the negative gradient $-\frac{\partial E}{\partial w}$ is used for the search direction $\delta(t)$.

By propagating the error back from the output layer towards the input layer and applying the chain rule repeatedly, the Backpropagation algorithm computes $\frac{\partial E}{\partial w_{ji}}$ for each weight in the network as follows:

$$\begin{aligned} \frac{\partial E}{\partial w_{ji}} &= \frac{\partial E}{\partial o_i}\frac{\partial o_i}{w_{ji}} \\ &= \frac{\partial E}{\partial o_i}\frac{\partial o_i}{\partial net_i}\frac{\partial net_i}{\partial w_{ji}} \\ &= \frac{\partial E}{\partial o_i} f'(net_i) o_j. \end{aligned} \tag{1.5}$$

Based on whether i is an output unit or not, the value of $\frac{\partial E}{\partial o_i}$ is computed as follows:

- Case 1: i is an output unit:

$$\frac{\partial E}{\partial o_i} = \frac{1}{2}\frac{\partial (t_i - o_i)^2}{\partial o_i} = -(t_i - o_i). \tag{1.6}$$

- Case 2: i is not an output unit:

$$\begin{aligned} \frac{\partial E}{\partial o_i} &= \sum_{k \in upper(i)} \frac{\partial E}{\partial o_k}\frac{\partial o_k}{\partial o_i} \\ &= \sum_{k \in upper(i)} \frac{\partial E}{\partial o_k}\frac{\partial o_k}{\partial net_k}\frac{\partial net_k}{\partial o_i} \\ &= \sum_{k \in upper(i)} \frac{\partial E}{\partial o_k} f'(net_k) w_{ik} \end{aligned} \tag{1.7}$$

where $upper(i)$ denotes the set of all units k in upper layers and the gradient information is passed down from the output layer to input layer successively. Once the gradient information is known, the weight update is computed as follows:

$$\triangle w_{ji}(t) = -\eta * \frac{\partial E}{\partial w_{ji}}(t). \tag{1.8}$$

Neural networks with backpropagation learning have great representational power and can be very effective for both discrete-valued and real-valued data that are often noisy. However, it is more difficult for humans to understand the resulting models from neural networks than learned rules from decision tree algorithms. Further, the performance of ANNs also depends on many parameters such as the number of training epochs, the activation functions at the hidden and output layers, learning rate, and number of hidden nodes. Longer training times than decision tree algorithms is another factor that hinders the usage of ANNs for high-dimensional data sets. For different neural networks algorithms, refer to [149] for recurrent networks that were proposed for the analysis of time series data and [129] for optimal brain damage approach that dynamically alters network structure.

1.2.3 Clustering algorithms

Clustering has many synonyms such as *unsupervised learning* [95], *numerical taxonomy* [189], and *vector quantization* [154] and makes it feasible to uncover multidimensional structures and interrelationship among the data points. In clustering, no labeled data are provided and the problem is to group unlabeled patterns into meaningful clusters in such a way that patterns within the same cluster are more similar to each other than other patterns in different clusters. Clustering is especially useful when there is no or little prior information about data or when it is necessary to minimize assumptions about the data as much as possible.

Hierarchical clustering algorithms [100] proceed either by successively merg-

ing different clusters or by successively splitting a cluster into two clusters. The first approach, which is called an agglomerative approach, begins with the individual patterns. Thus there are initially as many clusters as patterns. It successively combines from a pair of the most similar clusters together until a certain similarity criterion is satisfied. Eventually, as the similarity decreases, all subgroups are fused into one cluster.[1] On the contrary, a partitional (divisive) method begins with all patterns in a single cluster and performs splitting until a stopping criterion is met or each pattern forms a group.

Nonhierarchical clustering algorithms are designed to group items into a collection of K clusters, which can either be specified in advance or determined as part of the clustering procedure. Nonhierarchical methods can be applied to much larger data sets than hierarchical methods because a matrix of inter-cluster distances (similarities) does not have to be determined. Nonhierarchical methods start from either an initial partition of items into groups or an initial set of seed points, which form the centroid or medoid[2] of clusters.

The K-means algorithm is the simplest and most commonly used nonhierarchical algorithm. It employs a squared error criterion and implicitly assumes that clusters are represented by spherical Gaussian distributions located at the K cluster means. Starting with a random initial partition, it iteratively assigns each data point

[1]By setting the similarity criterion at different levels, we could have different clusterings of the same data.

[2]A medoid is the most centrally located point in a cluster.

to the cluster whose centroid is located nearest to the given point, and recalculates the centroids based on the new set of assignments until a convergence criterion is met.

The K-means algorithm is very efficient in terms of time complexity and it generates the same partition of the data irrespective of the order in which the patterns are presented. However, it is sensitive to the selection of the initial partition and may converge to a local minimum of the criterion function. Another problem is that it does not generalize well to the domains of discrete or categorical data because of the distance-based metric. Further, K-means is a *hard* clustering method and thus does not allow for uncertainty in the membership of patterns.

Unlike K-means, the mixture model, using Expectation Maximization (EM) algorithm [57] for optimization, generalizes to include realistic situations in which every data point belongs to *all* clusters with different membership probabilities. The underlying assumption is that the patterns to be clustered are drawn from one of several distributions, and the goal is to identify the parameters of each distribution and their number. The EM procedure begins with an initial estimate of the parameter vector and iteratively rescores the patterns using updated parameter estimates. In a clustering context, the pattern scores measure the likelihood of being drawn from particular components density function and thus give insights into the membership of each pattern to all classes.

The EM algorithm has been shown to be superior to other alternatives [48, 76, 81, 20]. In experiments [138], the EM significantly outperforms a "winner take

all" version of the EM and hierarchical agglomerative clustering. DBSCAN [68] is a relatively new mode seeking algorithm. DBSCAN can find clusters with arbitrary shaped clusters using two input parameters, the radius of the neighborhood of a point and the minimum number of points in the neighborhood [81]. The performance of DBSCAN, however, is sensitive to parameters and it can not handle data with many features [1].

Other clustering algorithms took an evolutionary approach [169, 12], deterministic approach [119], or stochastic approach [182, 6]. When data sets are too big to be stored in main memory, incremental clustering algorithms [84, 43, 70, 41] can be used. These algorithms assign the first data item to a cluster and assign the new item either to one of the existing clusters or to a new cluster based on the distance between the new item and the existing cluster centroids. Though they are very efficient in terms of time and space complexity, they are order dependent [96].

Recently, CLARANS [153], Focused CLARANS [67], and BIRCH [214] [81] have been proposed for large data sets. However, those algorithms are not suitable when clusters have different sizes or when clusters are non-spherical [81]. A new algorithm, CURE, was proposed to handle huge data sets with non-spherical shapes of clusters [81]. CURE first draws an appropriate number of samples from the whole database based on Chernoff bounds and partially clusters them. Clusters with the closest pair of representative points are merged into one cluster at each step. The performance of CURE, however, depends on data representation order and selected

samples.

A novel clustering algorithm CLIQUE [1] generates minimal cluster descriptions in the form of DNF expressions without pre-assumptions of data distribution. CLIQUE is different from other algorithms for large high dimensional data such as BIRCH, CLARANS, and DBSCAN in the sense that it discovers clusters in subspaces of the original data space. CLIQUE produces identical clustering results irrespective of presented order of data records and has good scalability as the number of features is increased.

1.3 Organization

The remainder of the thesis is organized as follows. In Chapter 2, we review the definition of feature selection and motivation of feature selection along with the relationship between relevant feature subsets and optimal feature subsets. We also divide feature selection algorithms into three main categories: embedded, filter, and wrapper approaches, and discuss their advantages and disadvantages. Various search algorithms for feature selection from simple greedy sequential algorithms to recent evolutionary algorithms are also discussed. Finally, our search algorithm, Evolutionary Local Selection Algorithm (ELSA), is proposed and explained in detail.

Chapter 3 presents an approach to feature selection using ELSA in supervised learning. Our algorithm is applied to a real world database marketing problem to identify and profile prospects who are most likely to respond to a direct marketing campaign. After multiple objectives are proposed and justified, the ELSA/ANN

model is explained in detail. Finally, the resulting model is evaluated and interpreted.

In Chapter 4, we propose a two-level evolutionary environment, Meta-Evolutionary Ensembles (MEE), that uses feature selection as the mechanism for boosting diversity and prediction accuracy of an individual classifier in an ensemble. In MEE, multiple ensembles are considered simultaneously and each component classifier is allowed to move into the best-fit ensemble. Genetic operators change the ensemble membership of the individual classifiers, allowing the size and membership of the ensembles to change over time.

Chapter 5 discusses the feature selection problem in unsupervised learning. Since prior class information is not available in unsupervised learning, we utilize the natural grouping based on the combination of a particular set of features and given the number of clusters. This is a harder problem because the inter-dependency between the number of clusters and the evaluation criteria of clustering results.

Finally Chapter 6 addresses directions of future research and concludes the thesis.

CHAPTER 2
FEATURE SUBSET SELECTION

This chapter first reviews the definition of feature selection. It next presents the motivation of feature selection and reviews the definition of feature relevance. Finally, it outlines the evolutionary local selection algorithm that will be used as our search algorithm to search the feature subset space.

2.1 Definition

2.1.1 Feature selection in supervised learning

Feature subset selection commonly involves finding a "good" set of features under some objective functions. As pointed out in [99], predictive accuracy, structure size, and the minimal use of input features are often used as objective functions. These objectives are consistent with notions such as Occam's razor [24], minimum description length [173], or minimum message length [205]. The importance of each objective is solely determined based on prior knowledge of problem domain. For example, when features are associated with medical tests or surgeries that are expensive or potentially hazardous, the minimal use of input features becomes the most important objective.

Formally, feature selection is defined as the process of selecting a subset of features which performs *best* with the underlying learning algorithms on a data set

with only the selected features in [97]. This definition captures most of the important aspect of feature selection and puts more weight on the predictive performance of the selected features together with the induction system. On the contrary, a somewhat different definition was proposed in [99] that puts more weight on the inherent properties of the feature sets. They defined feature selection as the process of selecting all *strongly* relevant features and a useful subset of the *weakly* relevant features that yield *good* performance.[1] Only irrelevant features should not be included in the chosen solution.

The appropriateness of these two different definitions seems to be dependent on the problem domain. If our main goal is to attain the highest predictive accuracy as often is desired in classification, the definition in [97] can be more intuitive. However, if we want to discover unknown patterns by observing each of the selected features in the resulting model, the latter [99] can be more relevant. Weighting the two main goals equally, we define feature selection in supervised learning as in Definition 1.

Definition 1 (Feature selection in supervised learning) *Feature selection in supervised learning is the process of choosing a subset of the original features that optimizes the predictive performance of a considered model by eliminating redundant features and those with little or no predictive information.*

The relationship between feature selection and an learning algorithm's performance is dependent on whether or not the monotonicity assumption holds. The

[1]We will review the definition of strong and weak relevance in Section 2.3.

monotonicity assumption implies that the addition of a feature always provide additional discriminating power, improving the performance of learning algorithms such as a Bayes classifier. Thus it is not recommended to restrict a Bayes classifier to a subset of features [113]. When features interact, selecting and combining features can find a better model even for naive Bayes [161].

However, the monotonicity assumption rarely holds because additional features can interfere with other more useful features and deteriorate the performance of learning algorithms [115]. Often additional noisy features can magnify noise in training examples and make it difficult for a learning algorithm to separate useful signal from noise. The more noisy features available, the less the learning algorithm will be able to identify predictive features with the given number of training points. Further, when *deceptive* features interact with the learning algorithm's bias counter-productively, these features deteriorate overall performance on unseen data. Therefore, feature selection becomes an important issue from a practical point of view.

Often feature extraction or construction methods such as Principal Component Analysis (PCA) [100] are confused with feature selection because they are also used to reduce problem dimensionality. Feature extraction, however, refers to more general methods that create new features based on transformations or combinations of the original feature set [97]. While the feature selection process might involve a trial-and-error process where various feature subsets are selected and evaluated, PCA does not

require training using labeled data [96]. This distinction between feature selection and feature extraction is also illustrated in [111, 97]. In this thesis, we intentionally restrict our interests to only feature subset selection, the selection of feature subsets from the *existing* collection of features.

2.1.2 Feature selection in unsupervised learning

When we do not have class information with which to evaluate different subsets of features, we instead wish to find natural grouping of the examples in the feature space via *clustering* or *unsupervised learning* and utilize the clustering results to evaluate solutions. The idea is to represent groups of points by a representative point or prototype after determining the inherent number of clusters in the given data set. Once the clusters have been formed based on some given features, we must evaluate how well this model represents the complexity of the data. Based on this intuition, we define feature selection in unsupervised learning as follows.

Definition 2 (Feature selection in unsupervised learning) *Feature selection in unsupervised learning is the process of choosing a subset of the original variables that forms a high quality clustering for the given number of clusters.*

Clustering may be performed using methods such as K-means [63], expectation maximization (EM) [57], or optimization models [30]. Recently a set of novel clustering algorithms have been proposed in the database community [214, 81]. For instance, Agrawal *et al.* [1] present an order-independent clustering algorithm, CLIQUE, that forms clusters in large data sets.

Feature selection in unsupervised learning can be considered as a sub-problem of unsupervised model selection. The problem of determining an appropriate model in unsupervised learning has gained popularity in the machine learning, pattern recognition, and data mining communities. Unsupervised model selection addresses either how to identify the optimal number of clusters K or how to select feature subsets while determining the correct number of clusters. The latter problem is more difficult because of the inter-dependency between the number of clusters and the feature subsets used to form the clusters [188]. To this point, most research on unsupervised model selection has considered the problem of identifying the right number of clusters using all available features [118, 188].

Other researchers [1, 194] have studied feature selection and clustering together. In particular, Devaney and Ram [58] combined a sequential forward and backward search algorithm with two concept learning algorithms, COBWEB [70] and AICC, an improved variant of COBWEB. The category utility score was employed as a quality measurement of feature subsets and the number of clusters was one of the factors used for the computation of utility score. In [201], a Bayesian framework with a unified objective function considering both the number of clusters and the feature subset was applied to the problem of document clustering. Recently, Dy and Brodley [65] proposed a wrapper approach that uses an EM algorithm to form clusters. Feature subsets are evaluated in terms of clustering quality based on either scatter separability or maximum likelihood.

2.2 Research motivation

In Section 2.1, we saw that feature selection becomes important in practical learning scenarios because the monotonicity assumption rarely holds in the real world. In this section, we review some other motivations for feature selection in supervised and unsupervised learning.

2.2.1 Improved comprehensibility and generalization

Feature selection can significantly improve the comprehensibility of the resulting classifier models, generated rules, or relationships among features. For example, top-down induction of decision trees, using such methods as CART [37], ID3 [164], or C4.5 [166], on high-dimensional data sets can result in a very complex tree structure. This makes it almost impossible for human inspectors to draw meaningful insights and utilize them to solve real problems. Even if the resulting relationships represent true patterns in data, they may be considered useless if they are too complex to comprehend. If a learning algorithm extracts most of information from a given data set with the smallest number of features, the resulting structure will be simpler. This simplified structure allows for better understanding on problem domains and the simplified relationships among a smaller number of features can be more easily inspected by human experts.

Feature selection can make learning algorithms generalize better on unseen data by eliminating irrelevant features and/or correlated and redundant features. Kira and Rendell [108] experimented with the effects of irrelevant features on ac-

curacy in Boolean target concept learning by showing that introduced redundant features degraded the performance of learning algorithms not only in speed but also in predictive accuracy. It also has been shown that CART, ID3, C4.5, and nearest neighbor algorithms suffer in the presence of irrelevant features. Naive Bayes classifiers [125, 63], which assume independence of features given the instance label, also degrade when using correlated and redundant features. Without correlated, redundant and/or irrelevant features, learning algorithms could estimate predictive performance more accurately from training data and generalize better on unseen data through finite sample size effects [97].

2.2.2 Reduced costs and computational time

Feature selection can reduce the costs occurred in data collection and maintenance dramatically by identifying whether features in the collected data are important or not. Due to the information technology explosion of the last few decades, the largest corporate databases are easily measured in terabytes. Those collected data represent a wealth of information that can be used to improve business decision-making in various areas. However, collecting and maintaining such huge databases is not free at all. It needs special equipment (e.g., scanners and bar-code readers) with software to collect data. Once data are collected, they are sent to a central warehouse over communication channels, incurring communication costs. Finally, database systems (including backup systems) and human experts are required to keep and maintain this high volume of data. If we can reduce data dimensionality through feature se-

lection, costs can be reduced. For example, if the data are transmitted over a long distance communication channel, it is cost efficient to extract the important features at the sender and transmit only the discriminatory information, reducing the required communication-channel capacity and time [111].

Feature selection is also very useful for reducing the computational time of learning algorithms. In practical learning scenarios, the learning algorithms are supposed to estimate the underlying patterns in data. In domains with a large number of features, these patterns are very complex and of high dimension. This situation becomes worse with the growing number of records. The time and space requirements for a learning algorithm often grow dramatically with the number of features, rendering the algorithm impractical for problems with a large number of features and records. For example, it is an well-known fact that a multilayer neural network requires a significant processing time to learn data patterns. According to [102, 23], training a three-node neural network in which the nodes use linear threshold function is a NP-hard problem. Similarly, it is noticed in [94] that finding the optimal binary decision tree is a NP-hard problem. If we could reduce the original high feature dimension into a manageable dimension, learning algorithms could return outputs within a reasonable time.

2.2.3 Feature selection as an independent objective

Identifying and selecting a predictive subset of input variables itself has its own significance as shown in [163, 193]. Punch et el. (1993) [163] provided an example

in biology. Suppose that biologists took three sets of 100 soil samples from near the crop roots (rhizosphere), away from the influence of the crop roots (non-rhizosphere), and from a crop residue. A Biolog test is a common method to verify whether the lack of diversity in the rhizosphere comes from the symbiotic relationship between the roots and their neighbor microbes. Typically, Biolog consists of a plate of 96 wells that contain a different substrate such as sugars, amino acids and other nutrients. Different substrates respond to different sets of microbes. Once each sample was tested on the 96 features, biologists could learn which of the Biolog features has the most discriminative power to explain the difference among three different sets of soil samples.

In [193], the result of feature selection can be used to determine whether an expensive test or a potentially dangerous surgery is necessary for breast cancer prognosis. Traditionally, the extent to which cancer is present in the lymph nodes has been used as the strongest available prognostic indicator. This information is determined by microscopic examination of lymph nodes only after they are surgically removed from the patient's armpit. This surgical procedure, however, not only increases medical costs but also causes side effects such as infection and lymphoedema, a severe swelling of the arm. By reducing a set of 32 features to five nuclear features, they found that the traditional medical prognostic factors of tumor size and lymph node status are not necessary for prognostic purpose. If those findings are confirmed, the potentially hazardous surgical removal of lymph nodes from patients could be

avoided.

2.3 Feature relevance

Through feature selection, we want to select relevant and predictive features, while eliminating irrelevant or redundant features. There are several questions to be asked, however. What are the criteria to discriminate relevant features from irrelevant ones? Are relevant features the same as predictive features? Could these definitions be applied without considering learning algorithms or should they be defined in terms of learning algorithms? In this section, we propose answers to these questions by mainly borrowing the contents from [113].

2.3.1 Review of existing definitions

One of the simplest definitions of relevance is proposed in [9] and summarized in Definition 1.

Feature relevance 1 *When all features including class label are Boolean and there are no noisy features, a feature X_i is relevant to the class label Y if X_i appears in every representation of Y.*

Though this definition is simple to understand, Definition 1 is not useful because of its strong assumptions. In order to overcome these restrictions, Gennari et al. [75] proposed a new definition as in Definition 2.

Feature relevance 2 *A feature X_i is relevant iff there exists some x_i and y for which*

$p(X_i = x_i) > 0$ *such that*

$$p(Y = y|X_i = x_i) \neq p(Y = y).$$

In Definition 2, X_i could be noisy and is not necessarily restricted to the Boolean domain. The conditional dependency of class label Y on X_i plays a critical role to determine the relevance of a feature X_i. As a result, the value of a relevant feature X_i changes systematically with Y [75]. Definition 2, however, can fail to identify relevant features because when all unlabeled possible instances are equiprobable as in parity learning, it does not consider other features X_j to compute the conditional dependency of Y on X_i.

Considering the conditional dependency of Y on X_i with given values of all other features, Kohavi and John [113] proposed the following somewhat general definition.

Feature relevance 3 *A feature X_i is relevant iff there exists some x_i, y, and s_i for which $p(X_i = x_i) > 0$ such that*

$$p(Y = y, S_i = s_i|X_i = x_i) \neq p(Y = y, S_i = s_i)$$

where $S_i = \{X_1, \cdots, X_{i-1}, X_{i+1}, \cdots, X_D\}$ and s_i is a value-assignment of all features in S_i.

Under the Definition 3, feature X_i is considered relevant only if, given the values of the other features, the removal of X_i changes the probability of the class label Y.

None of the definitions of relevance so far consider the learning algorithms being used. Other relevance definitions have been proposed to correct this problem in [99, 113]. They assume a Bayes classifier as the learning algorithm and propose two degrees of relevance: weak and strong relevance.

Feature relevance 4 (Strong relevance) *A feature X_i is strongly relevant iff there exists some x_i, y, and s_i for which $p(X_i = x_i, S_i = s_i) > 0$ such that*

$$p(Y = y|X_i = x_i, S_i = s_i) \neq p(Y = y|S_i = s_i).$$

In other words, a feature X_i is strongly relevant if two examples α and β have the same assignment to all features except to X_i and $c(\alpha) \neq c(\beta)$ where $c(\alpha)$ and $c(\beta)$ represents the classes of α and β, respectively [22]. Removing *strongly* relevant features increases ambiguity to the sample and decreases predictive accuracy. *Weakly* relevant features do not always contribute to predictive accuracy but they do sometimes. Formally, weak relevance is defined by [113] as in Definition 5.

Feature relevance 5 (Weak relevance) *A feature X_i is weakly relevant iff it is not strongly relevant, and there exists a subset of features S'_i of S_i for which there exists some x_i, y, and s'_i with $p(X_i = x_i, S'_i = s'_i) > 0$ such that*

$$p(Y = y|X_i = x_i, S'_i = s'_i) \neq p(Y = y|S'_i = s'_i).$$

A weakly relevant feature X_i becomes strongly relevant after removing a subset of the features [22]. Based on the Definition 4 and Definition 5, features are relevant if

they are either strongly or weakly relevant, but other features can never contribute to prediction accuracy.

From a different perspective, Blum and Langley [22] paid attention to *usefulness* [44] because a feature which contains information was not necessarily useful for prediction.[2] They formalized *incremental usefulness* in the following definition.

Feature relevance 6 (Incremental usefulness) *Given a learning algorithm A and a feature set F, a feature X_i is incrementally useful to A with respect to F if $X_i \cup F$ returns higher predictive accuracy than F does.*

Now, we compare these different definitions through a simple example.[3]

Example 1 (Correlated XOR) *There are five Boolean features and X_2 and X_3 are negations of X_4 and X_5, respectively. There are only eight possible instances, and we assume they are equiprobable. The target concept is given as follows:*

$$Y = X_1 \oplus X_2 \qquad (\oplus \textit{ denotes } XOR).$$

According to Definition 1, X_3 and X_5 are irrelevant because neither appears in the target concept formula or its equivalent expression $Y = X_1 \oplus \overline{X_4}$. Both X_2 and X_4 are also irrelevant because each feature appears only in *one* of two target expressions. Note that each of them is replaced by the negation of the other. By Definition 2,

[2]For example, social security number of customer is a highly relevant feature in terms of the contained information but useless for predicting whether she will buy a new product or not.

[3]We borrow this example from Kohavi & John (1997), page 4.

all features are irrelevant because the conditional probability of output value y given any feature value x is same as probability of y, $1/2$. By Definition 3, every feature is relevant. For example, X_1 is relevant because if we are given $s_1 = \{1, 1, 0, 0\}$ [4] and $y = 0$, and if we know $x_1 = 1$,

$$p(Y = y | X_i = x_1, S_1 = s_1) = \frac{1}{8} \neq p(Y = y, S_1 = s_1) = \frac{1}{4}$$

This happens because when we are given an s_i, knowing x_i changes the probability of four of the eight possible instances from $1/8$ to zero. By Definition 4, X_3 and X_5 are clearly irrelevant, and both X_2 and X_4 are irrelevant because they do not add any information to S_4 and S_2, respectively. X_2 and X_4 are weakly relevant by Definition 5[5]. Finally, we can easily check that X_1, X_2, X_4 are incrementally useful by Definition 6. We summarize our findings in Table 2.1.

Table 2.1: Summary of feature relevance.

Definition	Relevant	Irrelevant
Feature relevance 1	X_1	X_2, X_3, X_4, X_5
Feature relevance 2	None	All
Feature relevance 3	All	None
Feature relevance 4	X_1	X_2, X_3, X_4, X_5
Feature relevance 5	X_1, X_2, X_4	X_3, X_5
Feature relevance 6	X_1, X_2, X_4	X_3, X_5

[4] As defined in Definition 3, $S_i = \{X_1, \cdots, X_{i-1}, X_{i+1}, \cdots, X_D\}$ and s_i is a value-assignment of all features in S_i.

[5] X_1 is strongly relevant.

2.3.2 Optimal feature subset

According to Definition 4 and Definition 5, a feature is *relevant* if it is strongly or weakly relevant. A Bayes classifier therefore must use all strongly relevant features and possibly some weakly relevant features because it is based on the monotonicity assumption. Many learning algorithms, however, do not have access to the underlying distribution and sometimes work better for finite samples by ignoring relevant features [114]. It means that a relevant feature might not be useful or predictive in an learning algorithm (or vice versa). This notion leads to the definition of *optimal features* [113].

Definition 3 (Optimal feature subsets) *Given an learning algorithm and a dataset, an optimal feature set, S^*, is a subset of features such that the accuracy of the induced classifier is highest.*

It is possible that there is more than one optimal feature subset. For example, if $\rho(X_i, X_j) = 1$ where $i \neq j$, two different sets of features have same accuracy [113]. Now we show that relevant features do not imply the optimal features and optimal features do not imply relevant features through following two examples.[6]

Example 2 (Relevance $\not\Rightarrow$ optimality) *Assume three Boolean features, X_1, X_2, X_3 and uniform distribution of instances. If the target concept is $c(X_1, X_2, X_3) = (X_1 \wedge X_2) \vee X_3$, all features are relevant. If the hypothesis space, however, is the space of monomials, i.e., conjunctions of literals, the only optimal feature subset is X_3. The*

[6]We borrow them from Kohavi & John (1997, page 6.).

accuracy of the monomial X_3 is 87.5%, and adding another feature to the monomial will decrease the accuracy.

Example 3 (Optimality $\not\Rightarrow$ relevance) *If a feature X_i is always set to 1, X_i is irrelevant. Consider a limited Perceptron classifier that classifies new instances by comparing the linear combination of weights of input nodes with a threshold. When the threshold $= 0$, it has the same representation power as the regular Perceptron because of X_i. Removal of all irrelevant feature, however, would remove X_i too.*

From now on, we focus on finding optimal features rather than relevant features. This is consistent with our approach in the sense that we try to maximize the accuracy with respect to the specific learning algorithm and training set at hand. Since different learning algorithms have different biases, optimal features will be different over different algorithms. Different sizes of training data may also result in different sets of optimal features.[7]

2.4 Feature selection algorithms

We divide existing feature selection algorithms into three categories, embedded, filter, and wrapper approach as in [22].[8] We review each category in detail by analyzing representative algorithms.

---

[7]If a training set is very small, it is recommended to reduce the number of features and thus reduce the algorithm's variance. If more instances are given, however, more features can be chosen to reduce the algorithm's bias.

[8]John et al. (1994) use only two groups, filter and wrapper.

2.4.1 Embedded models

In [22], concept learning algorithms [145, 204] are illustrated as one of the embedded models because they look for alternative subsets of features in the concept description when they misclassify new points. Decision tree algorithms such as ID3 [164], CART [37] and C4.5 [166] also include feature selection as a subroutine. Tree algorithms heuristically search for a good subset of features through the space of tree structures. At each stage, each feature is evaluated to determine how well it classifies the associated data points and the best feature is selected. Once the best feature is determined, the data set is partitioned based on the selected feature. The same process is repeated until it reaches a terminal node in which all points belong to the same class or when all features are evaluated.

The concept learning and decision tree methods were believed to scale well to domains with many irrelevant features because of their explicit mechanism for choosing relevant features. However, their performance degrades significantly as irrelevant features are inserted into target concepts [7, 108, 128]. Further they are effective only in domains where there is little interaction among the relevant features [22]. Several attempts have been made to consider interaction among features by applying lookahead techniques [117] or by constructing new features from existing ones [158] with some success.

Mathematical programming models have also been used to solve the combinatorial feature selection problem. The Robust Linear Program (RLP) [135, 18]

formulates class separation as a linear program and solves it by minimizing the average distance in the feature space between the separating plane and the misclassified examples. The RLP model was developed into the concave optimization feature selection model [31] that penalizes the number of non-zero coefficient values. Recently support vector machines (SVM) [203, 50] has been used for feature selection [29, 130]. The idea is based on the fact that linear SVM with 1-norm regularization inherently performs variable selection. The subset of variables with nonzero weights in linear SVM is used for obtaining nonlinear SVM with a nonlinear separating surface for regression or classification purposes.

Artificial neural networks [129, 86] that dynamically adjust network structure by removing links from input units can also be used for feature selection. The o optimal brain damage (OBD) [129] uses an approximation to the second derivative of the error with respect to each weight to determine the saliency of that weight. It removes the link whose weight has the lowest saliency, or least importance. Optimal brain surgeon (OBS) [86] computes the second derivatives (almost) exactly, but is computationally very expensive. Setiono and Liu [183] proposed a method that uses the network classification performance on a validation dataset instead of using a saliency measure.

2.4.2 Filter models

Most feature selection algorithms in the statistics and pattern recognition communities do not take into account the differences among learning algorithms [114].

These approach were named *filter* methods in [99], because the filtering step of original features is done before the induction step. We represent the filter approach as in Figure 2.1. In filter models, learning algorithms take data with only the selected

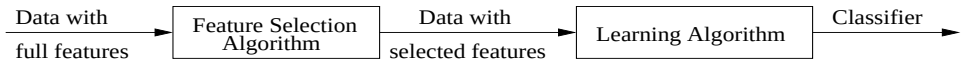

Figure 2.1: The filter approach to feature selection.

features and evaluate its accuracy. Filter models rely on abstract measures of features that indicate important properties of promising features such as orthogonality, large variance, multi-modality of marginal distributions, high kurtosis, and low entropy. The simplest filtering method used in text categorization tasks is to select the k features with the highest correlation value with the target function [11]. Statistical dependence such as mutual information between the features and the class label [15, 25] and distance metrics [59, 63, 110] also have been used.

A representative algorithm in filter models is FOCUS and its variants [7, 8, 9] that find the minimum subset of features that partitions the training data in such a way that no instances with the same values of the selected features have different classes. As noted in [99], FOCUS is very sensitive to noise and selects relevant but not useful features for generalization because of this restriction of pure partitions.[9]

[9]For example, FOCUS returns the patient's social security number (SSN) as the minimal feature subset to determine the label.

Further, FOCUS is prone to overfitting [44] and not practical for data with more than 25-30 features [115] because it will continuously select features to correct even a single inconsistency [44].

Relief [108, 109] and its variants [116, 113] focus on the ability to discriminate close pairs of examples that belong to different classes. Relief does not favor features that have a large variety of values [117] and can rank features in the order of the final relevance weights. Further, it considers the interaction among features and is robust against noisy data. In comparative experiments [108], FOCUS+ID3 was more effective than Relief+ID3 on noise-free data but Relief+ID3 was more accurate and efficient on noisy data. Relief, however, selects most of the features if they are relevant to the concept [108] and requires sufficient training examples.

Other filter algorithms have been tested with decision trees [42, 132, 133] and naive Bayes classifiers [122] based on various selection measurements such as rough sets theory [147], information theory [187], or cross-entropy [115].

2.4.3 Wrapper models

Another approach to feature selection uses the learning method as a subroutine, not as a postprocessor. In this approach, the feature selection algorithm exists as a wrapper around the learning algorithms as shown in Figure 2.2.

In the wrapper model, the feature selection algorithm searches for a good subset using the learning algorithms to evaluate feature subsets. The wrapper models first train the learning algorithms on training data with the selected features and

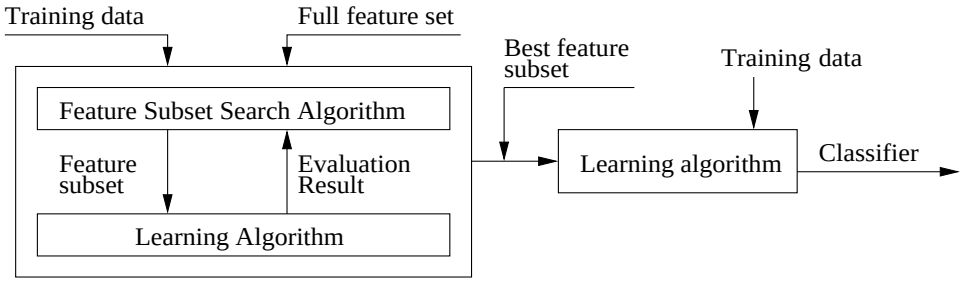

Figure 2.2: The wrapper approach to feature selection.

evaluate the resulting classifier on holdout data to get the estimated accuracy.[10] The feature subset with the highest accuracy is chosen as the final model. However, this approach can be very slow because the learning algorithm is called repeatedly.

Kohavi and John [113] showed that feature selection through the wrapper model significantly boosted the predictive accuracy of classifiers such as C4.5 and naive Bayes on several data sets. They used best first search based on accuracy estimates to find a subset of attributes. In particular, their comparative experiments between wrapper and filter models confirmed the old view noted by many researchers [17, 185] that it is inappropriate to evaluate the usefulness of an input variable without considering the algorithms that build the classification system.

Algorithms such as nearest-neighbor algorithms are expected to benefit from the wrapper approach because it by default takes into account all features. Experiments in [127] supported this fact. They proposed a heuristic wrapper algorithm,

[10]Holdout, bootstrap, and cross-validation are commonly used to estimate the predictive accuracy. For more information, please refer to [207].

OBLIVION, and compared it with simple nearest-neighbor algorithms and C4.5 on synthetic data. OBVILION can learn more accurate classifiers from many fewer instances in domains that involve irrelevant features.

BEAM is a nondeterministic search algorithm proposed by Aha and Bankert for cloud classification tasks [2, 3]. Their framework maintains a queue of feature subsets ordered by non-increasing 10-fold classification accuracy together with an indication of which derivable subsets have not been evaluated. They used IB1 [4] as an induction algorithm and beam-search variants of forward sequential selection (FSS) and backward sequential selection (BSS). In experiments on a cloud data set with 204 features and 1633 instances, feature selection using BEAM with FSS selected eleven features and increase IB1's 10-fold accuracy from 72.6% to 87.0%. On a sparse data set with 98 features but only 69 instances, they report a similar improvement, from 62.3% to 87.8%.

LVW [133] is a modified version of Las Vegas algorithms [32] and takes a probabilistic approach to explore the high-order relationship among features without resorting to exhaustive search. LVW generates random subsets of features and the smallest subset with the lowest error rate on the training data is saved. This process continues until the error rate is not updated for a certain number of cycles. LVW guarantees the optimal solutions at the $(k+1)$th experiment with the probability $l/(2^D - k)$ where D is the number of original features and l is the number of optima [133]. Though LVW with ID3 as a learning algorithm improved accuracy on some

real data sets, LVW with C4.5 did not significantly improve accuracy. In another experiment [126], the naive Bayes classifier was used with forward selection. The wrapper model was extended to Bayesian networks in [187] and for numeric prediction using k-nearest neighbor in [196].

2.4.4 Comparison of wrapper models with filter models

We compare wrapper models with filter models using four criteria: time complexity, bias, predictive accuracy, and susceptibility of overfitting.

In terms of computational cost, the filter approach is clearly the winner. The wrapper approach is slow because learning algorithms are called for each feature subset to estimate predictive accuracy. When cross validation is used to get a predictive estimate, time complexity in wrapper models gets much worse. Especially on large data sets, the filter approach becomes very attractive. There has been some research on improving the time complexity of wrapper models based on either caching [44] or instance selection [148].

In terms of bias, there is no clear winner. Koller & Sahami (1996) focus on an algorithm-independent paradigm for feature selection, viewing a learning algorithm as a biased method for approximating the probability distribution of class labels and transforming this distribution into a classification. By staying away from the bias of a particular learning algorithm, a feature subset selected truly reflects properties of the data itself and is independent of the learning algorithm. This, however, is clearly one of the disadvantages of the filter model in the sense that the subset selection algorithm

must take into account the biases of the learning algorithm. This is because ultimately the selected features are plugged into the same learning algorithm to predict unseen data [99, 113]. Selecting a subset of features should, therefore, not be based solely on the intrinsic discriminant properties of the data, but should be made relative to a given learning algorithm. Further, most of the learning algorithms conduct a very limited search in the space of possibly hypotheses. Ignoring these limitations can lead to feature subsets which are inappropriate for the learning algorithms used [112].

Since in the wrapper approach, the search bias for feature subsets is tailored to a particular learning algorithm, the wrapper model has been expected to return higher predictive accuracy. In [99, 113, 44], the wrapper approach was compared to the filter approach in terms of predictive accuracy. The wrapper model using various induction algorithms such as C4.5, ID3 and naive Bayes resulted in better performance than the filter model and showed in the most significant improvement on data sets with many features. However, the filter approach could be used as a preprocessing step of the wrapper approach for analysis of huge data sets [115].

A learning algorithm overfits the data set if it models the training data too well at the expense of generalization performance. Similarly, a search algorithm that explores a large portion of the space and that is guided by the accuracy estimates can choose a bad feature subset with a highly accurate estimate but poor predictive power [113, 103]. In the filter model, FOCUS, which relies on *consistency* to define the relevance of a feature to the target concept, is prone to overfit because

it will continuously select features to correct even a single inconsistency [44]. For the wrapper model, overuse of the accuracy estimates in feature subset selection can cause overfitting because the selected solution that has high predictive accuracy on the holdout data may not generalize well to test data. The effect of training data size on the possibility of overfitting was addressed in [114]. As a general guide, many algorithms employ the Occam's Razor bias to build as simple a model as possible that still achieves some acceptable level of performance on the training data.

Often, the tradeoff between more accurate estimates and more extensive exploration of the search space is referred to as the exploration versus exploitation problem [103]. However, it has been claimed that more extensive search can increase the probability of finding *fluke* rules that fit the data well but have low predictive accuracy [168, 150]. They used the term *oversearching* to describe this negative effect and provided a new search algorithm, layered search, to avoid the problems with oversearching. While the overfitting models are typically more complex than necessary to fit the training data, the final models in oversearching are not necessarily complex but still misleading. Recently Jensen and Cohen [98] claim that one single mechanism, multiple comparison procedures (MCPs), is responsible for attribute selection errors, overfitting and oversearching. Several solutions including cross-validation and Bonferroni adjustment are discussed in their paper [98].

2.5 Search algorithms

2.5.1 Review of previous algorithms

In this section, we review several search algorithms that could be used in the wrapper model when combined with a specific induction algorithm. They also could be used for feature selection directly without learning algorithms when combined with some feature selection criteria.[11] We divide search algorithms mainly into three groups: deterministic [137, 208, 191, 162], stochastic [186, 197, 198, 163, 79] and optimal methods [151, 71, 123].

One of the simplest algorithms, *backward sequential selection* (BSS) was introduced in [137]. This starts with the full set of features, and greedily removes the one that most improves performance, or degrades performance slightly. Another simple algorithm, *forward sequential selection* (FSS) or bottom-up [208], starts with the empty set of features, and greedily adds features. FSS is computationally faster than BSS because in BSS, the criterion function must be computed in larger dimensional spaces. BSS, however, is able to monitor continuously the amount of information loss incurred [111] and may capture interacting features more easily [113].

However, both algorithms produce suboptimal solutions because of their irrevocable selection process and inability to consider the interaction among features [198]. BSS and FSS are generalized to GBSS(g) and GFSS(g) in such a way that g features are evaluated at the same time and the best g-feature subset is chosen

[11] For various measures and their formula, refer to [111], pages 63-70.

for addition or deletion in the algorithms. The Plus l-take away r algorithm [191] adds l features by FSS and deletes r features by BSS and repeats this process. In a generalized algorithm GPTA(l, r), GFSS(l) and GBSS(r) are used instead of FSS and BSS in the inclusion and exclusion stage respectively [110]. The sequential forward floating selection (SFFS) algorithm in [162] is another generalized version of PTA(l,r). SFFS can backtrack indefinitely as long as the backtrack finds a better feature subset than the feature subset obtained so far. However, no non-exhaustive sequential procedure of feature selection can be guaranteed to produce the optimal subset because the number of possibilities grows exponentially [52].

Genetic algorithms (GAs) were initially introduced in [89] and have been applied to many optimization problems because of their robustness in large search spaces. Since GAs are domain independent, they are ideal for applications where domain knowledge and theory is not available [101]. GAs for feature selection have been combined with various classifiers [186, 51, 211] and clustering algorithms [160, 82, 106]. In a GA approach, a given feature subset is represented as a binary string, a *chromosome* of length D, with a zero or one in position i denoting the absence or presence of feature i. Note that D is the total number of features and a fixed or variable-sized of population of chromosomes is maintained over the generations. Each chromosome is evaluated to determine its "fitness," and may survive into the next generation and/or reproduce depending on its fitness. New chromosomes are created from old chromosomes by the processes of crossover and mutation. Instead of giving 0/1 weighting to

the features in a chromosome, a broader range of weights, 0 to 10, was tested in [163]. Two other genetic search algorithms, CHC and Common Features/Random Sample Climbing (CF/RSC) were proposed in [79].

The branch-and-bound (BAB) feature selection algorithm was proposed to find the optimal solutions without resorting to exhaustive search [151]. However, its monotonic assumption of the feature selection criterion may not be true and it is still impractical for problems with very large feature sets because of the exponential time complexity. Two variants of BAB were proposed, RBAB [71] that based on *approximate monotonicity* assumption and RBABM [123] that based on *k-monotonic* assumption.[12]

Comparative studies among search algorithms can be found in [186, 69, 97, 123].

2.5.2 Evolutionary Local Selection Algorithm (ELSA)

2.5.2.1 Local selection and algorithm details

ELSA springs from algorithms originally motivated by artificial life models of adaptive agents in ecological environments [139]. 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.

In a standard evolutionary algorithm, an agent is selected for reproduction

[12]A criterion function J is *k-monotonic* if $X' \subset X, |X| - |X'| \geq k \Rightarrow J(X') \leq J(X)$ for every X', X. BAB does the optimal search only when J is 1-monotonic.

based on how its fitness compares to that of other agents. In ELSA, an agent (candidate solution) may die, reproduce, or neither based on an endogenous energy level that fluctuates via interactions with the environment. The energy level is compared against a constant selection threshold for reproduction.

By relying on such *local* selection, ELSA reduces the communication among agents to a minimum. The competition and consequent selective pressure is driven by the environment [142]. There are no direct comparison with other agents. Further, the local selection naturally enforces the diversity of the population by evaluating genetic individuals based on both their quality measurements and on the number of similar individuals in the neighborhood in objective space. The bias of ELSA toward diversity makes it ideal for multi-objective optimization, giving the decision maker a clear picture of Pareto-optimal solutions from which to choose. Previous research has demonstrated the effectiveness of ELSA for feature selection in both supervised [141, 107] and unsupervised [106] learning.

2.5.2.2 Agents, mutation and selection

Figure 2.3 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 bits and each of D bits is an indicator as to the corresponding feature is selected or not (1 if a feature is selected, 0 otherwise). Mutation is the main operator used to explore the search space, and crossover can be added if required depending on the

```
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(crossover(a, randommate))
      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 2.3: 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 sampling effects. We stop the algorithm after T iterations.

problem domain. The mutation operator randomly selects one bit of the agent and flips it. The crossover operator could be implemented in a number of different ways such as single-point, two-point, and uniform crossover [146]. In our experiments, we adopt the commonality-based crossover framework [49], where the offsprings inherits all the common features of the parents.

Each agent competes for a scare resource, energy, based on multi-dimensional fitness and the proximity of other agents in solution space. That is, as we review in detail in the following section, agents get energy from multiple objectives based on their fitness and the environment to which they belong. In the selection part of the

algorithm, each agent compares its current energy level with a constant reproduction threshold θ. If its energy is higher than θ, the agent reproduces: the agent and its mutated clone that was just evaluated become part of the new population, each with half of the parent's energy. If the energy level of an agent is positive but lower than θ, only the agent itself joins the new population. If an agent runs out of energy, it is killed.

The population size is maintained dynamically over iterations and is determined by the carrying capacity of the environment depending on the costs incurred by any action, and the replenishment of resources [142]. The population is also independent of the reproduction threshold, θ which only affects the energy stored by the population at steady-state.

2.5.2.3 Energy allocation and replenishment

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 offspring's 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.[13] We have an energy source for each criterion, divided

[13]Continuous objective functions are discretized.

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

$$Fitness(a, c) = \frac{F_c(a)}{P_c(F_c(a))}. \tag{2.1}$$

Agents 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 ($E_{cost} < \theta$).

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

$$E^c_{tot} = \frac{p_{max} E_{cost}}{C} \tag{2.2}$$

where C is the number of criteria considered. 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. Note that the total replenishment energy that enters the system at each iteration is $p_{max} \cdot E_{cost}$, 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, $p \cdot E_{cost}$, cannot be larger than the replenishment energy.

2.5.2.4 Advantages and disadvantages

One of the major advantages of ELSA is its minimal centralized control over agents. By relying on *local* selection, ELSA minimizes the communication among agents, which makes the algorithm efficient in terms of time complexity. The possible significant speedup with parallel programming in a distributed environment lead to the development of an agent-based system for information retrieval [140].

In terms of scalability, ELSA shows good performance. ELSA maintains varying size of population through evolutionary process and does not need a prior knowledge on problem domains to determine population size in advance. In the graph search problem and the unitation versus pairs experiment with increasing problem size, ELSA shows evidence of the scalability properties of local selection [141]. Note also that ELSA can be easily combined with any other predictive models such as artificial neural networks or decision trees.

For application, ELSA can be useful for various tasks in which the maintenance of diversity within the population is more important than a speedy convergence to the optimum. Feature selection is one such promising applications. Based on the well-covered range of feature vector complexities, ELSA is able to find feature subsets that are good or bad as a predictor. The superior ability to locate most of the Pareto front was demonstrated in a comparative experiment with other multi-criteria evolutionary algorithms in [141]. Further, once the Pareto front is located, we can easily find the

maximum complexity of the feature subsets beyond which no increase in accuracy is expected.

However, for problems requiring effective selection pressure, local selection may not be ideal because of its weak selection scheme. The only selection pressure that ELSA can apply comes from the sharing of resources. Therefore the way in which environmental resources are coupled with the problem space in a particular application of ELSA is crucial to its success. Another limitation of ELSA may be in the fact that the appropriate mapping of a problem onto an environmental model may be hard to determine [141].

CHAPTER 3
FEATURE SELECTION IN SUPERVISED LEARNING

3.1 Introduction

Over last decade, feature selection has garnered a lot of attention from various research area such as pattern recognition, machine learning, statistics, and data mining. The main idea of feature selection is to choose a subset of the original variables that keeps most of predictive information. In a supervised learning context, feature selection has been used to optimize the predictive performance of a considered model. Typically, the error rate on the test set is used to estimate the true error rate of classifiers trained on the selected features of the training set.

It is well-known that GAs perform a global search of combinatorial search spaces and are ideal for applications where domain knowledge is difficult to provide [56]. GAs can capture the interaction among features by modifying many features at a time through genetic operators. Further the GA-based approach does not rely on the monotonicity assumption that is often assumed in greedy search algorithms. This makes GAs ideal for feature selection of large-scale problems [123].

GAs have been combined with various learning algorithms for feature selection purposes. In [186, 46, 38], GAs with k-nearest neighbor (k-NN) have been applied to reduce the number of input variables for classification. In [170], GAs with k-NN is used for both optimal feature weighting and feature selection on two biomedical

problems. In their experiments, the use of feature selection improved the performance of k-NN more than feature weighting alone.

GAs have been also combined with decision trees [13, 195] or rule induction systems [199, 200] with some success. In particular, Cherkauer and Shavlik [51] presented a new algorithm, SET-Gen, which combines GAs with C4.5. Within a fixed-length chromosome, SET-Gen allows a feature to be selected multiple times for promoting feature diversity. In order to evaluate candidate solutions, SET-Gen uses a somewhat subjective fitness function that linearly combines two different measurements, accuracy and simplicity. Compared to single pruned tree, SET-Gen showed the comparable performance in terms of predictive accuracy on ten real-world data sets while reducing the size of resulting trees significantly. In [171], a new approach, Automatic Discoverer of Higher-Order Correlations (ADHOC), was introduced. ADHOC partitions the observed features into a number of clusters, called factors, and GAs are used to select at most one feature from every factor. Using C4.5 as an induction algorithm, ADHOC showed the better performance than single decision tree on 11 out of 14 data sets.

There has been much research combining GAs with artificial neural networks (ANNs) based on the fact that while ANNs are very effective for a local search, GAs are appropriate for a global search. For example, GAs have been used to adjust the weights of neural networks, to design neural architectures, and to find and extract learning rules. A comprehensive review on evolutionary neural networks can be found

in [212]. Several researchers have combined GAs with ANNs for feature selection purposes [38, 211]. In particular, Yang and Honavar [211] proposed a wrapper-based approach that evaluates individuals in terms of multiple criteria – accuracy and measurement cost of features. They tested their algorithm on 26 data sets and reported improved performance over single ANNs with the full set of features. However, they combined multiple fitness criteria in a subjective manner because standard GAs cannot consider multiple fitness criteria.

In order to address this limitation, a number of multi-objective extensions of GAs have been proposed in recent years [179, 91, 190]. Good reviews on these algorithms can be found in [202, 215]. Feature selection using multi-objective evolutionary algorithms (MOEA) in supervised learning has also been studied [66, 141]. In [66], a variation of the Niched Pareto Genetic Algorithm (NPGA) [91] is employed with two different neural networks, probabilistic neural networks and multilayer perceptrons. The misclassification rate and the feature subset size are used as the primary objectives to be minimized. In their experiments on two real data sets, the NPGA-based feature selection showed better coverage of Pareto fronts and identified at least as good solutions as the sequential forward and backward selection, for each complexity level.

The ability of ELSA to cover most of the objective space and to incorporate multiple fitness functions is well-established in [141]. In comparative experiments with other multi-criteria algorithms, ELSA covered a wider range of non-dominated

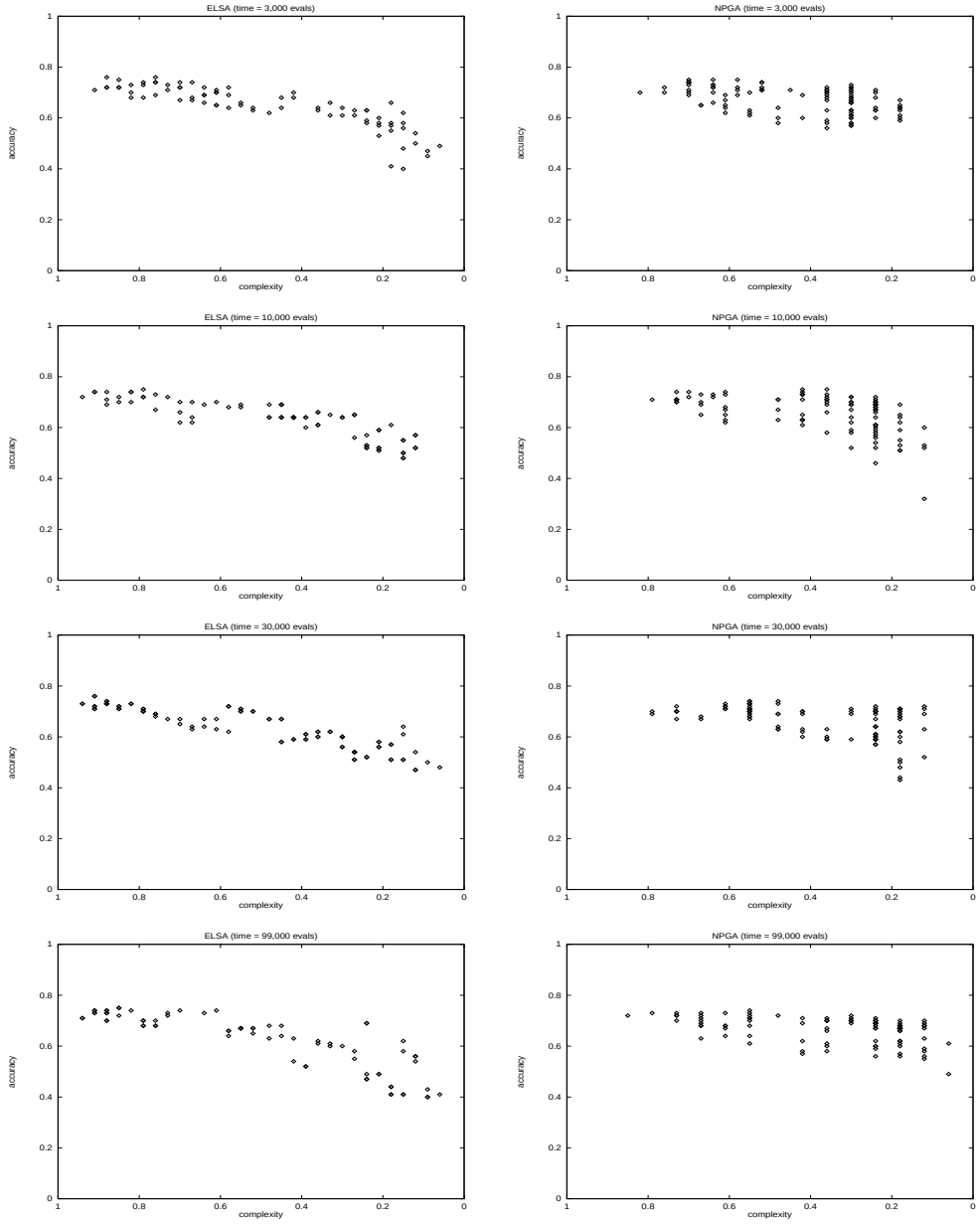

Figure 3.1: The Pareto fronts constructed from the ELSA and NPGA populations after 3, 10, 30 and 99 thousand function evaluations.

solutions as shown in Figure 3.1. After 3, 10, 30 and 99 thousand function evaluations, the ELSA population clearly covered more of the objective space than the NPGA population. This is due to the local selection mechanism in ELSA, which makes ELSA an ideal tool to be combined with neural networks for feature selection task.[1]

Based on this observation, we propose a new wrapper approach to feature selection that combines ELSA with ANNs. The returned output of ELSA/ANNs in the feature selection task will be the set of promising feature subsets of various complexities. At that point, we can identify the maximum complexity of the feature subsets that we should consider, beyond which no increase in accuracy is expected. Or we can do more thorough and costly analysis for attractive candidates such as cross-validation to obtain a more reliable estimate of generalization accuracy. This works very well in an any-time learning context, in which little overhead is required to maintain a record of the best individual encountered so far.

This chapter is organized as follows. Section 3.2 proposes a database marketing problem, to which our ELSA/ANN model will be applied. In Section 3.3, we describe the structure of the ELSA/ANN model, and review the feature subset selection procedure. In Section 3.4, we present experimental results of the ELSA/ANN model, and compare it to a PCA/logit model. Using test-mailing responses on insurance policies, we show that there is a trade-off between model interpretability and

[1]Note that this framework can easily be adapted to other predictive models such as decision trees or support vector machines.

predictive accuracy. In particular, we obtain both high model interpretability and high predictive accuracy only when the firm is specific about the way model forecasts will be used to select households in future mailings. In contrast, interpretability must be sacrificed to preserve predictive accuracy when the firm is vague about its selection rule. Section 3.5 concludes this chapter and provides suggestions about future research directions.

3.2 Problem specification

One of the key problems in database marketing is the identification and profiling of households who are most likely to be interested in a particular product or service. Due to the growing interest in micro marketing, many firms devote considerable resources to identifying households that may be open to targeted marketing messages. The availability of data warehouses combining demographic, psychographic and behavioral information further encourages marketing managers to use database-based approaches to develop and implement marketing programs.

Database marketers use different tools, depending upon what is known about particular households. Routine mailings to existing customers are typically based upon the RFM (recency, frequency, monetary) approach that targets households using knowledge of the customer's purchase history [181]. Mailings to households with no prior relationship with the firm are based upon the analysis of the relationship between demographics and the response to a test mailing of a representative household sample. Given the large number of potential demographics available, data dimen-

sion reduction is an important factor in building a predictive model that is easy to interpret, cost effective, and generalizes well to unseen cases. Commonly, principal component analysis (PCA) of demographic information [100] is used to prepare new variables for this type of analysis. These new variables are then used as predictors in a logistic regression on the test mailing responses.

We propose a new approach to building predictive models for identifying prospective households. The new methodology combines genetic algorithms (GAs) for choosing predictive demographic variables with artificial neural networks (ANNs) for developing a model of consumer response. We exploit the desirable characteristics of GAs and ANNs to achieve two principal goals of household targeting: model interpretability and predictive accuracy. Our approach is different from previous studies on direct marketing because of our consideration of multiple objectives [131] and data reduction [19].

Data reduction of demographic information is performed via feature selection in our approach. Feature selection can also significantly improve the comprehensibility of the resulting classifier models. Even a complicated model - such as a neural network - can be more easily understood if constructed only from a few variables. In database marketing applications, it is important for managers to understand the key drivers of consumer response. A predictive model that is essentially a "black box" is not useful for developing comprehensive marketing strategies.

In our work, the Evolutionary Local Search Algorithm (ELSA) is used to search

through the possible combinations of features. Two quality measurements – hit rate (which is maximized) and complexity (which is minimized) – are used to evaluate the quality of each feature subset. ELSA performs a local search in the space of feature subsets by evaluating genetic individuals based on both their quality measurements and on the number of similar individuals in the neighborhood in objective space.

The input features selected by ELSA are used to train an artificial neural network that predicts "buy" or "not buy." Using information from households with an observed response, the ANN is able to learn the typical buying patterns of customers in the dataset. The trained ANN is tested on an test set, and a proposed model is evaluated in terms of both the hit rate and the complexity (number of features) of the solution. This process is repeated many times as the algorithm searches for a desirable balance between predictive accuracy and model complexity. The result is a highly accurate predictive model that uses only a subset of the original features, thus simplifying the model and reducing the risk of overfitting. Because the algorithm identifies variables with no predictive value, it also provides useful information on reducing future data collection costs.

3.3 ELSA/ANN model for customer targeting

3.3.1 Structure of ELSA/ANN model

Our predictive model of household buying behavior is a hybrid of the ELSA and ANN procedures. In this approach, ELSA identifies relevant consumer descriptors that are used by the ANN to forecast consumer choice. We focus here on the structure

of the approach and the criteria used to select an appropriate predictive model.

The model setup is shown in Figure 3.2. ELSA searches for a set of feature subsets and passes them to an ANN. The ANN extracts predictive information from each subset and learns the patterns using a randomly selected 2/3 of the training data. Once an ANN learns the data patterns, the trained ANN is evaluated on the remaining 1/3 of the training data, and returns two evaluation metrics, $F_{accuracy}$ and $F_{complexity}$, to ELSA. It is important to note that in both the learning and evaluation procedures, the ANN uses only the selected features.

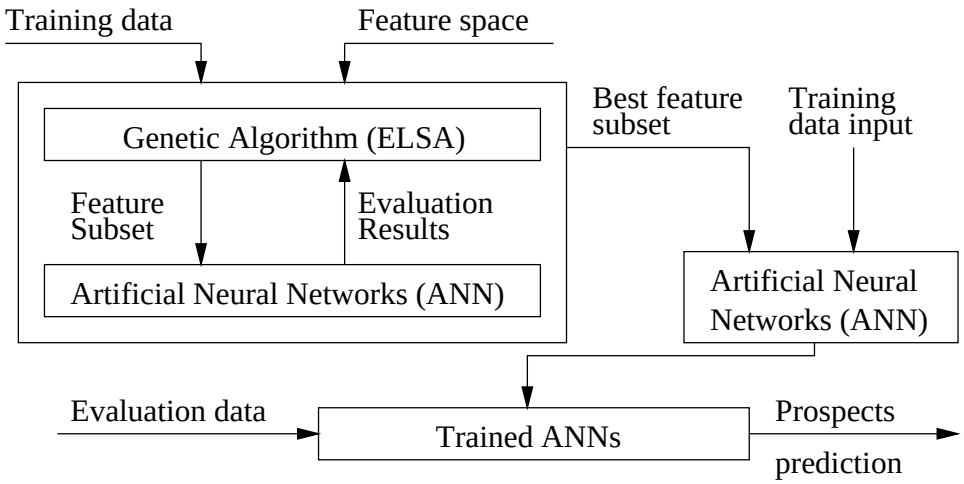

Figure 3.2: The structure of ELSA/ANN model. ELSA searches for a good subset of features and passes them to an ANN. The ANN calculates the "goodness" of each subset and returns two evaluation metrics to ELSA.

Based on the returned metric values, ELSA biases its search as described in Section 2.5.2.3 to maximize the two objectives. This routine continues until the max-

imum number of iterations is attained. All evaluated solutions over the generations are saved into an off-line solution set without comparison to previous solutions. In this way, high-quality solutions are maintained without affecting the evolutionary process.

Among all the evaluated subsets, we choose for further evaluation the set of candidates that satisfy a minimum hit rate threshold. With these chosen candidates, we start a more rigorous selection procedure, 10-fold cross validation. In this procedure, the training data is divided into 10 non-overlapping groups. We train an ANN using the first nine groups of training data and test the trained ANN on the remaining group. We repeat this procedure until each of the 10 groups is used as a test set once. We then take the average of the accuracy measurements over the 10 folds and call it an *intermediate* accuracy. We repeat the 10-fold cross validation procedure five times and average the five intermediate accuracy estimates. We call this the *estimated* accuracy through the following sections.

For evaluation purposes, we select a single "best" solution in terms of both estimated accuracy and complexity. We subjectively decided to pick a solution with the minimal number of features at the marginal accuracy level.[2] Once we decide on the best solution, we train the ANN using all the training data with the selected features only. The trained model is then used to rank the potential customers (the records in the evaluation set) in descending order by the probability of buying RV

[2]If other objective values are equal, we prefer to choose a solution with small variance.

insurance, as predicted by the ANN. We finally select the top $x\%$ of the prospects and calculate the *actual* accuracy of our model using the actual choices of the evaluation set households.

3.3.2 Evaluation metrics

We define two heuristic evaluation criteria, $F_{accuracy}$ and $F_{complexity}$, to evaluate selected feature subsets. Each objective, after being normalized into 25 intervals to allocate energy, is maximized by ELSA.

***$F_{accuracy}$*:** The purpose of this objective is to favor feature sets with a higher hit rate. Each ANN takes a selected set of features to learn data patterns and predicts which potential customers will actually purchase the product. In our application, we define two different measures, $F^1_{accuracy}$ and $F^2_{accuracy}$ for two different experiments. Experiment 1 assumes that the managers can specify in advance the rule to be used in select households for mailings. We select the top 20% of potential customers in descending order of the probability of purchasing the product and compute the ratio of the number of actual customers, AC, out of the chosen prospects, TC. We calculate $F^1_{accuracy}$ as follows:

$$F^1_{accuracy} = \frac{1}{Z^1_{accuracy}} \frac{AC}{TC} \tag{3.1}$$

where $Z^1_{accuracy}$ is an empirically derived constant to normalize $F^1_{accuracy}$.

In Experiment 2, we consider a generalization of Experiment 1. We first divide the range of customer selection percentages into 50 intervals with equal width

(2%) and measure accuracy at the first m intervals only.[3] At each interval $i \leq m$, we select the top $(2 \cdot i)\%$ of potential customers in descending order of the probability of purchasing the product and compute the ratio of the number of actual customers, AC_i, out of the total number of actual customers in the evaluation data, Tot. We multiply the width of interval and sum those values to get the area under the lift curve over m intervals. Finally we divide it by m to get our final metric, $F^2_{accuracy}$. We formulate it as follows:

$$F^2_{accuracy} = \frac{1}{Z^2_{accuracy}} \frac{1}{m} \sum_{i=1}^{m} \frac{AC_i}{Tot} \cdot 2 \tag{3.2}$$

where $Tot = 238$, $m = 25$ and $Z^2_{accuracy}$ is an empirically derived constant to normalize $F^2_{accuracy}$.

***F*$_{complexity}$:** This objective is aimed at finding parsimonious solutions by minimizing the number of selected features as follows:

$$F_{complexity} = 1 - \frac{d-1}{D-1} \tag{3.3}$$

where d and D represent the dimensionality of the selected feature set and of the full feature set, respectively. 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 generalization.

---

[3]This could be justified in terms of costs to handle the chosen prospects and the expected accuracy gain. As we select more prospects, the expected accuracy gain will go down. If the marginal revenue from an additional prospect is much greater than the marginal cost, however, we could sacrifice the expected accuracy gain. Information on mailing cost and customer value was not available in this study.

3.4 Application

The new ELSA/ANN methodology is applied to the prediction of households interested in purchasing an insurance policy for recreational vehicles. To benchmark the new procedure, we contrast the performance of the ELSA/ANN methodology to an industry-standard logit approach that summarizes household background information using principal components analysis. We evaluate the ELSA/ANN approach using two experiments. In Experiment 1, we inform the algorithm of the way in which the predictive model will be used by managers to select households for a direct mail solicitation. In Experiment 2, we leave this information vague. We show that the new approach provides improvements in forecasting accuracy, but that model complexity is contingent on the amount of information about the managerial decision rule.

3.4.1 Data description

The data are taken from a solicitation of 9,822 European households to buy insurance for a recreational vehicle. These data, taken from the CoIL 2000 forecasting competition [105], provide an opportunity to assess the properties of the ELSA/ANN procedure in a customer prospecting application.[4] In our analysis, we use two separate datasets: a training set with 5822 households and an evaluation set with 4000 households. The training data is used to calibrate the model and to estimate the hit

[4]We use a dataset on consumer responses to a solicitation for "caravan" insurance policies. A "caravan" is similar to a recreational vehicle in the United States. For more information about the CoIL competition and the CoIL datasets, refer to the Web site http://www.dcs.napier.ac.uk/coil/challenge/.

Table 3.1: Household background characteristics.

Feature ID	Feature Description
1	Number of houses owned by residents
2	Average size of households
3	Average age of residents
4-13	Psychographic segment: successful hedonists, driven growers, average family, career loners, living well, cruising seniors, retired and religious, family with grown ups, conservative families, or farmers
14-17	Proportion of residents with Catholic, Protestant, others and no religion
18-21	Proportion of residents of married, living together, other relation, and singles
22-23	Proportion of households without children and with children
24-26	Proportion of residents with high, medium, and lower education level
27	Proportion of residents in high status
28-32	Proportion of residents who are entrepreneur, farmer, middle management, skilled laborers, and unskilled laborers
33-37	Proportion of residents in social class A, B1, B2, C, and D
38-39	Proportion of residents who rented home and owned home
40-42	Proportion of residents who have 1, 2, and no car
43-44	Proportion of residents with national and private health service
45-50	Proportion of residents whose income level is < \$30,000, \$30,000-\$45,000, \$45,000-\$75,000, \$75,000-\$123,000, >\$123,000, and average
51	Proportion of residents in purchasing power class
52-72	Scaled contribution to various types of insurance policies such as private third party, third party firms, third party agriculture, car, van, motorcycle/scooter, truck, trailer, tractor, agricultural M/C, moped, life, private accident, family accidents, disability, fire, surfboard, boat, bicycle, property, social security
73-93	Scaled number of households holding insurance policies for the same categories as in scaled contribution attributes

rate expected in the evaluation set. Of the 5822 prospects in the training dataset, 348 purchased RV insurance, resulting in a hit rate of $348/5822 = 5.97\%$. From the manager's perspective, this is the hit rate that would be expected if solicitations were sent out randomly to consumers in the firm's database.

The evaluation data is used to validate the predictive models. Our predictive model is designed to return the top $x\%$ of customers in the evaluation dataset judged to be most likely to buy RV insurance. The model's predictive accuracy is examined by computing the observed hit rate among the selected households. It is important to understand that only information in the training dataset is used in developing the model. Data in the evaluation dataset is used exclusively for forecasting.

In addition to the observed RV insurance policy choices, each household's record also contains 93 additional variables, containing information on both socio-demographic characteristics (variables 1-51) and ownership of various types of insurance policies (variables 52-93). Details are provided in Table 3.1. The socio-demographic data are based upon postal code information. That is, all customers living in areas with the same postal code have the same socio-demographic attributes. The insurance firm in this study scales most socio-demographic variables on a 10-point ordinal scale (indicating the relative likelihood that the socio-demographic trait is found in a particular postal code area). This 10-point ordinal scaling includes variables denoted as "proportions" in Table 3.1. For the purposes of this study, all these variables were regarded as continuous. The psychographic segment assignments (at-

tributes 4-13), however, are household-specific and are coded into ten binary variables.

In our subsequent discussion, the word feature refers to one of the 93 variables listed in Table 3.1. For example, the binary variable that determines whether or not a household falls into the "successful hedonist" segment is a single feature. Accordingly, in the feature selection step of the ELSA/ANN model, the algorithm can choose to use any possible subset of the 93 variables in developing the predictive model.

3.4.2 Experiment 1

In Experiment 1, we maximize the hit rate when choosing the top 20% potential customers as in Kim and Street (2000). We select the top 20% of customers in the evaluation dataset using the model created by the ELSA/ANN procedure. The actual choices of these households provide a measure of the hit rate. For comparison purposes, we implemented a principal component analysis (PCA) of the household background characteristics followed by a logistic regression of the insurance policy choice data. PCA is analogous to our feature selection procedure to reduce data dimension. The logistic regression is, in fact, an example of a very simple ANN. The PCA/logit approach is commonly used by industry consultants in developing household selection rules.

We also implemented an intermediate model, ELSA/logit, for comparison purposes. The ELSA/logit model is different from ELSA/ANN in the sense that it uses

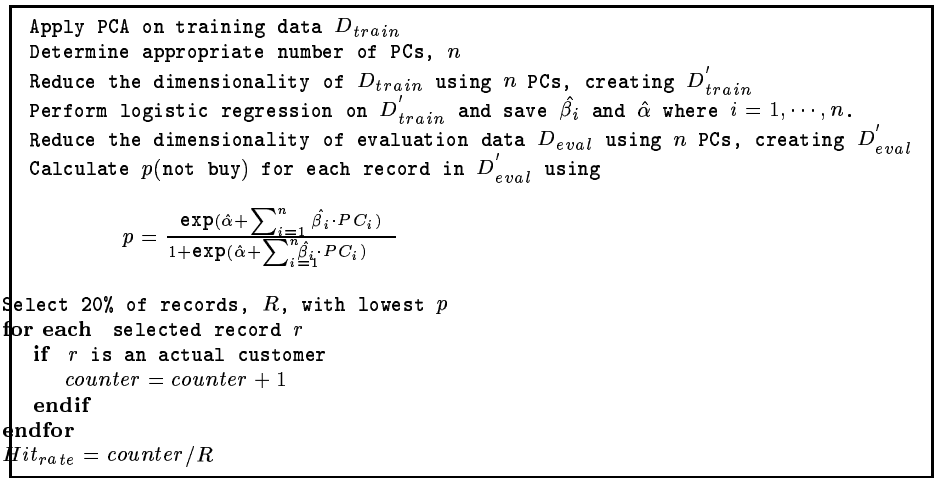

Figure 3.3: The implementation procedure of PCA/logit model.

only one hidden node.[5] We use the same criterion to select the final solution of ELSA/logit as in ELSA/ANN. The motivation behind the ELSA/logit model is the decomposition of the accuracy gain of ELSA/ANN into two sources: feature selection and response function approximation. The difference in results between PCA/logit and ELSA/logit can be attributed to characteristics of feature selection, while the difference in results between ELSA/logit and ELSA/ANNs can be attributed to the greater flexibility of ANN in approximating the response model.

Before discussing results, we first briefly summarize our implementation of the PCA/logit benchmark model in Figure 3.3. We selected 22 principal components. This is the minimum required to explain more than 90% of the variance in the training set. In order to get the estimated hit rate, we implement 10-fold cross validation on the training set as shown in Figure 3.4. In the cross validation procedure, the scores

[5]ELSA/ANN models use $\sqrt{node_{in}}$ where $node_{in}$ represents the number of input nodes.

of the PC's are estimated using different portions of the data each time to get the estimated hit rate.

```
Divide training data D_train into 10 equal-sized subsets
for each subsets D_train[i], i = 1, ..., 10
   Define D_train[i]^c = D_train - D_train[i]
   Apply PCA on D_train[i]^c, and select n PCs
   Reduce the dimensionality of D_train[i]^c using n PCs
   Do logistic regression on reduced D_train[i]^c
   Reduce the dimensionality of D_train[i] using n PC scores
   Calculate p(not buy) using the formula in Figure 3.3
   Pick 20% of records, R[i], with lowest p
   for each selected record r
      if r is an actual customer
         counter[i] = counter[i] + 1
      endif
   endfor
endfor
Hit_rate = sum_{i=1}^{10} counter[i] / sum_{i=1}^{10} R[i]
```

Figure 3.4: The implementation procedure of cross-validation for PCA/logit model. We used the same number of PCs, $n = 22$, as we did in Figure 3.3.

We set the values for ELSA parameters in the ELSA/ANN and ELSA/logit models as follows: $\Pr(mutation) = 1.0$, $p_{max} = 1{,}000$, $E_{cost} = 0.2$, $\theta = 0.3$, and $T = 2{,}000$. We select the single solution which has the highest expected hit rate among those solutions that have fewer than 10 features selected in both models. We evaluated each model on the evaluation set. Our results are summarized in Table 3.2 and Figure 3.5. The hit rates from the three different models are shown as percentages with standard deviation. The column marked "# Correct" shows the number of actual customers who are included in the chosen top 20%. The number in parenthesis represents the number of selected features except for the PCA/logit model, where it represents the number of PCs selected.

Table 3.2: Results of Experiment 1.

Model (# Features)	Training set	Evaluation set
	Hit Rate ± s.d	# Correct
PCA/logit (22)	12.83% ± 0.498%	109
ELSA/logit (6)	15.73% ± 0.203%	115
ELSA/ANN (7)	15.92% ± 0.146%	120

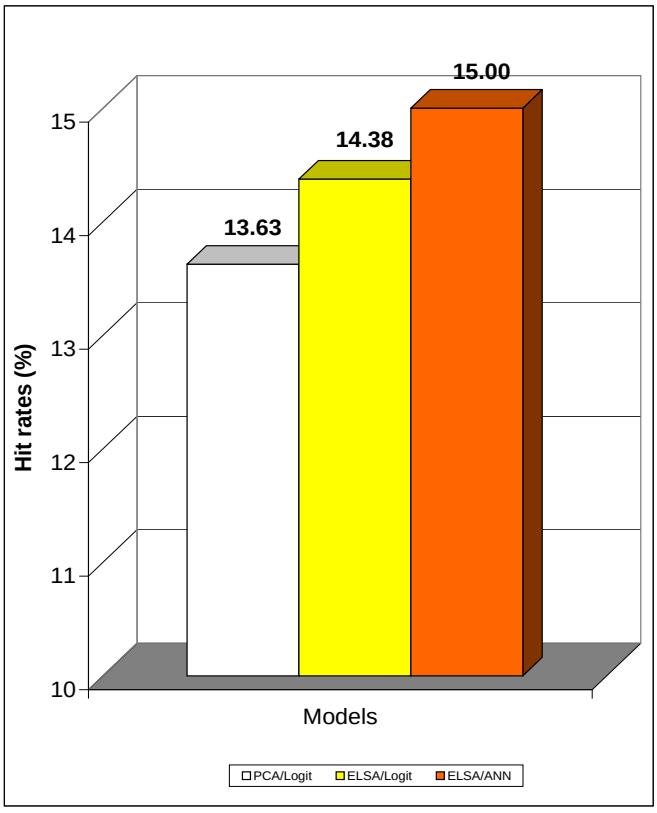

Figure 3.5: Hit rates of three different models on the evaluation data set. Each model chooses the best 20% of customers in the evaluation data set in terms of the estimated probability of buying a caravan insurance policy.

In terms of the actual hit rate, all three models work very well. Even the model with lowest actual hit rate (PCA/logit) is 2.3 times better than the hit rate expected by mailing to these households at random (5.97%). The model generated by the ELSA/ANN procedure returns the highest actual hit rate. As noted earlier, the difference in actual hit rate between PCA/logit and ELSA/logit provides an estimate of the accuracy gain that comes from the ELSA feature selection procedure. The difference in actual hit rate between ELSA/logit and ELSA/ANN provides an estimate of the accuracy gain that comes from the additional flexibility that ANN provides in approximating the true response function. Note that the highest hit rate is ($238/800 \approx 29.75\%$). In this application, both aspects of the ELSA/ANN procedure contribute equally to the improved accuracy of the model (1.37% point). This improvement in actual hit rate could lead to a significant gain in profit as the number of targeted prospects increases.

Judging the interpretability of a model is necessarily subjective. An advantage of the ELSA/ANN approach is that predictive features are clearly highlighted. In contrast, the PCA/logit model uses all of the features in constructing the principal component scores. We show the seven features that the ELSA/ANN procedure selected in Table 3.3.

With the exception of the "Average Family" psychographic segment, all other features are reports of the insurance buying behavior of the household's postal code area. The feature reporting car insurance makes considerable sense, given the fact that

Table 3.3: Selected features by ELSA/ANN in Experiment 1.

Feature Type	**Selected Features**
Demographic features	"Average Family" psychographic segment
Behavioral features	Amount of contribution to third party policy, car policy, moped policy and fire policy, and number of households holding third party policies and social security policies

the firm is soliciting households to buy insurance for recreational vehicles. Further evaluation shows that prospects with at least two insured autos are the most likely RV purchasers. Moped policy ownership is justified by the fact that many people carry their mopeds or bicycles on the back of RVs. Those two features are selected again by the ELSA/logit model.[6] Using this type of information, we are able to build a potentially valuable profile of likely customers [105].

In general, the results are in line with marketing science work on customer segmentation, which shows that information about current purchase behavior is most predictive of future choices [174]. The fact that the ELSA/ANN model used only seven features for customer prediction also implies that the firm could reduce data collection and storage costs considerably. This is possible through reduced storage requirements ($86/93 \approx 92.5\%$), and the reduced labor and data transmission costs.

We also compare the three models in terms of lift curves.[7] Figure 3.6 shows the

[6]The other four features selected by the ELSA/logit model are: contribution to bicycle and fire policy, and number of trailer and lorry policies.

[7]Lift is defined as the percentage of all buyers in the database who are in the group selected for a direct mail solicitation. Under random sampling, the lift curve is a 45-degree

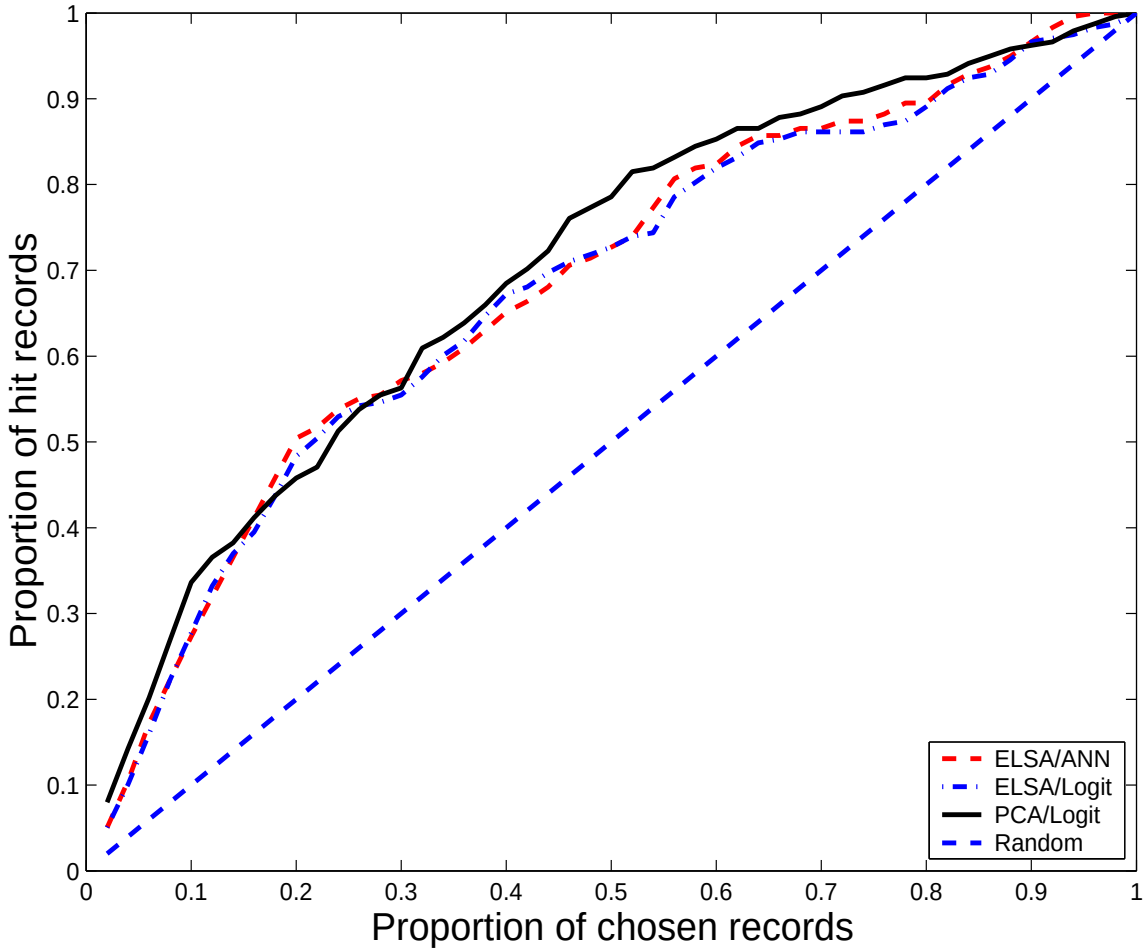

Figure 3.6: Lift curves of three models that maximize the hit rate when targeting the top 20% of prospects.

cumulative hit rate over the top $2 \leq x \leq 100$ % prospects. Clearly, our ELSA/ANN model is the best when the firm selects the top 20% of prospects for a direct mail solicitation. However, the performance of ELSA/ANN and ELSA/logit over all targeting percentages was worse than that of PCA/logit. This occurs because our solution is specifically designed to optimize the hit rate when managers select the top 20% of

line starting at the origin of the graph.

prospects. In contrast, the PCA/logit model is estimated without any knowledge of how model forecasts will be used in decision-making. This observation motivated a second experiment in which we attempt to improve the performance of ELSA/ANN model over a greater range of decision rules.

3.4.3 Experiment 2

In this experiment, we search for the best solution that maximizes the accuracy defined in a more global sense. The algorithm is designed to maximize the area under the lift curve, up to the top 50% of potential customers. Logically, the best solution from Experiment 1 is not necessarily the best solution in the more generalized environment of Experiment 2. In fact, our results are consistent with this observation. We also implemented the PCA/logit and the ELSA/logit model again for comparison purposes. We first show the generalized procedure of PCA/logit to get the estimated accuracy in Figure 3.7.

The ELSA/ANN and ELSA/logit models are adjusted to maximize the overall area under the lift curve over the same intervals as in PCA/logit. Because this new experiment is computationally much more expensive, we take a slightly different approach to choose the final solutions of ELSA/ANN and ELSA/logit. We used 2-fold cross validation estimates of all solutions and set the values of the ELSA parameters identically with the previous experiment except $p_{max} = 200$ and $T = 500$. Based on accuracy estimates, we choose a solution that has the highest estimated accuracy with less than half of original features in both models. We evaluate three models on

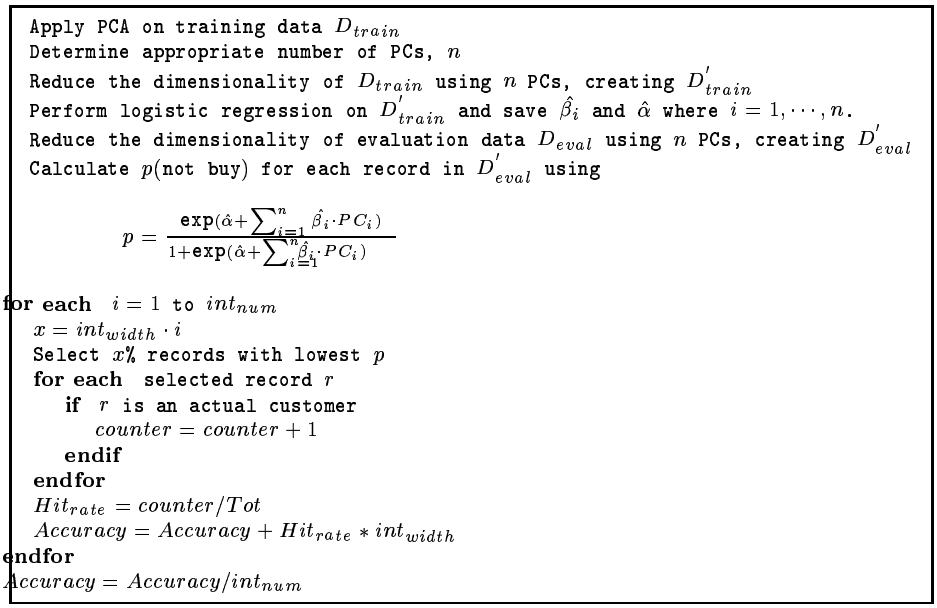

Figure 3.7: The generalized implementation of PCA/logit model. We use $n = 22$ (as in Experiment 1), $int_{num} = 25$, $int_{width} = 2$, and $Tot = 238$.

the evaluation set and summarize results in Table 3.4 and in Figure 3.8. In Table 3.4, the cumulative hit rates of the three models are shown over up to the top 50% of prospects. In practice, we optimize over the first 25 intervals which have the same width, 2%, to approximate the area under the lift curve.

Table 3.4: Summary of Experiment 2.

Model (# Features)	% of Selected									
	5	10	15	20	25	30	35	40	45	50
PCA/logit (22)	20.06	20.06	16.04	13.63	12.44	11.20	10.81	10.22	9.87	9.38
ELSA/logit (46)	23.04	18.09	15.56	13.79	12.13	12.04	10.97	10.54	10.03	9.53
ELSA/ANN (44)	19.58	17.55	16.40	14.42	13.13	11.96	10.97	10.40	9.98	9.64

Table 3.4 shows that the ELSA/ANN model has higher hit rates than PCA/logit over the solicitation range between 15% and 50% of total households. In particular, ELSA/ANN is best when choosing 15%, 20%, 25% and 50% of the targeting points, and tied for the best at 30%, 35% and 45%. The overall performance of ELSA/logit is better than that of PCA/logit. We attribute this to the fact that both models benefit from the ELSA feature selection methodology.

The lift curves in Figure 3.8 show that the ELSA/ANN has much improved global characteristics relative to Experiment 1. We, however, note that there are significant costs associated with this improved performance. First, the hit rate of ELSA/ANN at the 20% solicitation rate is now lower than in Experiment 1 (14.42% versus 15.00%). Second, it is no longer clear which aspects of the ELSA/ANN model are responsible for the improved global performance. Note that the rank order of ELSA/logit and ELSA/ANN shows no consistent pattern across the various solicitation percentages. Third, the well-established parsimony and interpretability of the models selected by ELSA/ANN in Experiment 1 is largely lost in Experiment 2. We attribute this partially to the fact that different selection points may have related but different optimal subsets of features. Correlation among features seems to contribute to the loss of parsimony. For instance, a particular variable related to insurance policy ownership that is part of the optimal subset at a 20% selection rate could easily be replaced by a different, correlated feature at 30%. It should be noted that the ELSA/ANN model is superior to PCA/logit model in the sense that ELSA/ANN

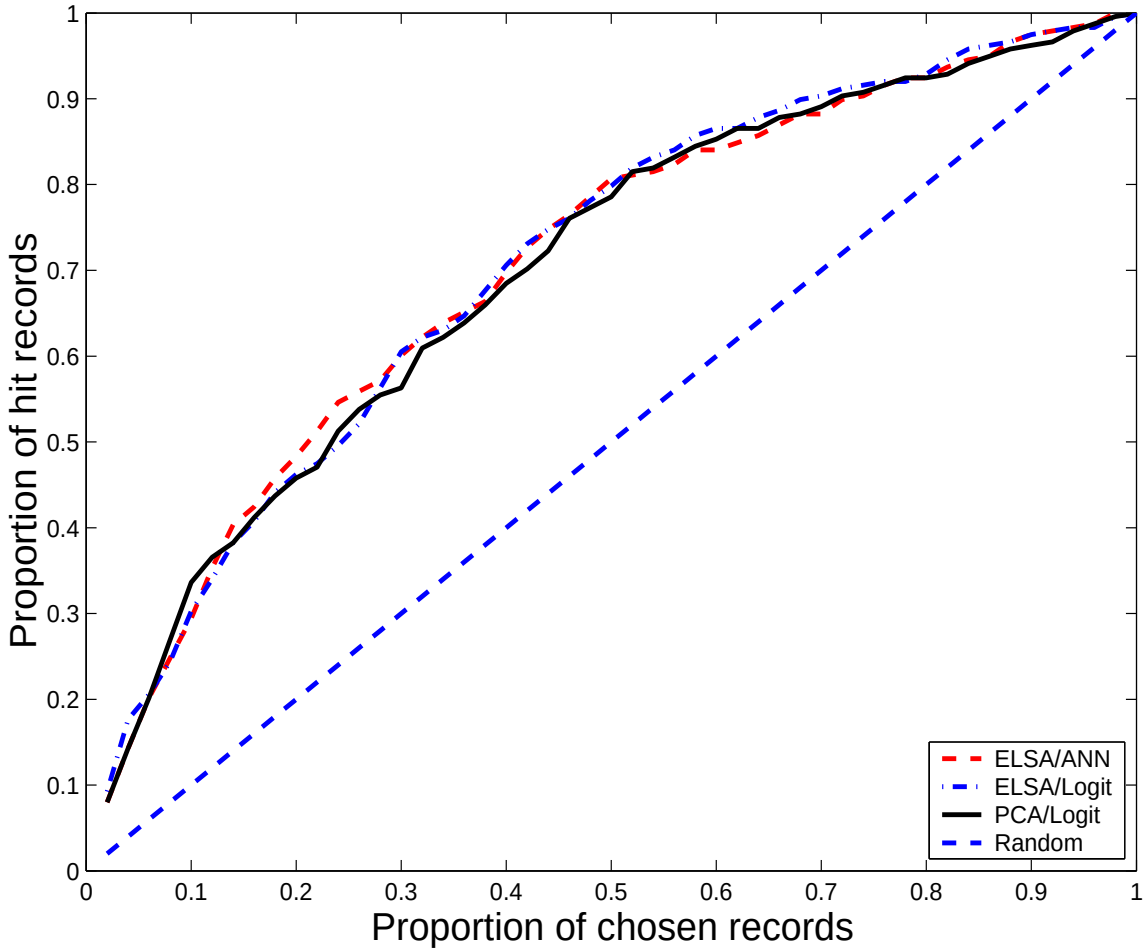

Figure 3.8: Lift curves of three models that maximize the area under lift curve when targeting upto top 50% of prospects. In practice, we optimize over the first 25 intervals which have the same width, 2%, to approximate the area under the lift curve.

works with feature subsets, while PCA/logit always requires the whole feature set to construct PC's.

These aspects of the solution provide strong evidence that there exists a key trade-off in building a predictive model. By focusing on a specific decision scenario (as in Experiment 1), we are able to construct a procedure that is parsimonious and has superior predictive performance. When the decision scenario is more ambiguous

(as in Experiment 2), we can improve predictive performance over a broad range, but sacrifice model interpretability.

3.5 Conclusion

In this chapter, we presented a novel approach for customer targeting in database marketing. We used an evolutionary algorithm, ELSA, to search for possible combinations of features and an artificial neural network (ANN) to score customers. When the decision rule was precise, the overall performance of ELSA/ANN was superior to the industry standard PCA/logit model both in terms of accuracy and in terms of interpretability. However, this superiority in interpretability is confined to specific decision conditions defined during model development and calibration. Under a more general decision scenario, ELSA/ANN yielded a more accurate model over a broad selection percentage range at the cost of increasing the number of predictive features in the specification.

One of the clear strengths of the ELSA/ANN approach is its ability to construct predictive models that reflect the direct marketer's decision process. Unlike a standard statistical approach like PCA/logit, the ELSA/ANN procedure can be easily modified to take into account different objectives. With information of campaign costs and profit per additional actual customer, a direct marketer could use ELSA/ANN to choose the best selection point where expected total revenue is maximized. In this way, it would be possible to determine the type of decision rule that the marketer should adopt, both in terms of solicitation percentage as well as

predictive rule. Because all mailing lists do not all have the same potential for the marketer, this approach would allow a predictive model and solicitation-mailing rule to be customized as the firm's database changes.

Our work provides additional evidence that there exists strong dependencies between model specification and managerial decision-making. When managers are clear about how a model will be used, the analyst can construct a highly specialized model that does better than general approaches (such as PCA/logit). When managers are vague, a less parsimonious model can be constructed which does better under some region of the decision space. The ELSA/ANN approach provides a new tool in which these trade-offs can be understood in the context of direct mail marketing applications.

CHAPTER 4
META-EVOLUTIONARY ENSEMBLES

4.1 Introduction

In recent years, a great deal of interest in the machine learning community has been generated by ensemble classifiers. These are predictive models that combine the predictions of a collection of individual classifiers, such as decision trees or artificial neural networks. Popular method such as Boosting, Bagging and Stacking differ in the ways that individual predictors are constructed, and in how their votes are combined. However, they have all demonstrated consistent – in some cases, remarkable – improvements in predictive accuracy over standard methods.

Much of the power of these methods comes from the diversity of the component classifiers. Intuitively, gathering a collection of problem solvers is only valuable if they are both accurate and diverse in their solutions. For instance, Boosting explicitly rewards a component classifier for correctly predicting difficult points. The diversity among component classifiers of ensemble has been proved critical to attain higher generalization accuracy [121, 85, 157] and can be obtained in many ways. Often different learning algorithms for the base classifiers, various sampling or weighting methods of the training examples, or projection of the examples onto different feature subspaces have been used to boost diversity among classifiers. However, little attention has been paid to the idea of creating an *optimal* collection of classifiers, or

indeed, what the idea of "optimality" might even mean in such a context.

We propose a new meta-ensembles algorithm to directly optimize ensembles by creating an two-level evolutionary environment. The various ensembles in this environment compete directly with one another, being judged on their estimated predictive performance. In addition, the underlying classifiers also compete with each other, being rewarded for correctly predicting the training examples. This reward is greater if the point in question is difficult, i.e., if it has been incorrectly classified by most of the other classifiers in the ensemble.

In particular, we employ feature selection not only to increase the predictive accuracy of an individual classifier but also to promote diversity among component classifiers in an ensemble [155]. Ensemble feature selection is based on the notion that different feature subsets among component classifiers of an ensemble can provide the necessary diversity. It is similar to the notion that different training samples among component classifiers provide the necessary diversity in ordinary ensemble methods.

Our approach to feature selection is again based on the wrapper model [113] of feature selection, which requires two components: a search algorithm that explores the combinatorial space of feature subsets, and one or more criterion functions that evaluate the quality of each subset based directly on the predictive model. In this work, we use artificial neural networks (ANNs) and decision trees as induction algorithms to evaluate the quality of the selected feature subsets. As a search algorithm, we turn to ELSA. For the general framework of ELSA, refer to Chapter 2.5.2.

In this chapter, we demonstrate the feasibility of such a model and show that the predictive accuracy obtained is better than a single classifier. Our model not only maintains higher or comparable predictive accuracy, but also builds ensembles smaller than traditional ensemble methods. In particular, our model provides the framework to answer how ensembles can best be constructed through the evolutionary process. We examine the relationship between classifier diversity and ensembles size to and the predictive accuracy of the ensemble. Finally, we compare the effectiveness of two different classifiers, neural networks and decision trees, in the framework of ensemble feature selection.

The remainder of this chapter is organized as follows. In Section 4.2 we review ensemble methods and ensemble feature selection algorithms, both separately and in combination. In Section 4.3 we present our bi-level approach to the ensemble feature construction, Meta-Evolutionary Ensembles (MEE) in detail. Section 4.4 presents and analyzes our experimental results. Section 4.5 addresses the directions of future research and concludes this chapter.

4.2 Ensemble methods and feature selection

4.2.1 Ensemble methods

Recently many researchers have combined the predictions of multiple classifiers to produce a better classifier, an ensemble, and often reported improved performance [35, 16, 210]. Bagging [33] and Boosting [72, 180] are the most popular methods for creating accurate ensembles. Bagging is a bootstrap ensemble method that trains

each classifier on a randomly drawn training set. Each classifier's training set consists of the same number of examples randomly drawn from the original training set, with the probability of drawing any given example being equal. Samples are drawn with replacement, so that some examples may be selected multiple times while others may not be selected at all. As a result, each classifier could return a higher test set error than a classifier using all of the data. However, when these classifiers are combined (typically by voting), the resulting ensemble produces lower test set error than a single classifier. The diversity among individual classifiers compensates for the increase in error rate of any individual classifier and improves prediction performance.

Boosting [72] produces a series of classifiers, with each training set based on the performance of the previous classifiers. New classifiers are constructed to better predict examples for which the current ensemble's performance is poor. This is accomplished using adaptive resampling, i.e., examples that are incorrectly predicted by previous classifiers are sampled more frequently, or alternately given a higher cost of misclassification. Boosting can be implemented in two different ways, Arcing [34] and AdaBoosting [72]. In Arcing, the classifiers' votes are weighted equally, while AdaBoost weights the predictions based on the classifiers' training error.

The effectiveness of Bagging and Boosting can be explained based on the bias-variance decomposition of classification error [16]. Bagging and Boosting are known to reduce errors by reducing the variance term [34]. According to [72], Boosting also reduces errors in the bias term by focusing on the misclassified examples. It

is noted that Boosting's effectiveness depends more on the data set than on the component learning algorithms, and it is often more accurate than Bagging. However, Boosting, unlike Bagging, can create ensembles that are much less accurate than a single classifier. In particular, Bagging performs much better than Boosting on noisy data sets because Boosting can easily overfit data by focusing more on the misclassified examples [60]. In most cases, the improved performance of an ensemble is largely obtained by combining the first few classifiers [156].

4.2.2 Ensemble feature selection algorithms

The improved performance of ordinary ensemble methods comes primarily from the diversity caused by re-sampling training examples. However, ensemble methods typically use the complete set of features to train component classifiers. Recently several attempts have been made to incorporate the diversity in feature dimension into ensemble methods. The Random Subspace Method (RSM) in [88, 87] was one early algorithm that constructed an ensemble by varying the feature subset. RSM used C4.5 as a base classifier and randomly chose half of the original features to build each classifier. Each classifier tree was constructed after all the training examples were projected to the subspace of selected features. The predictions were combined by simple majority voting. In comparative experiments, RSM demonstrated better performance on four public data sets than a single tree classifier with all the features and examples, and also outperformed Bagging and Boosting on the full-dimensional data sets [88, 87].

A more sophisticated way to select a subset of features for ensembles was proposed in [80]. They used a genetic algorithm (GA) to explore the space of all possible feature subsets. Their experiments paired four different ensemble methods, including Bagging and AdaBoost, with three different feature selection algorithms: complete, random, and genetic search. Using two table-based classification methods, ensembles constructed using features selected by the GA showed the best performance, followed by RSM. In [53], a new entropy measure of the outputs of the component classifiers was used to explicitly measure the ensemble diversity and to produce good feature subsets for ensemble using hill-climbing search.

Genetic Ensemble Feature Selection (GEFS) [155] also used a GA to search for possible feature subsets. Component classifiers (ANNs) in GEFS were explicitly evaluated in terms of both generalization accuracy and diversity. GEFS starts with an initial population of classifiers built using up to $2 \cdot D$ features, where D is the complete feature dimension. Using a variable feature subset size promotes diversity among the classifiers and allows some features to be selected more than once. Crossover and mutation operators search for new feature subsets, and new candidate classifiers are built for each of the new feature sets. Finally, GEFS prunes the population to the 100 most-fit members and majority voting is applied to determine the ensemble prediction. GEFS produces a good initial population, and in most cases produces better results the longer it runs. GEFS reported better estimated generalization than Bagging and AdaBoost on about two-thirds of 21 data sets tested. Longer chromosomes, however,

make GEFS computationally expensive in terms of memory usage [80]. Further, GEFS evaluates each classifier after combining two objectives in a subjective manner using $fitness = accuracy + \lambda\ diversity$, where $diversity$ is the average difference between the prediction of component classifiers and the ensemble. Since there is no obvious way to set the value of λ, GEFS dynamically adjusts the parameter based on the discrete derivatives of the ensemble error, the average population error and the average diversity within the ensemble.

Although all these methods reported improved performance using feature selection for ensemble construction ensemble, they have one common limitation in methodology: only one ensemble is considered. We propose a new algorithm for ensemble feature selection, Meta-Evolutionary Ensembles (MEE), that considers multiple ensembles simultaneously and allows each component classifiers to move into the best-fit ensemble. Genetic operators change the ensemble membership of the individual classifiers, allowing the size and membership of the ensembles to change over time. By having the various ensembles compete for limited resources, we can optimize their predictive performance.

In order to avoid costly global selection common to most GAs, we use a local selection mechanism in which classifiers compete with each other only if they belong to the same ensemble. Using ANNs as the base classifier and this EA for feature selection, we evaluate and reward each classifier based on two different criteria, accuracy and diversity. A classifier that correctly predicts data examples that other classifiers

in the same ensemble misclassify contributes more to the accuracy of the ensemble to which it belongs. We imagine that some limited "energy" is evenly distributed among the examples in the data set. Each classifier is rewarded with some portion of the energy if it correctly predicts an example. The more classifiers that correctly classify a specific example, the less energy is rewarded to each, encouraging them to correctly predict the more difficult examples. The predictive accuracy of each ensemble determines the total amount of energy to be replenished at each generation. Finally, we select the ensemble with the highest accuracy as our final classification model.

4.3 Meta-Evolutionary Ensembles

Pseudocode for the Meta-Evolutionary Ensembles (MEE) algorithm is shown in Figure 4.1, and a graphical depiction of the energy allocation scheme is shown in Figure 4.2. Each agent (candidate solution) in the population is first initialized with randomly selected features, a random ensemble assignment, and an initial reservoir of energy. The representation of an agent consists of $D+\log_2(G)$ bits. D bits correspond to the selected features (1 if a feature is selected, 0 otherwise). The remaining bits are a binary representation of the ensemble index, where G is the maximum number of ensembles. Mutation and crossover operators are used to explore the search space. A mutation operator randomly selects one bit of an agent and mutates it. Our crossover operator takes two agents, a parent a and a random mate, and scans through the bits of the two agents. If a difference is found, the value of the bit in a is flipped with a probability of 0.5. In this process, the mate contributes only to construct the

offspring's bit string, which inherits all the common features of the parents.

```
initialize population of agents, each with energy θ/2
while there are alive agents in Pop^i and i < T
   for each ensemble g
      for each record r in Data_test
         prevCount_{g,r} = count_{g,r}
         count_{g,r} = 0
      endfor
   endfor
   for each agent a in Pop^i
      a' = mutate(crossover(a, randomMate))
      g = group(a)
      train(a)
      for each record r in Data_test
         if (class(r) == prediction(r, a))
            count_{g,r} + +
            ΔE = E^{g,r}_{envt} / min(5, prevCount_{g,r})
            E^{g,r}_{envt} = E^{g,r}_{envt} − ΔE
            E_a = E_a + ΔE
         endif
      endfor
      E_a = E_a − E_cost
      if (E_a > θ)
         insert a, a' into Pop^{i+1}
         E_{a'} = E_a/2
         E_a = E_a − E_{a'}
      else if (E_a > 0)
         insert a into Pop^{i+1}
      endif
   endfor
   for each ensemble g
      replenish energy based on predictive accuracy
   endfor
   i = i + 1
endwhile
```

Figure 4.1: Pseudo-code of Meta-Evolutionary Ensembles (MEE) algorithm. In each iteration, the environmental energy for each pair of an ensemble g and a test example r is replenished based on the predictive accuracy of g. The main loop calls agents in random order and agents are rewarded based on their accuracy on each test record r, normalized by the number of other agents that correctly classify r in the same ensemble.

In each iteration of the algorithm, an agent explores a candidate solution (classifier) similar to itself, obtained via crossover and mutation. The agent's bit

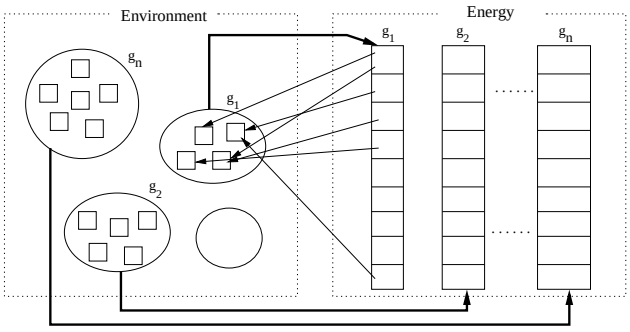

Figure 4.2: Graphical depiction of energy allocation in the MEE algorithm. Individual classifiers (small boxes in the environment) receive energy by correctly classifying test points. Energy for each ensemble is replenished between generations based on the accuracy of the ensemble. Ensembles with higher accuracy have their energy bins replenished with more energy per classifier, as indicated by the varying widths of the bins.

string is parsed to get a feature subset J. An ANN is then trained on the projection of the data set onto J, and returns the predicted class labels for the test examples. The agent collects ΔE from each example it correctly classifies, and is taxed once with E_{cost}. 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 classification task. But the energy also depends on the state of the environment. We have an energy source for each ensemble, divided into bins corresponding to each data point. For ensemble g and record index r in the test data, the environment keeps track of energy $E_{envt}^{g,r}$ and the number of agents in ensemble g, $count_{g,r}$ that correctly predict record r. The energy received by an agent for each correctly classified record r is given by

$$\Delta E = \frac{E_{envt}^{g,r}}{\min(5, prevCount_{g,r})}. \tag{4.1}$$

An agent receives greater reward for correctly predicting an example that most in its ensemble get wrong. The min function ensures that for a given point there is enough energy to reward at least 5 agents in the new generation.[1] 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, 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, an agent compares its current energy level with a constant reproduction threshold θ. If its energy is higher than θ, the agent reproduces: the agent and its mutated clone become part of the new population, with the offspring receiving half of its parent's energy. If the energy level of an agent is positive but lower than θ, only that agent joins the new population.

The environment for each ensemble is replenished with energy based on its predictive accuracy, as determined by majority voting with equal weight among base classifiers. We sort the ensembles in ascending order of estimated accuracy and apportion energy in linear proportion to that ranking, so that the most accurate ensemble is replenished with the greatest amount of energy per base classifier. Since the total

[1]Note that the number '5' is empirically determined through experiments. The better approach to avoid this problem is to automatically adjust the number of energy bins, which warrants further investigation.

amount of energy replenished also depends on the number of agents in each ensemble, it is possible that an ensemble with lower accuracy can be replenished with more energy in total than an ensemble with higher accuracy.

4.4 Experimental results

4.4.1 Experimental results of MEE/ANN

We first tested the performance of MEE combined with neural networks on several publicly available data sets [21]. We show the characteristics of our data sets in Table 4.1. For comparison purposes we chose several data sets that were also used in [155].

Table 4.1: Summary of the data sets used in experiments.

			Features		Neural Network	
Dataset	Records	Classes	Cont.	Disc.	Inputs	Epochs
credita	690	2	6	9	47	50
creditg	1000	2	7	13	63	50
diabetes	768	2	8	-	8	50
glass	214	6	9	-	9	50
heart-cleveland	303	2	8	5	13	50
hepatitis	155	2	6	13	32	50
house-votes-84	435	2	-	16	16	50
hypo	3772	4	6	21	31	50
ionosphere	351	2	34	-	34	50
iris	150	3	4	-	4	50
krvskp	3196	2	-	36	40	50
labor	57	2	8	8	29	50
segment	2310	7	19	-	19	50
sick	3772	2	6	21	31	50
sonar	208	2	60	-	60	50
soybean	683	19	-	35	84	50
vehicle	846	4	18	-	18	50

The weights and biases of the neural networks are initialized randomly between 0.5 and -0.5, and the number of hidden node is determined heuristically as $\sqrt{inputs}$. The other parameters for the neural networks include a learning rate of 0.1 and a momentum rate of 0.9. The number of training epochs was kept small (50) for computational reasons. The values for the various ELSA parameters are: Pr($mutation$) = 1.0, Pr($crossover$) = 0.8, $E_{cost} = 0.2$, $\theta = 0.3$, and $T = 30$. The value of $E_{envt}^{tot} = 30$ is chosen to maintain a population size around 100 classifier agents.

All computational results for MEE are based on the performance of the best ensemble and are averaged over five standard 10-fold cross-validation experiments. For each 10-fold cross-validation the original data set is first partitioned into 10 equal-sized sets, each maintaining the original class distribution. Each set is in turn used as an evaluation set while the classification system is trained on the other four sets. Within the training algorithm, each ANN is trained on two-thirds of the training set and tested on the remaining third for energy allocation purposes.

Experimental results are shown in Table 4.2. We present the performance of a single neural network using the complete set of features as a baseline algorithm. In the win-loss-tie results shown at the bottom of Table 4.2, a comparison is considered a tie if the intervals defined by one standard error [2] of the mean overlap. On the data sets tested, MEE shows consistent improvement over a single neural network.

---

[2]In our experiments, standard error is computed as standard deviation / $\sqrt{iter}$ where $iter = 5$

Table 4.2: Results of MEE/ANN with the fixed number of epochs.

	Single Net					MEE	
Dataset	Avg.	S.D.	Bagging	AdaBoost	GEFS	Avg.	S.D.
credita	85.4	0.87	86.2	84.3	86.8	85.9	0.72
creditg	71.7	0.43	75.8	74.7	75.2	75.6	0.78
diabetes	76.4	0.93	77.2	76.7	77.0	76.8	0.42
glass	56.6	2.27	66.9	68.9	69.6	58.8	1.21
cleveland	80.7	1.83	83.0	78.9	83.9	83.3	1.54
hepatitis	81.8	1.96	82.2	80.3	83.3	84.1	1.17
votes-84	95.4	0.30	95.9	94.7	95.6	95.6	0.52
hypo	93.8	0.09	93.8	93.8	94.1	93.9	0.06
ionosphere	90.2	0.38	90.8	91.7	94.6	92.7	0.42
iris	96.4	1.30	96.0	96.1	96.7	95.3	1.49
krvskp	98.8	0.63	99.2	99.7	99.3	99.3	0.10
labor	91.6	2.29	95.8	96.8	96.5	94.4	0.78
segment	92.3	0.97	94.6	96.7	96.4	93.2	0.28
sick	95.2	0.47	94.3	95.5	96.5	99.3	0.03
sonar	81.7	1.63	83.2	87.0	82.2	82.5	0.26
soybean	92.0	0.92	93.1	93.7	94.1	93.8	0.19
vehicle	74.7	0.48	79.3	80.3	81.0	76.4	1.12
Win-loss-tie	12-0-5		5-5-7	7-6-4	2-6-9		

We also include the results of Bagging, AdaBoost, and GEFS from [155] for indirect comparison. In these comparisons, we did not have access to the accuracy results of the individual runs. Therefore, a tie is conservatively defined as a test in which the one-standard-deviation interval of our test contained the point estimate of accuracy from [155].

However, MEE shows slightly worse performance compared to GEFS. We note that such comparisons are inevitably inexact, since subtle methodological differences can cause variations in estimated accuracy. For example, it is possible that the more complex structure of neural networks used in GEFS can learn more difficult patterns in

Table 4.3: Results of MEE/ANN with a varying number of epochs.

	Single Net					MEE		
Dataset	Avg.	S.D.	Bagging	AdaBoost	GEFS	Avg.	S.D.	Epoch
credita	84.3	0.30	86.2	84.3	86.8	86.4	0.52	40
creditg	71.7	0.43	75.8	74.7	75.2	75.6	0.78	50
diabetes	76.4	0.93	77.2	76.7	77.0	76.8	0.42	50
glass	57.1	2.69	66.9	68.9	69.6	61.1	1.73	100
cleveland	80.7	1.83	83.0	78.9	83.9	83.3	1.54	50
hepatitis	81.5	0.21	82.2	80.3	83.3	84.9	0.65	40
votes-84	95.9	0.41	95.9	94.7	95.6	96.1	0.44	40
hypo	93.8	0.09	93.8	93.8	94.1	93.9	0.06	50
ionosphere	89.3	0.85	90.8	91.7	94.6	93.5	0.81	100
iris	95.9	1.10	96.0	96.1	96.7	96.5	0.73	100
krvskp	98.8	0.63	99.2	99.7	99.3	99.3	0.10	50
labor	91.6	2.29	95.8	96.8	96.5	94.4	0.78	50
segment	92.3	0.97	94.6	96.7	96.4	93.2	0.28	50
sick	95.2	0.47	94.3	95.5	96.5	99.3	0.03	50
sonar	80.5	2.03	83.2	87.0	82.2	85.2	1.57	100
soybean	92.0	0.92	93.1	93.7	94.1	93.8	0.19	50
vehicle	74.7	0.48	79.3	80.3	81.0	76.4	1.12	50
Win-loss-tie	15-0-2		7-4-6	9-6-2	4-7-6			

data sets such as Glass and Labor data. Another factor to consider is training epochs. While the training epochs in GEFS were empirically determined for each data set, we used a uniform number of epochs on all data sets. In order to demonstrate this effect, we summarize the results with a varying number of epochs on the given data sets in Table 4.3. This simple parameter tuning improved the predictive accuracy on some data sets and the overall win-loss-tie summary.

We note that while it is generally a good idea to overfit the individual classifiers in an ensemble, we have not done so in the reported experiments, and may in fact be underfitting. As we can see from the comparison between Table 4.3 and Table 4.2,

on several data sets such as Sonar, Iris, Glass, and Ionospere, increasing the number of epochs by 50 resulted in significantly improved performance.

From the perspective of computational cost, our algorithm can be very slow compared to Bagging and Boosting. GEFS can be very slow compared to MEE because it uses at least twice as many as input features as used in MEE. In addition, the larger number of hidden nodes and longer training epochs can make GEFS extremely slow.

4.4.2 Experimental results of MEE/C4.5

In this section, we show the performance of MEE combined with C4.5 on the same data sets used in both [72] and [155]. This experiment aims to compare the performance of the two classifiers in the MEE framework. We used the same values for the ELSA parameters. All computational results including win-loss-tie summary are measured and reported in the same way as described in Section 4.4.1. In our experiments, unpruned trees are used to build ensembles in order to overfit each tree and thus promote the diversity. However, we use a pruned tree to estimate the predictive performance of single tree. We show our experimental results in Table 4.4. Bagging and AdaBoosting results were cited from [72], where 100 trees were used to build one ensemble.

In terms of the overall performance, MEE/C4.5 demonstrates better performance compared to single tree and Bagging, and comparable performance compared to Boosting. We also compare the results of MEE/C4.5 and MEE/ANN to see

Table 4.4: Results of MEE/C4.5.

	Single tree				MEE	
Dataset	Avg.	S.D.	Bagging	AdaBoost	Avg.	S.D.
credita	85.9	0.86	86.4	86.2	86.0	0.85
creditg	70.9	0.78	75.4	75.0	74.2	0.75
diabetes	74.9	0.84	75.6	74.3	74.7	0.69
glass	67.4	2.67	74.3	77.3	73.3	1.25
cleveland	74.0	1.39	79.1	78.3	79.2	0.23
hepatitis	81.3	2.14	82.5	83.7	83.2	2.28
votes-84	96.6	0.16	96.4	94.9	95.6	0.84
hypo	99.5	0.02	99.2	99.0	99.5	0.07
ionosphere	89.2	1.07	93.8	94.2	93.1	1.04
iris	94.4	1.12	95.0	95.0	94.0	1.41
krvskp	99.4	0.06	99.4	99.7	99.4	0.03
labor	79.3	1.47	88.7	86.9	86.7	4.04
segment	97.0	0.17	97.3	98.6	98.2	0.34
sick	98.6	0.12	97.9	97.9	98.6	0.11
sonar	72.8	1.82	75.7	81.0	79.8	2.74
soybean	92.6	0.42	87.8	93.2	93.4	0.01
vehicle	72.8	0.92	73.9	77.4	73.7	0.74
Win-loss-tie	9-1-7		5-2-10	4-5-8		

whether the performance of MEE is dependent on the type of classifier. In terms of predictive accuracy, MEE/C4.5 seems to be worse than MEE/ANN. This claim is supported by the observation that the win-loss-tie summary (7-7-3) of single ANN compared to single tree is different from the win-loss-tie summary (10-4-3) of MEE/ANN in Table 4.3 compared to MEE/C4.5. However, it warrants further investigation to see whether the feature selection for tree ensembles is not as effective as feature selection for neural networks ensembles and if so, why.

One clear advantage of MEE/C4.5 over MEE/ANN is its speed. MEE/C4.5 is much faster (roughly by a factor of 3 depending on the data dimension) than

MEE/ANN to evaluate a fixed number of solutions. Further if no pruning is performed, there is no parameter tuning with C4.5.

4.4.3 Guidelines toward optimized ensemble construction

In this section, we use MEE to examine ensemble characteristics and provide some guidelines for building optimal ensembles. We expect that by optimizing the ensemble construction process, MEE will in general achieve comparable accuracy to other methods using fewer individuals. We use data collected from the first fold of the first cross-validation routine for the following analyses.

We first investigate whether the ensemble size is positively related with the predictive accuracy. It has been well-established that to a certain degree, the predictive accuracy of an ensemble improves as the number of classifiers in the ensemble increases. Our result in Figure 4.3 indicates that accuracy improvements flatten out at an ensemble size of approximately 10–15, seeming to confirm the results in [156]. Note that some 95% confidence intervals (e.g., when ensemble is 13 or 31) are relatively wider than others. This happens when MEE explores through evolutionary process few ensembles with the same size but significantly different accuracy.

We also investigate whether the diversity among classifiers is positively related with the ensemble's classification performance. In our experiments, we measured the diversity in two different ways. The first measurement of diversity is based on the difference of predicted class between each classifier and the ensemble. We first define

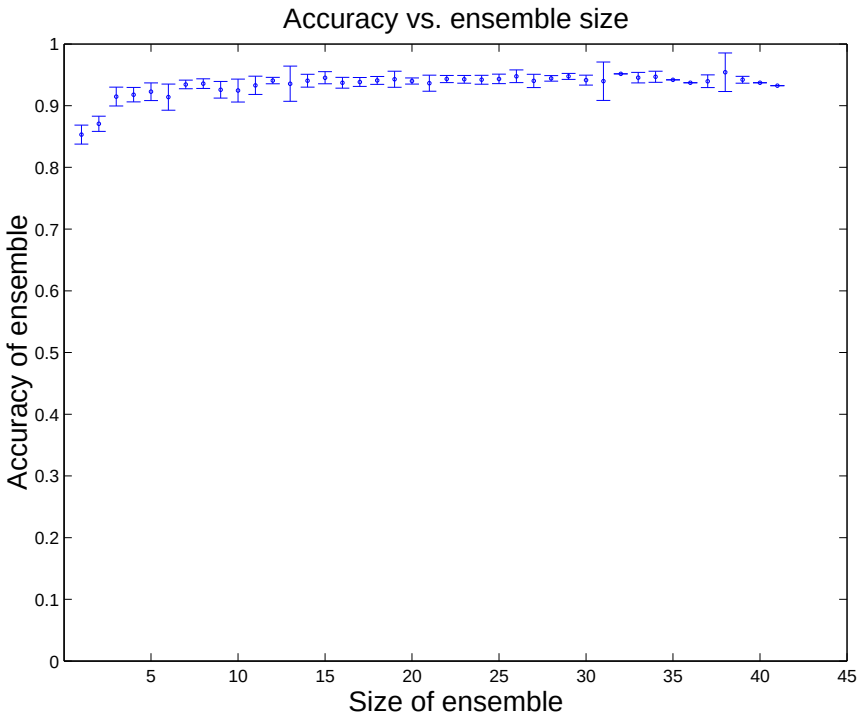

Figure 4.3: The relationship between the predictive accuracy and ensemble size with 95% confidence interval on the Soybean data. We observed similar patterns on other data sets.

a new operator $\oplus$ as follows:

$$\alpha \oplus \beta = \begin{cases} 0 & \text{if } \alpha = \beta \\ 1 & \text{otherwise} \end{cases} \tag{4.2}$$

When an ensemble e consists of g classifiers, the diversity of ensemble e, $diversity^e$, is defined as follows:

$$diversity^e = \frac{\sum_{i=1}^{g} \sum_{j=1}^{N} (pred_j^i \oplus pred_j^e)}{g \cdot N} \tag{4.3}$$

where N is the number of records in the test data and $pred_j^i$ and $pred_j^e$ represent

the predicted class label for record j by classifier i and ensemble e respectively. The larger the value of $diversity^e$, the more diverse ensemble is.

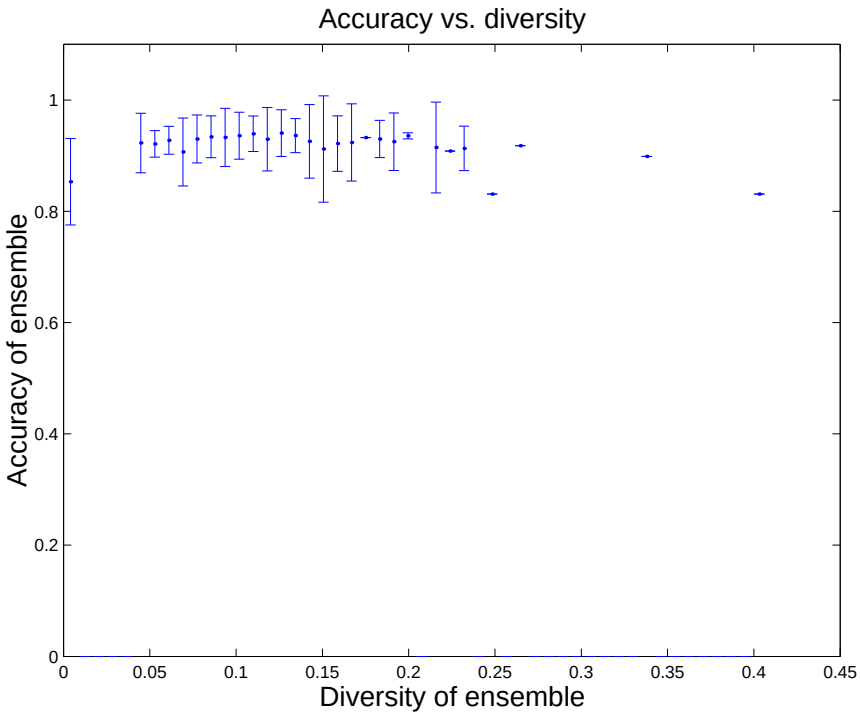

Figure 4.4: The relationship between the predictive accuracy and ensemble diversity with 95% confidence interval on the Soybean data. Similar patterns were observed on other data sets.

For the second diversity measurement, we slightly modified Yule's Q statistic [213]. According to [124], the Q statistic is the only measurement of diversity that satisfies the requirement of orthogonality between accuracy and diversity. In its original form, the Q statistic is used to compute the relationship between a pair of classifiers. When an ensemble consists of g classifiers, it is computed by taking

the averaged Q statistics of all possible pairs of classifiers. However, our simplified Q^* considers the relationship between an ensemble and its component classifiers in order to avoid lengthy computation. Table 4.5 shows the four possible relationships between the ensemble and a single classifier i prediction.

Table 4.5: Prediction relationships between the ensemble and a classifier.

		Classifier i	
		Correct	Incorrect
Ensemble	Correct	C^i_{11}	C^i_{10}
	Incorrect	C^i_{01}	C^i_{00}

For example, C^i_{11} is the number of records in the test set that both ensemble e and classifier i correctly classify. For notational convenience, we denote a, b, c, and d as the sums of C^i_{11}, C^i_{00}, C^i_{10}, C^i_{01} for all the classifiers that belongs to an ensemble e. Now we can define our $Q^*(e)$ as follows:

$$Q^*(e) = 1 - \frac{ad - bc}{ad + bc} \tag{4.4}$$

If there is no dependence at all between ensemble and classifier, $Q^*(e) = 1$. If there is only one classifiers in ensemble, $Q^*(e) = 0$ by definition.

We show the relationship between the predictive accuracy and two measurements of ensemble diversity in Figures 4.4 and 4.5. These two different measurements show the expected the positive relationship between accuracy and diversity. However, our results also show that too much diversity among classifiers can deteriorate en-

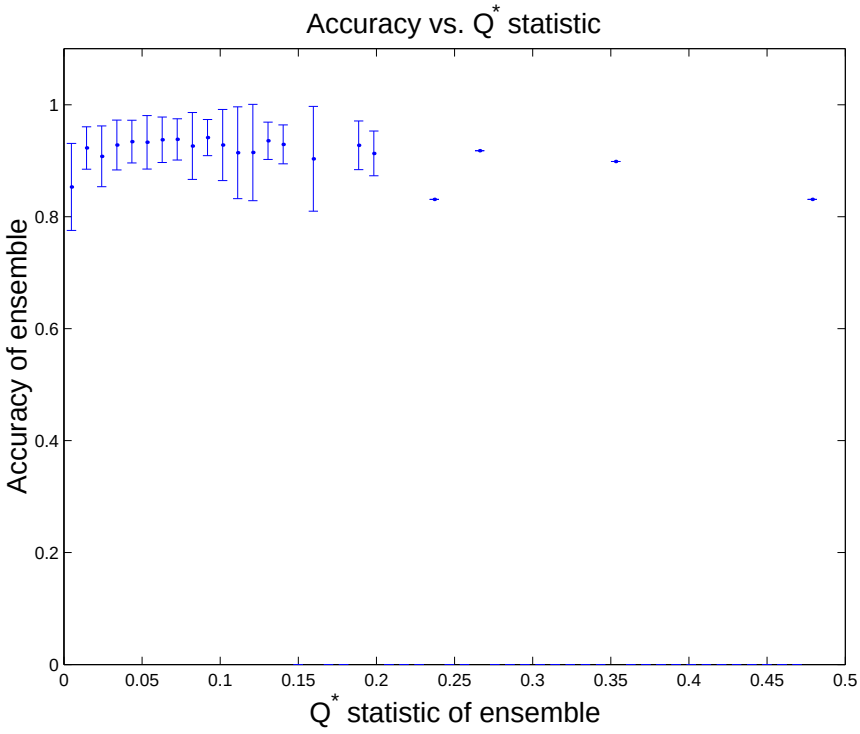

Figure 4.5: The relationship between the predictive accuracy and Q^* statistic with 95% confidence interval on Soybean data. We obtained similar results from other data sets.

semble performance, as the final decision made by ensemble becomes a random guess. Thus highly diverse ensembles are not well explored because MEE biases its search toward more promising ensembles.

4.5 Conclusions

In this chapter, we propose a new ensemble construction algorithm, Meta-Evolutionary Ensembles (MEE). This algorithm employs a novel two-level evolutionary search through the space of ensembles, using feature selection as the diversity mechanism. At the first level, individual classifiers compete against each other to

correctly predict held-out examples. Classifiers are rewarded for predicting difficult more points, relative to the other members of their respective ensembles. At the top level, the ensembles compete directly based on classification accuracy.

Our model has several nice properties. First of all, our experimental results indicate that this method shows very comparable classification accuracy while keeping the ensemble size small by optimizing it directly. The final solution shows consistently improved classification performance compared to a single classifier at the cost of computational complexity. Compared to the traditional ensembles (Bagging and Boosting) and GEFS, our resulting ensemble shows comparable performance while maintaining a smaller ensemble.

Our model also makes it possible to understand and analyze how and why ensemble methods achieve improved predictive accuracy. Our two-level evolutionary framework confirms that more diversity among classifiers can improve predictive accuracy. Up to a certain level, the ensemble size also has a positive effect on the ensemble performance. Further, our framework is a meta-search algorithm, meaning that it is independent of classifier types and/or various mechanism to promote diversity among classifiers. For example, we use feature selection as the mechanisms for individual diversity in this study. However, our flexible framework enables us to use data sampling (or both) to promote diversity among classifiers as in traditional ensemble methods. Our preliminary experiments show no big difference in overall performance between two methods for promoting diversity.

The next step is to compare this algorithm more rigorously to others on a larger collection of data sets, and perform any necessary performance tweaks on the EA energy allocation scheme. This new experiment is to verify Breiman's claim [36] based on experiments on synthetic data sets. He claimed that there is relatively little room for other types of ensemble construction algorithm to obtain further improvement because his decision forest method performs at or near the Bayes optimal level.

Along the way, we will examine the role of various characteristics of ensembles (size, diversity, etc.) and classifiers (type, number of dimensions / data points, etc.). By giving the system as many degrees of freedom as possible and observing the characteristics that lead to successful ensembles, we can directly optimize these characteristics and translate the results to a more scalable architecture [192] for large-scale predictive tasks. For example, we will implement a new framework, where the component classifiers move into a randomly chosen ensemble, rather than into the best-fit ensemble. This way we can measure the effect of ensemble size and diversity on the ensemble accuracy when the ensemble is a collection of randomly chosen classifiers. By directly optimizing ensembles, we expect that our MEE algorithm can construct smaller size of ensembles than a random method and the traditional ensemble algorithms.

Another direction of future research is to investigate whether oversearching affects the classification performance of our model. Through the evolutionary process, MEE evaluates a number of ensemble models and selects one as the final solution.

However, more extensive search can increase the probability of finding *fluke* rules that fit the data well but have low predictive accuracy [168]. We will investigate the relationship between the number of models and the predictive accuracy, particularly on data sets where MEE did not perform well.

CHAPTER 5
FEATURE SELECTION IN UNSUPERVISED LEARNING

5.1 Background

Feature selection has primarily been studied in a supervised learning context, where predictive accuracy is commonly used to evaluate feature subsets. Specifically, the data set could be divided into training and test sets, with the error rate on the test set used to estimate the true error rate of classifiers. However, in many situations we don't have information about the class to which each data point belongs, and thus we cannot apply supervised learning to estimate subset quality.

When we do not have prior information to evaluate candidate solutions, we instead wish to find natural groupings of the examples in the feature space via *clustering* or *unsupervised learning* and utilize the clustering results to evaluate solutions. The idea is to represent groups of points by a cluster prototype after determining the inherent number of clusters in the given data set. Once the clusters have been formed based on some given features, we must evaluate how well this model represents the complexity of the data.

The problem of determining an appropriate model in unsupervised learning has gained popularity in the machine learning, pattern recognition, and data mining communities. Unsupervised model selection addresses either how to identify the optimal number of clusters K or how to select feature subsets while determining the

correct number of clusters. The latter problem is more difficult because of the inter-dependency between the number of clusters and the feature subsets used to form the clusters [188]. To this point, most research on unsupervised model selection has considered the problem of identifying the right number of clusters using all available features [118, 188]. Other researches have studied feature selection and clustering together based on a heuristic sequential search with a single or unified criterion such as density threshold [1], cluster separability [65], or the category utility score [58].

The model we propose differs from other approaches in two main aspects of methodology: the evaluation of candidate solutions along multiple independent heuristic criteria, and the use of a local evolutionary algorithm to effectively cover the space of feature subsets and of cluster numbers.

First, we consider multiple fitness criteria simultaneously for evaluating clustering models. A number of heuristic criteria, such as cluster compactness, inter-cluster separation, and maximum likelihood have been proposed, and attempts have been made to combine some or all of these into a single objective [55, 73]. Previous research on unsupervised model selection [65, 58, 1] considered only one (single or combined) criterion. We claim that our approach is a generalization of such previous work, in the sense that it could capture both linear and non-linear relationships among the criteria.

From the perspective of knowledge discovery, our goal is to provide a clear picture of the (possibly nonlinear) tradeoffs among the various objectives. This is

important because no single criterion for unsupervised feature selection is best for every application [64] and only the decision maker can determine the relative weights of criteria for her application. In such situations we must use *multi-objective* or *Pareto* optimization.

Our goal in Pareto optimization is to approximate as best possible the Pareto front, presenting the decision maker with a set of high-quality compromise solutions from which to choose. Non-Pareto solutions will not be considered because they are inferior to those in the Pareto front by definition. By providing a set of alternative solutions to the decision maker, our approach helps her to choose the *right* solution at the right time. This could present a big advantage over other decision support systems that provide the decision maker with a single solution, given that she might not be familiar with how the algorithm reached such solution.

Secondly, as a search process, we turn to evolutionary algorithms (EAs) to intelligently search the space of possible feature subsets and to determine the appropriate number of clusters. Our choice of EAs as a search algorithm is reasonable because of their potential capability to search through spaces in a more global fashion than many other machine learning algorithms. EAs have also been used for clustering, using an adjacency-based representation [160] or in conjunction with other algorithms [106, 82].

This chapter is organized as follows. We motivate our approach by illustrating possible application areas in Section 5.2. In Section 5.3 we review the K-means clus-

tering algorithm and heuristic metrics to evaluate the quality of clusters constructed by K-means. In Section 5.4 we present the EM algorithm and justify our clustering quality metrics. We discuss our approach in detail in Section 5.5, illustrating the evolutionary algorithm and describing how ELSA is combined with K-means or EM. Sections 5.6 and 5.7 present some experimental results with a synthetic data set and a real data set, and discuss the interpretation of the ELSA output to select a subset of good features. Finally section 5.8 addresses directions of future research and concludes this chapter.

5.2 Customer segmentation

Feature selection in unsupervised learning can be very useful for enhancing customer relationship management (CRM) for market managers and finding critical but unnoticed patterns for financial analysts. For example, in the marketing research community, clustering and its variants [178, 206], neural networks [14] and conjoint analysis [78] have been widely used for market structure analysis and market segmentation. It is well known that manufacturers use different marketing strategies based on customer behavior such as brand loyalty, price sensitivity, or quality sensitivity. Furthermore, they can save time and expense by restricting their concern to a group of customers who are most likely to buy their goods.

Standard application of cluster analysis uses the complete set of features or a pre-selected subset of features. For instance, a market survey typically contains various types of questions with regard to respondents' demographic and psychographic

information, attitudes toward products and benefits sought. Commonly, separate clustering analyses are implemented to find respondent segments and decide the number of segments based on each different type of variable. A market manager might choose to cluster using only customer demographic variables, in order to offer different campaign options to different customer segments based on their age, sex, education level or income level. Or, the manager might consider customer responses to changes in price, display style, or advertisements to define market segments. Therefore clustering analysis has been implemented in a top-down fashion, dependent on the prior knowledge of market managers who pre-determine the features to be used to segment customers.

However, this top-down approach could not find and exploit interactions among various types of features on segments. Further, some segments can be discovered only if different types of variables are considered together. Therefore, the utility of such a top-down approach is limited from the perspective of knowledge discovery, because it cannot provide new marketing models that could be effective but have not been considered. Our data-driven approach remedies this limitation by searching the space of models, varying the feature subsets and the number of clusters. This way we can present the decision maker with a set of high-quality solutions from which to choose.

As an example, consider the application of our approach to datasets like those collected by insurance companies, containing customer information on both socio-demographic characteristics and ownership of various types of insurance policies (see,

for instance, the CoIL data sets [105].) When insurers try to identify customers that are likely to buy a new policy, they consider only a few models dependent on the prior knowledge and past experience of the market managers. Our data-driven approach searches a much broader space of models and provides a compact summary of solutions over possible feature subset sizes and numbers of clusters. Among such high-quality solutions, the manager can select a specific model after considering the model's complexity and accuracy. Further, newly-discovered feature subsets that form well-differentiated clusters can affect the way new marketing campaigns should be implemented. Let us suppose that an insurance company uses our data-driven approach to campaign a new recreational vehicle policy. Let us also assume that our model selects as a solution a set of features including ownership of moped and car policies. The market manager notes that moped policy features are included in a final solution, even though she has never used this information before to identify customer segments. However, further investigation reveals that many people who purchase a moped policy might also purchase a recreational vehicle policy because they often carry their mopeds or bicycles on the back of their vehicles [107].

Similarly, our approach can be useful for the analysis of finance and accounting data. For instance, forecasting corporate bankruptcy has been studied extensively in the accounting, economics, and finance community [10]. However, we are more interested in finding common and unknown factors that affect the financial structure and eventually lead companies to go bankrupt. Our approach provides a number of

clustering results from different sets of selected features. If, say, profitability-related features form well-separated clusters in terms of how soon companies go bankrupt, credit analysts can build a model that predicts bankruptcy time more accurately. Or, if clustering analysis reveals that market-driven variables such as market size have serious effects on the performance of the small-sized companies, forecasters of corporate mergers should pay more attention to the changes in these variables than in other variables.

5.3 K-means algorithm

5.3.1 Algorithm detail

K-means is one of the most often used nonhierarchical clustering methods [73, 27]. Nonhierarchical clustering algorithms are designed to group items into a collection of K clusters that can be specified in advance or determined as part of the clustering procedure. Nonhierarchical methods start from either an initial partition of items into groups or an initial set of seed points, which form the centroid or medoid[1] of clusters.

The K-means algorithm employs a squared error criterion and implicitly assumes that clusters are represented by spherical Gaussian distributions located at the K cluster means [26]. Starting with a random initial partition, it iteratively assigns each data point to the cluster whose centroid is located nearest to the given point,

[1] A medoid is the most centrally located point in a cluster.

```
assign each data point to a randomly chosen cluster
calculate the centroid γ_k of each cluster k
do
   for each point x_n, n ∈ {1, ···, N}
      move x_n to nearest cluster arg min_k distance(x_n, γ_k)
   endfor
   for each cluster k with changed membership
      update γ_k
   endfor
while at least one point changed cluster assignment
```

Figure 5.1: K-means clustering algorithm.

and recalculates the centroids based on the new set of assignments until a convergence criterion is met. 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 [120, 12]. However, in our study we use the standard K-means algorithm [100] as summarized in Figure 5.1.

5.3.2 Heuristic metrics for clustering evaluation

The most difficult part of unsupervised learning is how to measure the fitness of each solution. A number of numerical measurements are available to evaluate clustering quality [84, 55]. 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 [24]. 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 co-

hesiveness. It is inspired by the total within-cluster sum of squares (TWSS) measure. Formally, let $x_n, n = 1, \cdots, N$, be data points and x_{nj} be the value of the j-th feature of x_n. 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_{nk} as follows:

$$\alpha_{nk} = \begin{cases} 1 & \text{if } x_n \text{ belongs to cluster } k \\ 0 & \text{otherwise} \end{cases} \tag{5.1}$$

where $k = 1, \cdots, K$ and $n = 1, \cdots, N$.

The centroid of the k-th cluster, γ_k, can be defined by its coordinates:

$$\gamma_{kj} = \frac{\sum_{n=1}^{N} \alpha_{nk} x_{nj}}{\sum_{n=1}^{N} \alpha_{nk}},\ j \in J. \tag{5.2}$$

F_{within} can be computed as follows:

$$F_{within} = 1 - \frac{1}{Z_{within}} \frac{1}{d} \sum_{k=1}^{K} \sum_{n=1}^{N} \alpha_{nk} \sum_{j \in J} (x_{nj} - \gamma_{kj})^2 \tag{5.3}$$

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:

$$F_{between} = \frac{1}{Z_{between}} \frac{1}{d} \frac{1}{K-1} \sum_{k=1}^{K} \sum_{n=1}^{N} (1 - \alpha_{nk}) \sum_{j \in J} (x_{nj} - \gamma_{kj})^2 \tag{5.4}$$

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 than can be justified by the data, and thus less generalizable. Therefore, other things being equal, we want fewer clusters:

$$F_{clusters} = 1 - \frac{K - K_{min}}{K_{max} - K_{min}} \tag{5.5}$$

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:

$$F_{complexity} = 1 - \frac{d-1}{D-1}. \tag{5.6}$$

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

5.4 EM algorithm for mixture models

5.4.1 Algorithm detail

The expectation maximization algorithm [57] is based on the well-established theory of probability and is one of the most often used statistical modeling algorithms [48, 76]. The EM algorithm often significantly outperforms other clustering methods [138] and is superior to the distance-based algorithms (e.g. K-means) in the sense that it can handle categorical data. The EM algorithm for mixture models assumes that the patterns are drawn from one of several given distributions, and the goal is to identify the parameters of each distribution. In the EM framework, the parameters of the clusters are unknown, and these are estimated from the given data.

The EM algorithm starts with an initial estimate of the parameters and iteratively recomputes the likelihood that each pattern is drawn from a particular density function, and then updates the parameter estimates. Formally, let $x_n, n = 1, \cdots, N$, be a data point and x_{nj} be the value of the j-th feature of x_n. Let d be the dimension of the *selected* feature set, J, and K be the number of clusters. If we model each cluster with a d-dimensional Gaussian distribution, we can approximate the data distribution by fitting K density functions c_k, $k = 1, \cdots, K$, to the data set $\{x_n | n = 1, \cdots, N\}$. The probability density function evaluated at x_n is the sum of all densities:

$$P(x_n) = \sum_{k=1}^{K} p_k \cdot c_k(x_n|\theta_k) \tag{5.7}$$

where the *a priori* probability p_k is the fraction of the data points in cluster k and

$\sum_{k=1}^{K} p_k = 1,\ p_k \geq 0$. The functions $c_k(x_n|\theta_k)$ are the density functions for patterns of the cluster k and θ_k represents the parameters of the density function. For Gaussian distributions, the parameters are the mean μ_k and covariance matrix Σ_k. For greater efficiency and reduced overfitting, we ignore cross terms and represent Σ_k as a vector of the variances for each dimension. The membership probability of pattern x_n in cluster k is computed as follows:

$$p_k(x_n) = \frac{p_k \cdot c_k(x_n|\theta_k)}{\sum_{i=1}^{K} p_i \cdot c_i(x_n|\theta_i)}. \tag{5.8}$$

Now, the original problem of finding clusters is reduced to the problem of how to estimate the parameters $\Theta = \{\theta_1, \cdots, \theta_K\}$ of the probability density [39]. Under the independence assumption among attributes within a given cluster, we can represent each density function as a product of density functions over each selected attribute $j = 1, \cdots, d$:

$$c_k(x_n|\theta_k) = \prod_{j \in J} c_{kj}(x_{nj}|\theta_{kj}) \tag{5.9}$$

where θ_{kj} represents the parameters of the j-th feature of cluster k.

Finally, the multivariate Gaussian distribution for cluster $k = 1, \cdots, K$ is parameterized as follows:

$$c_k(x_n|\mu_k, \Sigma_k) = \prod_{j \in J} \frac{1}{\sqrt{2\ \pi\ \sigma_{kj}^2}} \exp\left(\frac{(x_{nj} - \mu_{kj})^2}{-2\ \sigma_{kj}^2}\right) \tag{5.10}$$

where μ_{kj} and σ_{kj}^2 represent the mean and variance of the j-th feature of cluster k, respectively.[2] Now we can quantify the quality of a given set of parameters Θ

[2]Since small values of σ_{kj}^2 can cause overflow in our computations, we set a lower bound value of σ_{kj}^2 to 10^{-10}.

$t = 0$
initialize p_k^t, μ_k^t, and Σ_k^t for each cluster $k \in \{1, \cdots, K\}$
compute $memberProb(t)$
compute $L(\Theta^t)$
do
  for each cluster $k \in \{1, \cdots, K\}$

$$p_k^{t+1} = \sum_{n=1}^{N} p_k^t(x_n)$$

$$\mu_k^{t+1} = \frac{\sum_{n=1}^{N} p_k^t(x_n) \cdot x_n}{\sum_{n=1}^{N} p_k^t(x_n)}$$

$$\Sigma_k^{t+1} = \frac{\sum_{n=1}^{N} p_k^t(x_n)(x_n - \mu_k^{t+1})(x_n - \mu_k^{t+1})^T}{\sum_{n=1}^{N} p_k^t(x_n)}$$

  endfor
  compute $memberProb(t+1)$
  compute $L(\Theta^{t+1})$
  $t = t + 1$
while $|L(\Theta^t) - L(\Theta^{t-1})| > \epsilon$ and $t < maxIteration$

compute $memberProb(t)$
{
  for each pattern x_n, $n \in \{1, \cdots, N\}$

$$p_k^t(x_n) = \frac{p_k^t \cdot c_k(x_n|\mu_k^t, \Sigma_k^t)}{\sum_{i=1}^{K} p_i^t \cdot c_i(x_n|\mu_i^t, \Sigma_i^t)}$$

  endfor
}

Figure 5.2: Summary of the EM algorithm where $\epsilon > 0$ is a stopping tolerance and p_k^t, μ_k^t, and Σ_k^t represent the mixture model parameters of cluster k at iteration t. In our implementation, we set $\epsilon = 1.0$ and $maxIteration = 15$ for fast convergence.

using the Equation (5.10). At this point, our only problem is to find the mixture parameters μ_k and Σ_k along with p_k. The maximum likelihood (ML) method [63] is used to maximize the probability of the data set given a particular mixture model, and often the log-likelihood is maximized for analytical purposes as follows:

$$L(\Theta) = \sum_{n=1}^{N} \log P(x_n) = \sum_{n=1}^{N} \log \left(\sum_{k=1}^{K} p_k \cdot c_k(x_n|\mu_k, \Sigma_k) \right). \tag{5.11}$$

The EM algorithm begins with an initial estimation of Θ and iteratively updates it in such a way that the sequence of $L(\Theta)$ is non-decreasing. In our implementation, EM iterates until $|L(\Theta^{t+1}) - L(\Theta^t)| \leq \epsilon$, $\epsilon > 0$ or up to $maxIteration$ iterations.

We choose the somewhat loose convergence criteria, $\epsilon = 1.0$ and $maxIteration = 15$ because the marginal likelihood gain per additional computing resource for more restrictive criteria is negligible. We outline the standard EM algorithm in Figure 5.2.

5.4.2 Heuristic metrics for clustering

In order to evaluate the quality of the clusters formed by the EM algorithm, we use three heuristic fitness criteria, described below. One of the criteria is inspired by statistical metrics and two by Occam's razor. Each objective is again normalized into the unit interval and maximized by the EA.

$F_{accuracy}$: This objective is meant to favor cluster models with parameters whose corresponding likelihood of the data given the model is higher. With estimated distribution parameters μ_k and Σ_k, $F_{accuracy}$ is computed as follows:

$$F_{accuracy} = \frac{1}{Z_{accuracy}} \sum_{n=1}^{N} \log \left(\sum_{k=1}^{K} p_k \cdot c_k(x_n | \mu_k, \Sigma_k) \right) \tag{5.12}$$

where $Z_{accuracy}$ is an empirically derived, data-dependent normalization constant meant to achieve $F_{accuracy}$ values spanning the unit interval.

$F_{clusters}$: This criterion is defined in the same way as in Section 5.3 (Equation (5.5)).

$F_{complexity}$: This is another criterion defined as in Section 5.3 (Equation (5.6)).

5.5 Evolutionary Local Selection Algorithm

We first show the wrapper model of ELSA with clustering algorithms in Figure 5.3. In our proposed wrapper model, there are three relevant spaces: *search space*,

data space, and *objective space*. In search space, ELSA searches the space of feature subsets and number of clusters K. Once a specific feature subset (e.g. c) and number of clusters (e.g. $K = 3$) is selected, this information is encoded into a chromosome of an agent. In data space, three clusters are formed via K-means or EM. In objective space, clusters are evaluated in terms of the evaluation criteria and an agent is rewarded energy from each objective based on its fitness and the local environment to which it belongs. Each agent will survive, reproduce, or die depending on its energy level, and ELSA biases its search in the direction of high energy levels. This process is repeated for a fixed number of iterations, T.

We outline the ELSA algorithm in Figure 5.4. 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.[3] This representation is motivated by the desire to preserve the regularity of the number of clusters under the genetic operators: changing any one bit will change K by one.

Mutation and crossover operators are used to explore the search space. The mutation operator randomly selects one bit of an agent and flips it. Our crossover operator follows the commonality-based crossover framework [49]. It takes two agents,

[3]The cases of zero or one cluster are meaningless, therefore we count the number of clusters as $K = \kappa + 2$ where κ is the number of ones and $K_{min} = 2 \leq K \leq K_{max}$.

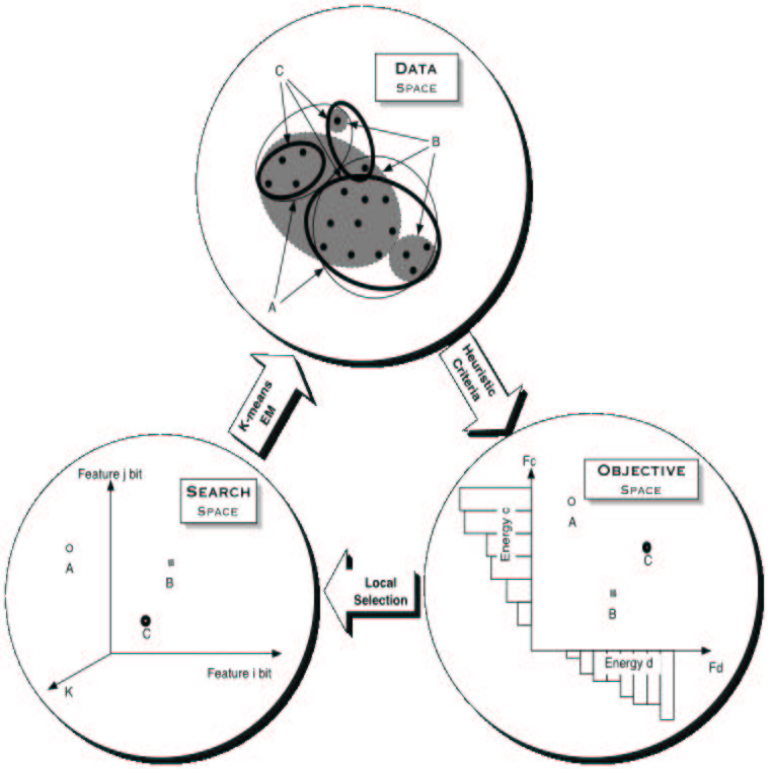

Figure 5.3: The wrapper model of ELSA with clustering algorithms.

a parent a and a random mate, and scans through every bit of the two agents. If it locates a different bit, it flips a coin to determine the offspring's bit. In this process, the mate contributes only to construct the offspring's bit string, which inherits all the common features of the parents.

For the search and evaluation routine, please refer to Section 2.5.2.3. For the selection and energy replenishment part of the algorithm, please refer to Section 2.5.2.2. In order to assign energy to a solution, ELSA must be informed of clustering quality. In the experiments described here, the clusters to be evaluated are

```
initialize p_max agents, each with energy η/2
while there are alive agents in Pop^g and t < T
   Replenishment()
   for each agent a in Pop^g
      Search & Evaluation()
      Selection()
      t = t + 1
   endfor
   g = g + 1
endwhile

Replenishment()
{
   for each energy source c ∈ {1, ···, C}
      for each v ∈ {1/B, 2/B, ···, 1} where B is number of bins
         E^c_envt(v) ← 2v E^c_tot
      endfor
   endfor
}

Search & Evaluation()
{
   a' ← mutate(crossover(a, randommate))
   for each energy source c ∈ {1, ···, C}
      v ← Fitness(a')
      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
}

Selection()
{
   if (E_a > η)
      insert a, a' into Pop^{g+1}
      E_a' ← E_a/2
      E_a ← E_a − E_a'
   else if (E_a > 0)
      insert a into Pop^{g+1}
   endif
}
```

Figure 5.4: ELSA pseudo-code. In each iteration, the environment is replenished and then each alive agent executes the main loop. The program terminates after T solutions (agents) are evaluated. The details of the algorithm are illustrated in the text.

constructed based on the selected features using a standard K-means or EM algorithm (cf. Figure 5.3). 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 clustering algorithm is given the projection of the data set onto J, uses it to form K clusters, and returns the fitness values.

5.6 Experiments on synthetic data set

5.6.1 Data description and baseline algorithm

It is difficult to evaluate the quality of an unsupervised learning algorithm, and 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 from different subsets. In order to evaluate our approach, we construct a moderate-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. We evaluate the evolved solutions by their ability to discover five pre-constructed clusters in a ten-dimensional subspace.

The data set has $N = 500$ points and $D = 30$ features. The feature set consists of "significant" features, "Gaussian noise" features, and "white noise" features. It is constructed so that the first 10 features are significant, with 5 "true" normal 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 di-

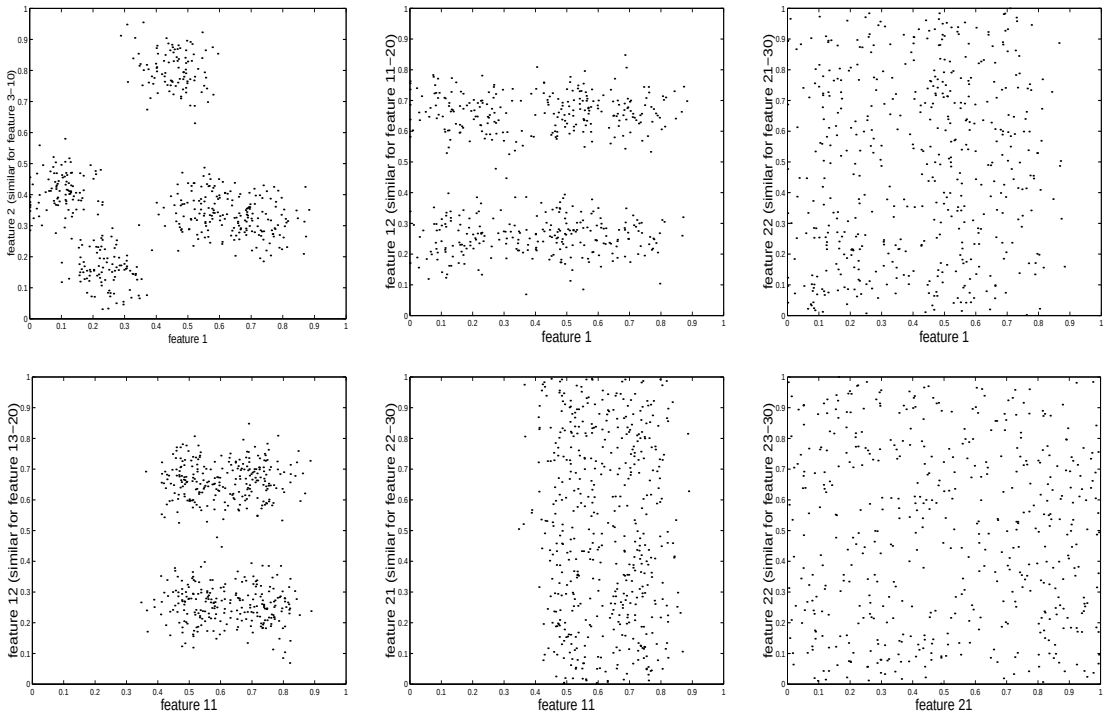

Figure 5.5: A few 2-dimensional projections of the synthetic data set.

mensions. The remaining 10 features are white noise in which points are drawn from uniform distributions. 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. We present some 2-dimensional projections of the synthetic data set in Figure 5.5.

For further comparisons we have implemented a greedy heuristic algorithm known as the *plus 2-take away 1 sequential selection* algorithm [111]. This is a reasonable choice for a comparative algorithm because we want our algorithm to outperform most commercial statistical programs (e.g. SAS and SPSS) that implement

simpler search algorithms, such as sequential forward and backward selection, for feature selection. Since the greedy algorithm we have implemented allows limited backtracking, it performs better than feature selection algorithms typically used in commercial programs. Our implementation of this algorithm for clustering requires a set value of K and uses F_{within} and $F_{between}$ for K-means, and $F_{accuracy}$ for EM as the optimization criteria. It begins by finding the single dimension along which the objective is optimized. At each successive step, the algorithm adds an additional 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. We ran the algorithm for each of the values of K considered by ELSA.

Individuals are represented by 36 bits, 30 for the features and 6 for K ($K_{max} = 8$). There are 15 energy bins for all energy sources, $F_{clusters}$, $F_{complexity}$, F_{within}, $F_{between}$, and $F_{accuracy}$. The values for the various ELSA parameters are: $\Pr(mutation) = 1.0$, $\Pr(crossover) = 0.8$, $p_{max} = 100$, $E_{cost} = 0.2$, $E_{total} = 40$, $\eta = 0.3$, and $T =$ 30,000.

5.6.2 Results using K-means

We first show two different types of Pareto approximation evolved by ELSA/K-means in Figure 5.6, one based on F_{within} and the other one based on $F_{between}$, in order to observe the usefulness of our two clustering quality metrics. Recall that both are used in ELSA to evaluate the quality of clusters. We use the term *candidate front* for

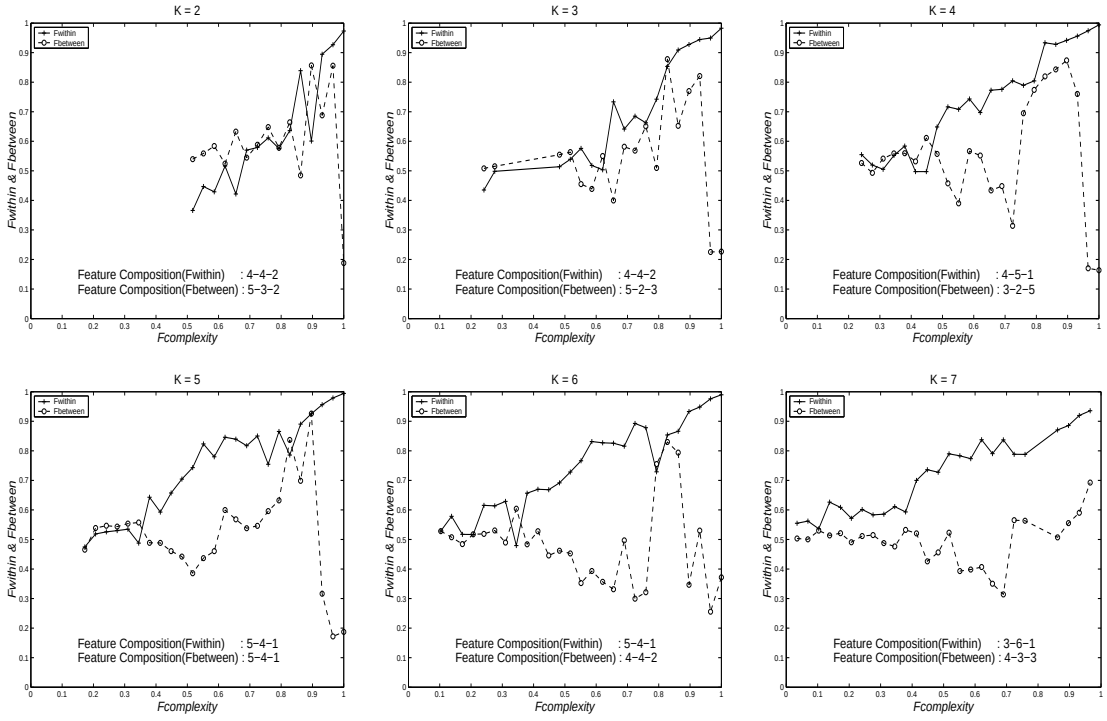

Figure 5.6: The ELSA/K-means fronts with composition of features selected for $F_{complexity}$ corresponding to 10 features (see text). We omit the candidate fronts of $K = 8$ because of its incomplete coverage of the search space.

the set of solutions with the highest measured clustering quality at every $F_{complexity}$ value for each K. We construct candidate fronts based on all solutions evaluated during the evolutionary process with two different clustering quality measures, F_{within} and $F_{between}$. In order to show the candidate fronts of each different number of clusters K, we sort all the non-dominated solutions by K.

We expect the candidate front based on F_{within} for any reasonable K to typically decrease from higher values of $F_{complexity}$ (lower complexity) to lower values of $F_{complexity}$ (higher complexity). This is because we normalize F_{within} by the number of

selected features d. Selecting more features make it more likely to select less relevant features, deteriorating the clustering quality. The fronts based on F_{within} in Figure 5.6 show the trend that we expect. The clustering quality in terms of F_{within} improves as the number of clusters approaches the true number of clusters, $K = 5$. In particular, the fronts for $K = 5$ and $K = 6$ not only explore most $F_{complexity}$ values but also show high clustering quality. A decision maker would determine the correct number of clusters to be either 5 or 6.

The candidate fronts based on $F_{between}$ are less stable than those based on F_{within}. We attribute this to the fact that $F_{between}$ is more sensitive to outliers than F_{within}. $F_{between}$ is affected explicitly by both d and K in its computation, while F_{within} is affected explicitly by d but implicitly by K via clustering quality. However, the fronts become stable with more than half of features selected because many features neutralize the effects of outliers from certain features. Although we feel that $F_{between}$ captures useful information about the quality of the clusters, its instability makes it inappropriate as a single metric to determine the best solution to be presented to a decision maker.

We also show in Figure 5.6 the composition of selected features, i.e., the number of significant-Gaussian noise-white noise features selected at $F_{complexity} = 0.69$ (10 features). Note that the selected features at this value of $F_{complexity}$ are not necessarily all the "significant" features that we constructed. We attribute this finding to the fact that if one or more Gaussian noise features form good clusters with the previously

selected significant features, the clustering quality can be improved by adding these features. This is also consistent with the notion that not all strongly relevant features are selected and some weakly relevant features could be selected as "relevant" features [113]. Though even mixes of significant and Gaussian features are selected at $F_{complexity} = 0.69$, ELSA/K-means found a better composition of selected features at values of $F_{complexity}$ near 0.69. For example, the composition of selected features for $K = 5$ based on F_{within} were 8-3-1, 7-3-1, 5-4-1, and 6-3-0 over $0.62 \leq F_{complexity} \leq 0.73$ (9–12 features), respectively.

Figure 5.7 shows snapshots of the candidate fronts with $K = 5$ based on F_{within} at intervals of every 3,000 solution evaluations. It is evident that ELSA/K-means identifies better solutions and explores an increasingly broad space of feature subsets as it evaluates more solutions.[4] We show the improvement of the candidate fronts by computing the $coverage_{KM}$ as follows:

$$coverage_{KM} = \sum_{i \in F_{complexity}} F^i_{within} \tag{5.13}$$

where F^i_{within} is the F_{within} value at $F_{complexity} = i$. As ELSA finds new and better solutions (with higher F_{within}), the coverage increases.

We finally evaluated ELSA/K-means in terms of classification accuracy. We compute accuracy by assigning a class label to each cluster based on the majority class of the points contained in the cluster, and then computing correctness on *only those classes*, e.g., models with only two clusters are graded on their ability to find

[4]Similar results were obtained for different number of clusters K and for $F_{between}$.

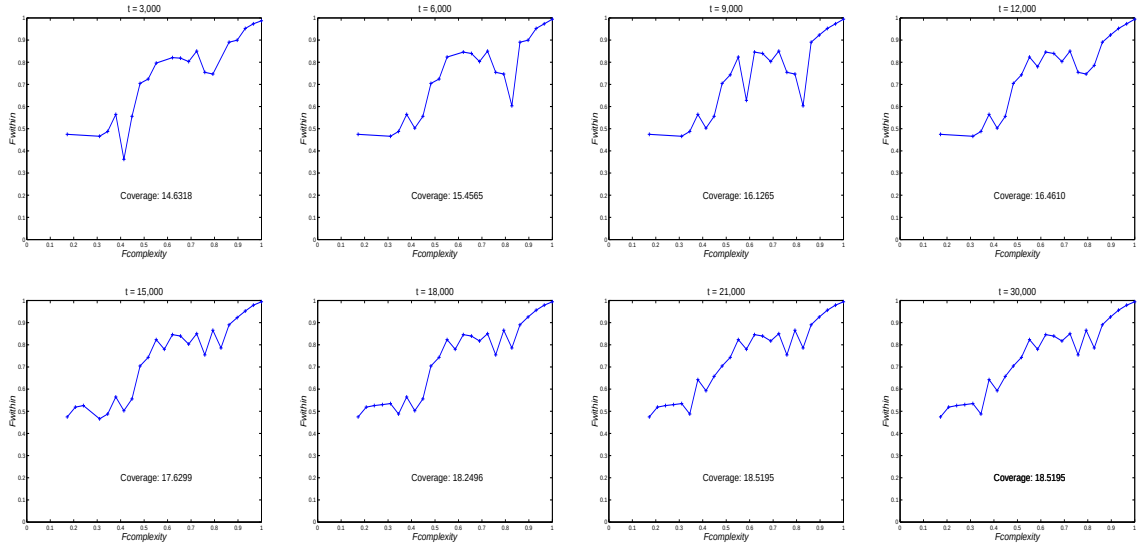

Figure 5.7: The candidate fronts for $K = 5$ based on F_{within} evolved in ELSA/K-means. It is captured every 3,000 evaluated solutions. There is no further improvement in coverage after the 7th interval.

two classes. ELSA results represent individuals chosen from fronts based on $\widetilde{F}_{accuracy} = F_{within} \cdot F_{between}$.[5] This criterion is based on the fact that neither F_{within} nor $F_{between}$ truly represents the quality of the clusters. The classification accuracy of candidate solutions based on either F_{within} or $F_{between}$ was inferior to that based on $\widetilde{F}_{accuracy}$. Table 5.1 shows the classification accuracy with standard error of various models formed by both ELSA/K-means and the greedy feature search. ELSA/K-means results represent individuals with less than eight features from the candidate fronts. All accuracy measures are averaged over five different runs of ELSA/K-means

[5]This new measurement is used only for selecting a final solution. In ELSA, F_{within} and $F_{between}$ are considered separately.

and of the greedy search.

The overall performance of ELSA/K-means is superior to that of the greedy search on models with few features and few clusters — exactly the sort of models the algorithm was designed to find. The last row and column shows the number of win-loss-tie cases of ELSA/K-means compared with greedy search. A tie is assumed when error bars overlap. The performance of ELSA/K-means for $d = 3$ across different K is slightly inferior, although the difference is small for $K = 3$ and $K = 5$. For more complex models with more than 10 selected features (not shown), the greedy method is often better able to reconstruct the original classes. This is reasonable, since ELSA by design does not concentrate on this part of the search space.

Table 5.1: The classification accuracy of ELSA/K-means and greedy.

K		Number of selected features						
		2	3	4	5	6	7	W-L-T
2	ELSA/KM	100±0.0	100±0.0	100±0.0	100±0.0	59±0.0	100±0.0	5-0-1
	Greedy	59±0.0	59±0.0	59±0.0	59±0.0	59±0.0	59±0.0	
3	ELSA/KM	93.2±5.2	39.4±0.4	98.6±1.4	100±0.0	100±0.0	100±0.0	3-1-2
	Greedy	40.6±0.3	40.8±0.2	40.2±0.2	63.6±3.9	100±0.0	100±0.0	
4	ELSA/KM	85.2±4.1	31.4±0.3	92±4.9	100±0.0	100±0.0	100±0.0	5-1-0
	Greedy	30.8±0.2	55±0.0	55±0.0	55±0.0	55±0.0	55±0.0	
5	ELSA/KM	62±0.6	48.4±1.5	75±2.1	58.4±1.9	63.4±0.4	79.4±0.6	4-1-1
	Greedy	25.6±0.3	53.4±0.6	53.4±0.6	55±0.0	63±0.0	66.4±3.4	
W-L-T		4-0-0	1-3-0	4-0-0	4-0-0	1-0-3	3-0-1	17-3-4

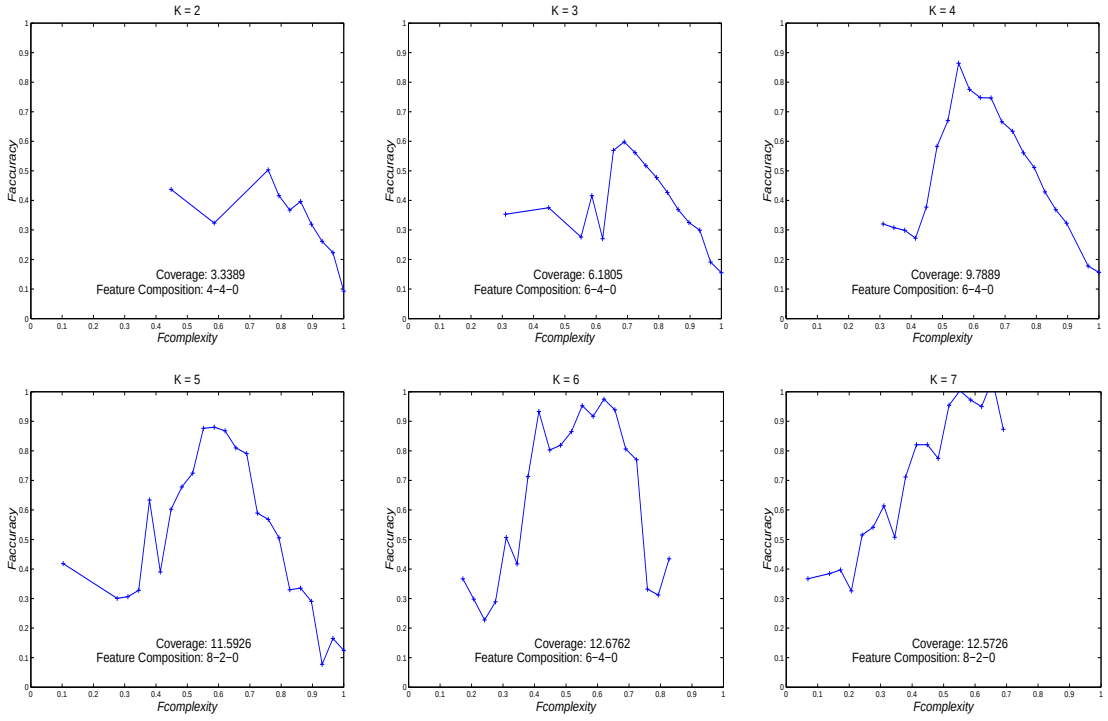

Figure 5.8: The candidate fronts of ELSA/EM model. We omit the candidate front for $K = 8$ because of its inferiority in terms of clustering quality and incomplete coverage of the search space. Composition of selected features is shown for $F_{complexity}$ corresponding to 10 features (see text).

5.6.3 Results using EM

We show the candidate fronts found by the ELSA/EM algorithm for each different number of clusters K in Figure 5.8. In contrast with the ELSA/K-means model, we have a single measurement of clustering quality $F_{accuracy}$ in ELSA/EM. We did the same analysis to see whether our ELSA/EM model is able to identify the correct number of clusters based on the shape of the candidate fronts across different values of K and $F_{accuracy}$. A different characteristic shape of the Pareto fronts is observed in ELSA/EM because of the different measurement of clustering quality: an

ascent in the range of higher values of $F_{complexity}$ (lower complexity), and a descent for lower values of $F_{complexity}$ (higher complexity). This is reasonable because adding additional significant features will have a good effect on the the clustering quality with few previously selected features. However, adding noise features will have a negative effect on clustering quality in the probabilistic model, which, unlike Euclidean distance, is not affected by dimensionality. Hence the curve forms a shape similar to the supervised learning curve, with a global maximum indicating the optimal number of features. The coverage of the ELSA/EM model shown in Figure 5.8 is defined as:

$$coverage_{EM} = \sum_{i \in F_{complexity}} F^i_{accuracy} \tag{5.14}$$

We note that the clustering quality and the search space coverage improve as the evolved number of clusters approaches the "true" number of clusters, $K = 5$. The candidate front for $K = 5$ not only shows the typical shape we expect but also an overall improvement in clustering quality. The other fronts do not cover comparable ranges of the feature space either because of the agents' low $F_{clusters}$ ($K = 7$) or because of the agents' low $F_{accuracy}$ and $F_{complexity}$ ($K = 2$ and $K = 3$). A decision maker again would conclude the right number of clusters to be 5 or 6.

As noticed in ELSA/K-means, the first 10 selected features, $0.69 \leq F_{complexity} \leq 1$, are not all significant. This notion is again quantified through the number of significant-Gaussian noise-white noise features selected at $F_{complexity} = 0.69$ (10 features) in Figure 5.8.[6] None of the "white noise" features is selected and the overall

[6]For $K = 2$, we use $F_{complexity} = 0.76$, which is the closest value to 0.69 represented in

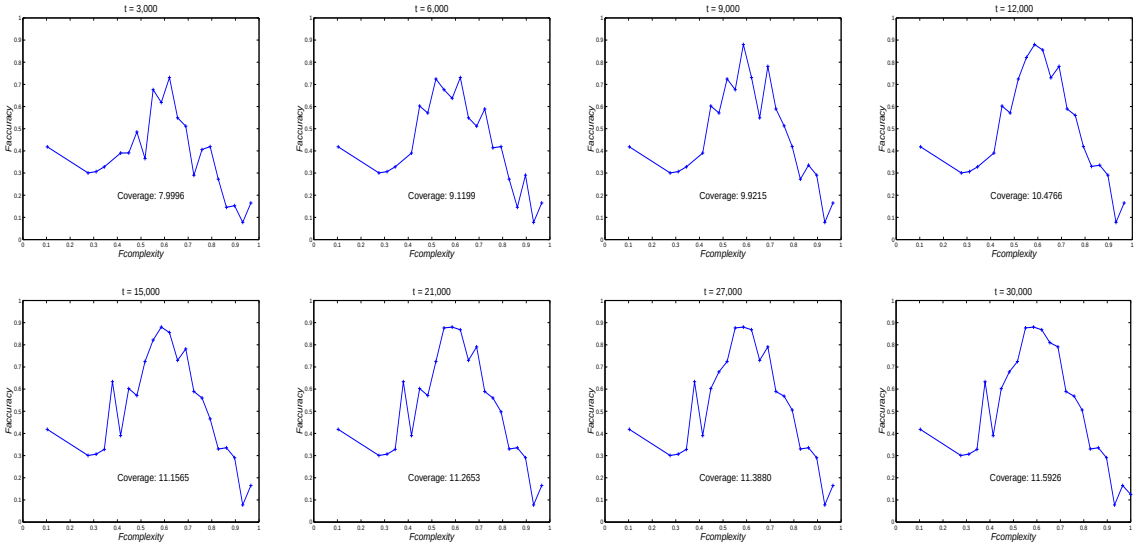

Figure 5.9: Candidate fronts for $K = 5$ based on $F_{accuracy}$ evolved in ELSA/EM. It is captured at every 3,000 solution evaluations and two fronts (t = 18,000 and t = 24,000) are omitted because they have the same shape as the ones at t = 15,000 and t = 21,000, respectively.

composition of selected features is better in ELSA/EM than in ELSA/K-means.

We also show snapshots of the ELSA/EM fronts for $K = 5$ at every 3,000 solution evaluations in Figure 5.9. Similarly to the ELSA/K-means model, ELSA/EM explores a broad subset of the search space, and thus identifies better solutions across $F_{complexity}$ as more solutions are evaluated. We observed similar results for different number of clusters K.

Table 5.2 shows classification accuracy of various models formed by both ELSA/EM and the greedy feature search. We compute accuracy in the same way

---

the front.

that we did in ELSA/K-means. ELSA results represent individuals selected from candidate fronts with less than eight features. The "–" entry indicates that no solution is found by ELSA/EM. The number of win-loss-tie cases of ELSA/EM compared with greedy search is shown in the last row and column. ELSA/EM consistently outperforms the greedy search on models with few features and few clusters. As we noticed in the ELSA/K-means case, for more complex models with more than 10 selected features, the greedy method often shows higher classification accuracy.

Table 5.2: The classification accuracy of ELSA/EM and greedy.

K		Number of selected features						
		2	3	4	5	6	7	W-L-T
2	ELSA/EM	52.6±0.3	56.6±0.6	92.8±5.2	100±0.0	100±0.0	100±0.0	5-0-1
	Greedy	51.8±1.3	52.8±0.8	55.4±1.1	56.6±0.4	62.8±3.2	80.2±8.5	
3	ELSA/EM	83.2±4.8	52±6.6	91.6±5.7	93.8±6.2	99±1.0	100±0.0	4-0-2
	Greedy	40.6±0.3	40.8±0.2	40.2±0.2	63.6±3.8	100±0.0	100±0.0	
4	ELSA/EM	46.2±2.2	–	50.6±0.6	89.6±5.9	52±1.0	60.6±5.1	4-2-0
	Greedy	27.8±0.8	27.8±0.4	29±0.4	29.6±0.9	38±4.4	74.2±3.5	
5	ELSA/EM	44.6±2.0	32.6±3.8	72±3.8	62.4±1.9	66.4±3.7	88 ±4.9	5-0-1
	Greedy	23±0.4	22.2±0.8	24.2±0.9	23.8±0.5	29.6±1.7	81.2±3.0	
W-L-T		3-0-1	3-1-0	4-0-0	4-0-0	3-0-1	1-1-2	18-2-4

5.7 Experiments on WPBC data

In addition to the artificial data set discussed in Section 5.6, we also tested our algorithm on a real data set, the Wisconsin Prognostic Breast Cancer (WPBC) data [136]. This data set records 30 numeric features quantifying the nuclear grade of breast cancer patients at the University of Wisconsin Hospital, along with two

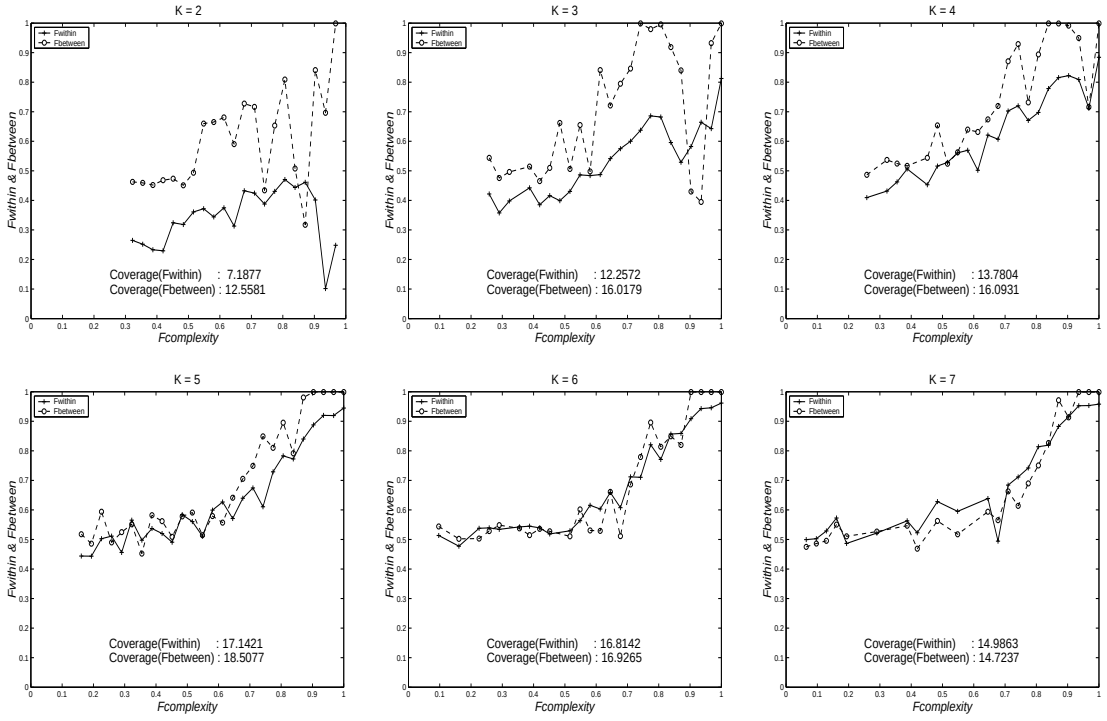

Figure 5.10: Candidate fronts evolved by ELSA/K-means on the WPBC data. The front for $K = 8$ is omitted because of its incomplete coverage of the search space.

traditional prognostic variables — tumor size and number of positive lymph nodes. This results in a total of 32 features for each of 198 cases. For the experiment, individuals are represented by 38 bits, 32 for the features and 6 for K ($K_{max} = 8$). Other ELSA parameters are the same as those used in the previous experiments.

5.7.1 Clustering analysis

In this experiment, we assume that we have no prior knowledge about the clusters and the relevant features. We first show two different types of fronts evolved by ELSA/K-means in Figure 5.10, one based on F_{within} and the other based on $F_{between}$.

The candidate fronts based on F_{within} in Figure 5.10 again show a typical decrease in quality from higher values of $F_{complexity}$ (lower complexity) to lower values of $F_{complexity}$ (higher complexity). It is interesting to note that the fronts based on $F_{between}$ not only show much more stable patterns than those in Figure 5.6 but also become almost identical to the fronts based on F_{within}, as K increases. We attribute this partially to the composition of correlated features in the WPBC data. The correlation among features comes from the fact that the mean, the standard error and the largest value of the 10 measurements that quantify the nuclear grade of breast cancer were recorded into the WPBC data, resulting in a highly correlated set of 30 features. Further, none of these features is regarded as white noise because each feature reflects some aspect of nuclear grade.

A decision maker might pick $K = 5$ as the correct number of clusters because the candidate front for $K = 5$ not only explores most of $F_{complexity}$ values but also shows a stable pattern with high clustering quality in terms of both F_{within} and $F_{between}$. However, let us select a model with $K = 3$ in order to compare our approach to previous research in which three clusters have been used to analyze this data set. In addition, the smaller number of clusters makes it easier to understand clustering results and satisfies one of our criteria, the preference for parsimonious models.

We also select a "best" solution (feature set) for prognostic analysis based on the value of $\widetilde{F}_{accuracy}$. In particular, we chose the solution with three clusters and the highest value of $\widetilde{F}_{accuracy}$ among solutions that have between five and ten features.

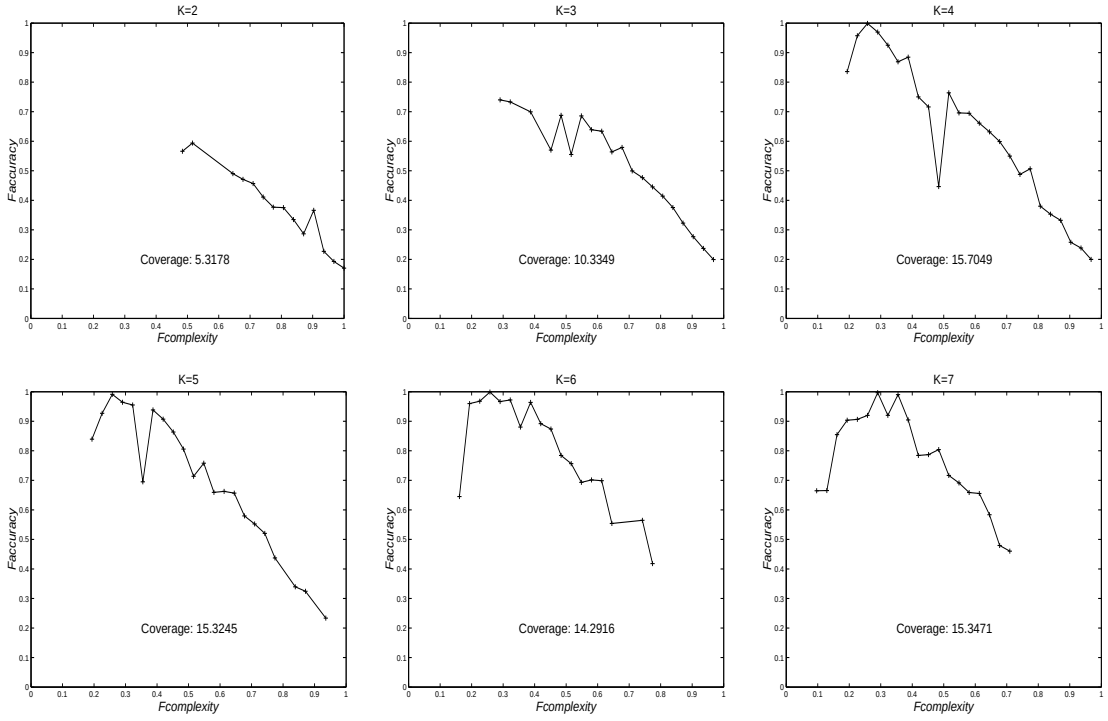

Figure 5.11: Candidate fronts evolved by ELSA/EM on the WPBC data. The front for $K = 8$ is omitted because of its incomplete coverage of the search space.

These minimum and maximum limits on the number of features are used to find a robust but simple solution, respectively. The chosen solution has seven features and its implication for prognostic analysis is discussed in Section 5.7.2.

Our findings by ELSA/K-means are confirmed in the candidate fronts evolved by ELSA/EM, shown in Figure 5.11. The correlation among features and the absence of white noise features result in a different characteristic shape of the candidate fronts from those in Figure 5.8. The fronts show a steady increase from the range of higher values of $F_{complexity}$ (lower complexity) to the range of lower values of $F_{complexity}$ (higher complexity). However, the curves peak at a certain point (e.g., $F_{complexity} = 0.26$ for

$K \geq 4$) because most of the information in the feature set is already extracted through previously selected features.

A decision maker might determine the correct number of clusters to be $K = 4$ or $K = 5$ because those models not only explore most of the $F_{complexity}$ values but also show a stable pattern with high clustering quality in terms of $F_{accuracy}$. For prognostic analysis, however, we again will consider solutions with three clusters, in order to be consistent with previous research. We note that the $F_{accuracy}$ values of solutions with up to 10 features are steadily improving, which makes it difficult to choose any one of them as our final solution. This makes us turn to the gradient information in the candidate front for $K = 3$. We choose a solution that causes the greatest improvement in clustering quality in terms of $F_{accuracy}$. The chosen solution has 11 features ($F_{complexity} = 0.68$) and we discuss its prognostic implication in the following section.

5.7.2 Prognostic analysis

We analyzed performance on this data set by looking for clinical relevance in the resulting clusters. Specifically, we observe the actual outcome (time to recurrence, or known disease-free time) of the cases in the three clusters. Figure 5.12 shows Kaplan-Meier estimates [104] of the true disease-free survival times for patients in the clusters found by ELSA/K-means.

Figure 5.12 displays well-separated survival characteristics of three prognostic groups: good (88 patients), intermediate (83 patients), and poor (27 patients). The

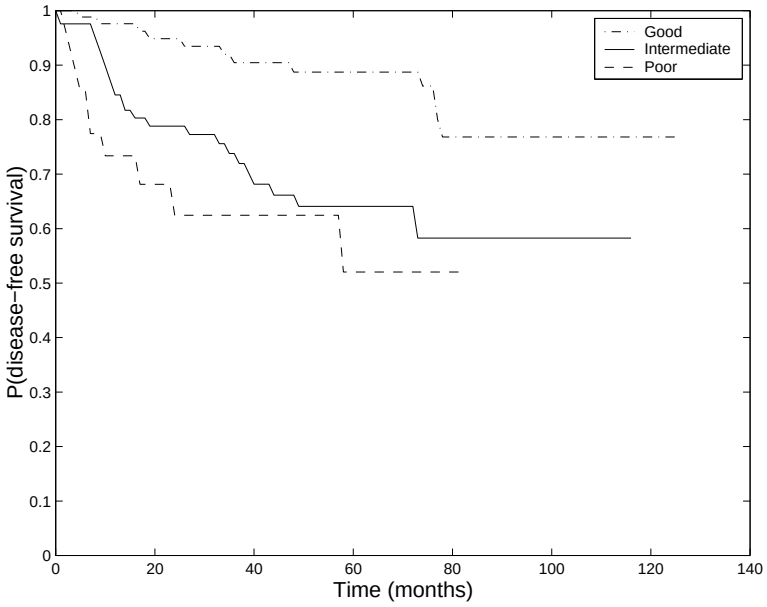

Figure 5.12: Estimated survival curves for the three groups found by ELSA/K-means.

good prognostic group was significantly different from the intermediate group ($p < 0.01$) and the intermediate group was well-differentiated from the poor group ($p < 0.06$).

Five-year recurrence rates of these groups were 11.28%, 35.91%, and 47.96%, respectively. The chosen solution used to cluster patients into three groups has seven dimensions including the mean nuclear radius and area, the standard error of the radius and area, and the largest value of the radius, perimeter and area. It is interesting that neither of the traditional medical prognostic factors, tumor size and lymph node status, is selected by ELSA/K-means.

Similarly, Figure 5.13 shows the survival characteristics of three prognostic groups found by ELSA/EM. The three groups showed well-separated survival char-

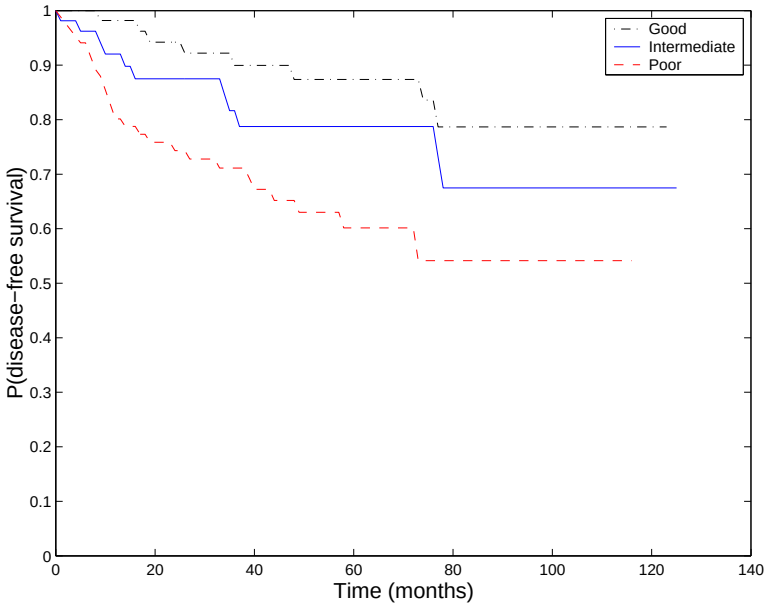

Figure 5.13: Estimated survival curves for the three groups found by ELSA/EM.

acteristics and more balanced clustering quality in the sense that patients are more evenly distributed. Out of 198 patients, 59 patients belong to the good prognostic group, and 54 patients and 85 patients belong to intermediate and poor prognostic groups, respectively. The good prognostic group was well-differentiated from the intermediate group ($p < 0.076$) and the intermediate group was significantly different from the poor group ($p < 0.036$). Five-year recurrence rates were 12.61%, 21.26%, and 39.85% for the patients in the three groups.

The chosen dimensions by ELSA/EM included a mix of nuclear morphometric features such as the mean and the standard error of the radius, perimeter and area, and the largest value of the area and symmetry along three other features. We note that again neither of the traditional medical prognostic factors is chosen, which is

consistent with the result of ELSA/K-means. This finding is potentially important because one of the traditional prognostic factors, the lymph node status, can be determined by microscopic examination of lymph nodes only after they are surgically removed from the patient's armpit [193]. Our experiments tend to support the hypothesis that prognostic groups with significantly different expected outcomes can be formed without this data.

In order to address this matter, we investigate whether other solutions with lymph node information can form three prognostic groups as good as our EM chosen solution. For this purpose, we selected Pareto solutions across all different K values that have fewer than 10 features including lymph node information and formed three clusters using these selected features, disregarding the evolved value of K. The survival characteristics of the three prognostic groups found by the best of these solutions was very competitive with our chosen solution. The good prognostic group was well-differentiated from the intermediate group ($p < 0.10$), and the difference between the intermediate group and the poor group was significant ($p < 0.026$). This suggests that lymph node status may indeed have strong prognostic effects, even though it is excluded from the best models evolved by our algorithms.

5.8 Conclusions

We presented a novel evolutionary multi-objective local selection algorithm for unsupervised feature selection. ELSA, an evolutionary local selection algorithm, was used successfully in previous work in conjunction with supervised learning [141, 105].

As an extension of our previous work [106], we used ELSA to search for possible combination of features and numbers of clusters, with the guidance of two representative clustering algorithms, K-means and EM. 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 while simultaneously optimizing the multiple criteria separately.
- A number of possibly conflicting heuristic metrics can be plugged into the algorithm, while remaining agnostic about their relative worth or their relationships.
- Our algorithm, both ELSA/K-means and ELSA/EM, outperforms a greedy algorithm in terms of classification accuracy.
- Most importantly, in the proposed framework we can reliably select an appropriate clustering model, including significant features and the number of clusters. The result is a set of clusters that accurately models the data, and is more interpretable and scalable due to the reduced dimensionality.

In future work we would like to compare the performance of ELSA on the unsupervised feature selection task with other multi-objective EAs, using each in conjunction with clustering algorithms.

Another promising future direction will be a direct comparison of different clustering algorithms. In the results presented in this chapter, ELSA/EM shows

better results than ELSA/K-means on the synthetic data in terms of the composition of selected features and prediction accuracy. Furthermore, EM allows for easier choice of best compromise solution because of single quality metric. However, ELSA/K-means shows very competitive results on the real data set in terms of well-separated survival curves. Further, ELSA/K-means is much faster (roughly by a factor of 3) than ELSA/EM to evaluate the fixed number of solutions. Thus, it is possible for ELSA/K-means to find better solutions given the same amount of computing time as ELSA/EM.

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 problem of finding relevant prognostic factors in breast cancer, or determining the variables that define relevant market segments.

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, allowing the application of iterative algorithms like K-means on much larger problems.

CHAPTER 6
CONCLUSION

This chapter summarize the thesis and highlights its contributions. We then draw conclusions from experiments conducted and addresses directions of future research. The thesis began by emphasizing the importance of feature selection and reviewed the fundamental concepts and representative algorithms for feature selection in supervised and unsupervised learning. Through the thesis, we restricted our interests to the wrapper model of feature selection mainly because the wrapper model has been proved to return higher predictive accuracy in supervised learning. As a search algorithm, we used Evolutionary Algorithms (EAs) to intelligently search the space of possible feature subsets for large-scale problems.

In particular, we note that each feature subset should be evaluated in terms of multiple objectives. For example, both the number of selected features (which should be minimized) and the accuracy (which should be maximized) are important criteria for a feature selection task in supervised learning. The relative weights of the various objectives can be determined only by the final decision maker for her application. However, standard EAs assume a single (or combined) fitness function to be optimized and thus cannot consider multiple fitness criteria effectively. We claim that our approach is a generalization of such previous work, in the sense that it could capture both linear and non-linear relationships among the criteria.

From the perspective of knowledge discovery, our goal is to provide a clear picture of the tradeoffs among the various objectives. This is important because no single criterion for feature selection is best for every application. By providing a set of alternative solutions to the decision maker, our approach helps the decision maker to choose the right solution at the right time from a set of high-quality compromise solutions. This could present a big advantage over other decision support systems that provide the decision maker with a single solution, given that she might not be familiar with how the algorithm reached the solution.

Though a number of multi-objective extensions of evolutionary algorithms have been proposed, most of them employ computationally expensive selection mechanisms to favor dominating solutions and to maintain diversity. In this thesis, we proposed an algorithm, Evolutionary Local Selection Algorithms (ELSA), where each individual solution is allocated to a local environment based on its criteria values. Each solution competes with others to consume shared resources only if they are located in the same environment. The more densely populated the local environment, the more competition among individuals for resources, resulting in bias toward different local environments.

This *local* selection mechanism minimizes the communication among agents, which makes ELSA efficient and scalable. In particular, ELSA can be useful for various tasks in which the maintenance of diversity within the population is more important than a speedy convergence to the optimum. The superior ability to locate

more of the Pareto front was proved in a comparative experiment with other multi-criteria evolutionary algorithms. Based on this notion, we applied ELSA combined with supervised and unsupervised learning algorithms to feature selection problems over three chapters.

In Chapter 3, ELSA with neural networks model (ELSA/ANN) was proposed for customer targeting in database marketing. In the ELSA/ANN model, ELSA and neural networks were used to search for possible combinations of features and to score customers based on the probability of buying new insurance product respectively. The ELSA/ANN model showed promising results in two different experiments, when market managers have clear decision scenario or not. When market managers are clear about how a model will be used, the overall performance of ELSA/ANN was superior to the industry standard PCA/logit model both in terms of accuracy and in terms of interpretability. Under a more general decision scenario, ELSA/ANN yielded a more accurate model over a broad selection percentage range at the cost of increasing the number of predictive features in the specification.

The ELSA/ANN procedure can be easily modified to take into account different objectives. With information of campaign costs and profit per additional actual customer, a direct marketer could use ELSA/ANN to choose the best selection point where expected total revenue is maximized. In this way, it would be possible to determine the type of decision rule that the marketer should adopt, both in terms of solicitation percentage as well as predictive rule. Because all mailing lists do not

all have the same potential for the marketer, this approach would allow a predictive model and solicitation-mailing rule to be customized as the firm's database changes.

In Chapter 4, we proposed a new ensemble construction algorithm, Meta-Evolutionary Ensembles (MEE), where the ensembles compete directly based on classification accuracy while individual classifiers in the same ensemble compete against each other to correctly predict held-out examples. In MEE, feature selection is used as the diversity mechanism among classifiers in the ensemble and classifiers are rewarded for predicting difficult points, relative to the other members of their respective ensembles. Our experimental results indicate that this method shows consistently improved performance compared to a single classifier and comparable performance compared to the traditional ensembles and GEFS.

Once we have a better understanding of how and why ensemble methods achieve improved predictive accuracy, we can compare this algorithm more rigorously to others on a larger collection of data sets. We also need to perform any necessary performance tweaks on the EA energy allocation scheme. Along the way, we will examine the role of classifiers type and various characteristics of ensembles including size and diversity. By giving the system as many degrees of freedom as possible and observing the characteristics that lead to successful ensembles, we can directly optimize these characteristics and translate the results to a more scalable architecture for large-scale predictive tasks.

In Chapter 5, ELSA was used for unsupervised feature selection. Specifically,

we used ELSA to search for both possible combination of features and numbers of clusters, with the guidance of two representative clustering algorithms, K-means and EM. Our algorithm, both ELSA/K-means and ELSA/EM, outperforms a greedy algorithm in terms of classification accuracy. Most importantly, in the proposed framework we can reliably select an appropriate clustering model, including significant features and the number of clusters.

One direction of future research on the unsupervised feature selection task is to compare ELSA with other multi-objective EAs using each in conjunction with clustering algorithms. Another promising future direction will be a direct comparison of different clustering algorithms. In the results presented in Chapter 5, ELSA/EM shows better results than ELSA/K-means on the synthetic data in terms of the composition of selected features and prediction accuracy. Furthermore, EM allows for easier choice of best compromise solution because of single quality metric. However, ELSA/K-means shows very competitive results on the real data set in terms of well-separated survival curves and is much faster than ELSA/EM to evaluate the fixed number of solutions. Thus, it is worthy to see whether ELSA/K-means can find better solutions given the same amount of computing time as ELSA/EM.

In addition to research direction tailored to each specific tasks, there are some more fundamental issues to be studied in future research. One major direction of future research on the feature selection with ELSA is to find a way that can boost weak selection pressure of ELSA while keeping its local selection mechanism. For problems

requiring effective selection pressure, local selection may be too weak because the only selection pressure that ELSA can apply comes from the sharing of resources. Dynamically adjusting the local environmental structure based on the certain ranges of the observed fitness values over a fixed number of generation could be a promising solution. In this way, we could avoid the case in which the solution with the worst performance can survive into the next generation because there are no other solutions in its local environment.

Another major direction of future research is related with the scalability issue. By minimizing the communication among agents, our local selection mechanism makes ELSA efficient and scalable. However, our models suffer the inherent weakness of the wrapper model, the computational complexity. Further by combining EAs with ANN to take the advantages of both algorithms, it is possible that the combined model can be so slow that it cannot provide solutions in a timely manner. With the rapid growth of records and variables in database, this failure can be critical. Combining ELSA with faster learning algorithms such as decision tree algorithms and Support Vector Machine (SVM) will be worthy to pursue.

REFERENCES

[1] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan. Automatic subspace clustering of high dimensional data for data mining applications. In *Proc. of the ACM SIGMOD Int'l Conf. on Management of Data*, pages 94–105, Seattle, WA, 1998.

[2] D. W. Aha and R. L. Bankert. Feature selection for case-based classification of cloud types: An empirical comparison. In *Working Notes of the AAAI-94 Workshop on Case-Based Reasoning*, pages 106–112, 1994.

[3] D. W. Aha and R. L. Bankert. A comparative evaluation of sequential feature selection algorithms. In D. Fisher and H. Lenz, editors, *Proc. of 5th Int'l Workshop on Artificial Intelligence and Statistics*, pages 1–7, Ft. Lauderdale, FL, 1995.

[4] D. W. Aha, D. Kibler, and M. K. Albert. Instance-based learning algorithms. *Machine Learning*, 6(1):37–66, 1991.

[5] S. C. Ahalt, A. K. Krishnamurthy, P. Chen, and D. E. Melton. Competitive learning algorithms for vector quantization. *Neural Networks*, 3:277–290, 1990.

[6] K. S. Al-Sultan. A tabu search approach to the clustering problem. *Pattern Recognition*, 28(9):1443–1451, 1995.

[7] H. Almuallim and T. G. Dietterich. Learning with many irrelevant features. In *Proc. of 9th National Conf. on Artificial Intelligence*, pages 547–552. AAAI Press, 1991.

[8] H. Almuallim and T. G. Dietterich. Efficient algorithms for identifying relevant features. In *Proc. of 9th Canadian Conf. on Artificial Intelligence, Vancouver, Britich Columbia*, pages 38–45. Morgan Kaufmann, 1992.

[9] H. Almuallim and T. G. Dietterich. Learning boolean concepts in the presence of many irrelevant features. *Artificial Intelligence*, 69(1–2):279–306, 1994.

[10] E. I. Altman. *Corporate Financial Distress and Bankruptcy: A Complete Guide to Predicting and Avoiding Distress and Profitting from Bankruptcy.* John Wiley and Sons, New York, NY, 1993.

[11] C. Apt, F. J. Damerau, and S. M. Weiss. Automated learning of decision rules for text categorization. *ACM Transactions on Information Systems*, 12(3):233–251, 1994.

[12] 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.

[13] J. Bala, J. Huang, H. Vafaie, K. DeJong, and H. Wechsler. Hybrid learning using genetic algorithms and decision trees for pattern classification. In *Proc. of 14th Int'l Joint Conf. on Artificial Intelligence*, pages 719–724, 1995.

[14] P. V. (Sundar) Balakrishnan, M. C. Cooper, V. S. Jacob, and P. A. Lewis. Comparative performance of the FSCL neural net and K-means algorithm for market segmentation. *European Journal of Operation Research*, 93(10):346–357, 1996.

[15] R. Battiti. Using mutual information for selecting features in supervised neural net learning. *IEEE Transactions on Neural Networks*, 5(4):537–550, July 1994.

[16] E. Bauer and R. Kohavi. An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. *Machine Learning*, 36:105–139, 1999.

[17] M. Ben-Bassat. Use of distance measures, information measures and error bounds in feature evaluation. In P. R. Krishnaiah and L. N. kanal, editors, *Handbook of Statistics*, volume 2, pages 773–791. North-Holland Publishing Company, 1982.

[18] K. P. Bennett and O. L. Mangasarian. Robust linear programming discrimination of two linearly inseparable sets. *Optimization Methods and Software*, 1:23–34, 1992.

[19] S. Bhattacharyya. Evolutionary algorithms in data mining: Multi-objective performance modeling for direct marketing. In *Proc. of 6th ACM SIGKDD Int'l Conf. on Knowledge Discovery & Data Mining (KDD-00)*, pages 465–473, 2000.

[20] C. Bishop. *Neural Networks for Pattern Recognition.* Oxford University Press, 1995.

[21] 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.

[22] A. L. Blum and P. Langley. Selection of relevant features and examples in machine learning. *Artificial Intelligence*, pages 245–271, 1997.

[23] A. L. Blum and R. L. Rivest. Training a 3-node neural network is NP-complete. *Neural Networks*, 5:117–127, 1992.

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

[25] B. Bonnlander. *Nonparametric selection of input variables for connectionist learning.* PhD thesis, University of Colorado, 1996.

[26] Lon Bottou and Yoshua Bengio. Convergence properties of the K-means algorithm. In G. Tesauro, D. Touretzky, and T. Leen, editors, *Advances in Neural Information Processing Systems*, volume 7, pages 585–592. MIT Press, 1995.

[27] 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, *Proc. of 4th Int'l Conf. on Knowledge Discovery and Data Mining (KDD98)*, pages 9–15, Menlo Park, CA, 1998. AAAI Press.

[28] P. S. Bradley, U. M. Fayyad, and C. Reina. Scaling EM (expectation-maximization) clustering to large databases. Technical Report MSR-TR-98-35, Microsoft, 1998.

[29] P. S. Bradley and O. L. Mangasarian. Feature selection via concave minimization and support vector machines. In J. Shavlik, editor, *Proc. of 15th Int'l Conf. on Machine Learning*, pages 82–90, San Francisco, CA, 1998. Morgan Kaufmann.

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

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

[32] G. Brassard and P. Bratley. *Fundamentals of Algorithms.* Prentice Hall, New Jersey, 1996.

[33] L. Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

[34] L. Breiman. Bias, variance, and Arching classifiers. Technical Report 460, University of California, Department of Statistics, Berkeley, California, 1996.

[35] L. Breiman. Stacked regression. *Machine Learning*, 24(1):49–64, 1996.

[36] L. Breiman. Random forests–Random features. Technical Report 567, University of California, Department of Statistics, Berkeley, California, 1999.

[37] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees.* Wadsworth International Group, 1984.

[38] F. Z. Brill, D. E. Brown, and W. N. Martin. Fast genetic selection of features for neural network classifiers. *IEEE Transactions on Neural Networks*, 3(2):324–328, 1992.

[39] J. Buhmann. Data clustering and learning. In M. Arbib, editor, *Handbook of Brain Theory and Neural Networks.* Bradfort Books/MIT Press, 1995.

[40] W. Buntine and T. Niblett. Technical note: A further comparison of splitting rules for decision tree induction. *Machine Learning*, 8(1):75–85, 1992.

[41] F. Can. Incremental clustering for dynamic information processing. *ACM Transactions on Information Systems*, 11(2):143–164, 1993.

[42] C. Cardie. Using decision trees to improve case-based learning. In *Proc. of 10th Int'l Conf. on Machine Learning*, pages 25–32. Morgan Kaufmann, 1993.

[43] G. Carpenter and S. Grossberg. ART3: Hierarchical search using chemical transimtters in self-organizing pattern recognition architectures. *Neural Networks*, 3:129–152, 1990.

[44] R. Caruana and D. Freitag. Greedy attribute selection. In *Proc. of 11th Int'l Conf. on Machine Learning*, pages 28–36, New Brunswick, NJ, 1994. Morgan Kaufmann.

[45] G. Cestnik, I. Kononenko, and I. Bratko. Assistant 86: A knowledge acquisition tool for sophisticated users. In I. Bratko and N. Lavrac, editors, *Progress in Machine Learning.* Sigma Press, 1987.

[46] E. I. Chang and R. P. Lippmann. Using genetic algorithms to improve pattern classification performance. In R. P. Lippmann, J. E. Moody, and D. S. Touretzky, editors, *Advances in Neural Information Processing Systems*, volume 3, pages 797–803. Morgan Kaufmann, 1991.

[47] P. Cheeseman, J. Kelly, M. Self, J. Stutz, W. Taylor, and D. Freeman. AutoClass: A Bayesian classification system. In *Proc. of 5th Int'l Conf. on Machine Learning*, pages 54–64, San Francisco, CA, 1988. Morgan Kaufmann.

[48] P. Cheeseman and J. Stutz. Bayesian classification system (AutoClass): Theory and results. In U. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*, pages 153–180, San Francisco, CA, 1996. MIT Press.

[49] Stephen Chen, Cesar Guerra-Salcedo, and Stephen Smith. Non-standard crossover for a standard representation – commonality-based feature subset selection. In *GECCO-99: Proc. of the Genetic and Evolutionary Computation Conference*, pages 129–134. Morgan Kaufmann, 1999.

[50] V. Cherkassky and F. Mulier. *Learning from Data: Concepts, Theory and Methods*. John Wiley and Sons, 1998.

[51] K. J. Cherkauer and J. W. Shavlik. Growing simpler decision trees to facilitate knowledge discovery. In *Proc. of 2nd Int'l Conf. on Knowledge Discovery & Data Mining*, pages 315–318, 1996.

[52] T. M. Cover and J. M. Campenhout. On the possible orderings in the measurement selection problem. *IEEE Transactions on Systems, Man, and Cybernetics*, 7:657–661, 1977.

[53] P. Cunningham and J. Carney. Diversity versus quality in classification ensembles based on feature selection. Technical Report TCD-CS-2000-02, Trinity College Dublin, Department of Computer Science, 2000.

[54] G. Cybenko. Continuous valued neural networks with two hidden layers are sufficient. Technical report, Tufts University, Department of Computer Science, 1988.

[55] 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.

[56] K. DeJong. Learning with genetic algorithms: An overview. *Machine Learning*, 3:121–138, 1988.

[57] 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.

[58] Mark Devaney and Ashwin Ram. Efficient feature selection in conceptual clustering. In *Proc. 14th Int'l Conf. on Machine Learning*, pages 92–97. Morgan Kaufmann, 1997.

[59] P. A. Devijver and J. Kittler. *Pattern Recognition: A Statistical Approach.* Prentice Hall International, 1982.

[60] T. G. Dietterich. An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting and randomization. *Machine Learning*, 40(2):139–157, August 2000.

[61] T. G. Dietterich, H. Hild, and G. Bakiri. A comparison of ID3 and backpropagation for English text-to-speech mapping. *Machine Learning*, 18(1):51–80, 1995.

[62] N. R. Draper and H. Smith. *Applied Regression Analysis.* John Wiley and Sons, second edition, 1981.

[63] R. Duda and P. Hart. *Pattern Classification and Scene Analysis.* John Wiley and Sons, New York, 1973.

[64] J. G. Dy and C. E. Brodley. Visualization and interactive feature selection for unsupervised data. In *Proc. 6th ACM SIGKDD Int'l Conf. on Knowledge Discovery & Data Mining (KDD-00)*, pages 360–364, 2000.

[65] Jennifer G. Dy and Carla E. Brodley. Feature subset selection and order identification for unsupervised learning. In *Proc. 17th Int'l Conf. on Machine Learning*, pages 247–254. Morgan Kaufmann, San Francisco, CA, 2000.

[66] C. Emmanouilidis, A. Hunter, and J. MacIntyre. A multiobjective evolutionary setting for feature selection and a commonality-based crossover operator. In *2000 Congress on Evolutionary Computation (CEC-2000), San Diego, California*, pages 309–316. IEEE Service Center, July 2000.

[67] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A density based algorithm for discovering clusters in large spatial database with noise. In *Proc. of 2nd Int'l Conf. on Knowledge Discovery and Data Mining (KDD'96)*, pages 226–231, August 1996.

[68] M. Ester, H.-P. Kriegel, and X. Xu. A database interface for clustering in large spatial databases. In *Proc. of 1st Int'l Conf. on Knowledge Discovery and Data Mining KDD-95*, pages 94–99. AAAI Press, 1995.

[69] F. Ferri, P. Pudil, M. Hatef, and J. Kittler. Comparative study of techniques for large scale feature selection. In E. Gelsema and L. Kanal, editors, *Pattern Recognition in Practice IV*, pages 403–413. Elsevier Science B.V., 1994.

[70] D. Fisher. COBWEB: Knowledge acquisition via conceptual clustering. *Machine Learning*, 2:139–172, 1987.

[71] I. Foroutan and J. Sklansky. Feature selection for automatic classification of non-Gaussian data. *IEEE Transactions on Systems, Man, and Cybernetics*, 17:187–198, 1987.

[72] Y. Freund and R. Schapire. Experiments with a new boosting algorithm. In *Proc. of 13th Int'l Conf. on Machine Learning*, pages 148–156, Bari, Italy, 1996.

[73] K. Fukunaga. *Introduction to Statistical Pattern Recognition.* Academic Press, San Diego, CA, 1990.

[74] I. Gath and A. B. Geva. Unsupervised optimal fuzzy clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(7):773–781, 1988.

[75] J. H. Gennari, P. Langley, and D. Fisher. Models of incremental concept formation. *Artificial Intelligence*, 40:11–61, 1989.

[76] C. Glymour, D. Madigan, D. Pregibon, and P. Smyth. Statistical themes and lessons for data mining. *Data Mining and Knowledge Discovery*, 1(1):11–28, 1997.

[77] D. E. Goldberg and J. Richardson. Genetic algorithms with sharing for multimodal function optimization. In *Proc. of 2nd Int'l Conf. on Genetic Algorithms*, pages 41–49, Hillsdale, NJ, 1987. Lawrence Erlbaum.

[78] P. E. Green and V. Srinivasan. Conjoint analysis in marketing: New developments with implications for research and practice. *Journal of Marketing*, 54(4):3–19, 1990.

[79] C. Guerra-Salcedo, S. Chen, D. Whitley, and S. Smith. Fast and accurate feature selection using hybrid genetic strategies. In *Proc. of Genetic and Evolutionary Computation Conference*, pages 177–184, Piscataway, NJ, 1999. IEEE Service Center.

[80] C. Guerra-Salcedo and D. Whitley. Genetic approach to feature selection for ensemble creation. In *GECCO-99: Proceedings of the Genetic and Evolutionary Computation Conference*, pages 236–243. Morgan Kaufmann, 1999.

[81] S. Guha, R. Rastogi, and K. Shim. CURE: An efficient clustering algorithm for large databases. In *Proc. of ACM SIGMOD Int'l Conf. on Management of Data (SIGMOD98)*, pages 73–84, 1998.

[82] L. O. Hall, I. B. Ozyurt, and J. C. Bezdek. Clustering with a genetically opitmized approach. *IEEE Transactions on Evolutionary Computation*, 3(2):103–112, 1999.

[83] A. Hart. Experience in the use of an inductive system in knowledge engineering. In M. Bramer, editor, *Handbook of Evolutionary Computation*. Cambridge Univ. Press, Cambridge, MA, 1984.

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

[85] S. Hashem. Optimal linear combinations of neural networks. *Neural Networks*, 10(4):599–614, 1997.

[86] B. Hassibi and D. G. Stork. Second order derivatives for network pruning: Optimal brain surgeon. In S. J. Hanson, J. Cowan, and L. Giles, editors, *Advances in Neural Information Processing Systems*, volume 5, pages 164–171, San Mateo, CA, 1993. Morgan Kaufmann.

[87] T. K. Ho. C4.5 decision forests. In *Proc. of 14th Int'l Conf. on Pattern Recognition*, pages 545–549, 1998.

[88] T. K. Ho. The random subspace method for constructing decision forests. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(8):832–844, 1998.

[89] J. H. Holland. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, MI, 1975.

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

[91] J. Horn, N. Nafpliotis, and D. E. Goldberg. A niched pareto generic algorithms for multiobjective optimization. In *Proc. of 1st IEEE Conf. on Evolutionary Computation*, pages 82–87, Piscataway, NJ, 1994. IEEE Service Center.

[92] H. Hruschka and M. Natter. Comparing performance of feedforward neural nets and K-means for market segmentation. *European Journal of Operational Research*, 114:346–353, 1999.

[93] E. B. Hunt, J. Marin, and P. J. Stone. *Experiments in Induction.* Academic Press, New York, NY, 1966.

[94] L. Hyafil and R. L. Rivest. Constructing optimal binary decision trees is NP-complete. *Information Processing Letters*, 5(1):15–17, 1976.

[95] A. Jain and R. Dubes. *Algorithms for Clustering Data.* Prentice-Hall, Englewood Cliffs, NJ, 1989.

[96] A. Jain, M. Murty, and P. Flynn. Data clustering: A review. *ACM Computing Surveys*, 31(3):264–323, 1999.

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

[98] D. D. Jensen and P. R. Cohen. Multiple comparisons in induction algorithms. *ml*, 38(3):309–338, 2000.

[99] G. H. John, R. Kohavi, and K. Pfleger. Irrelevant features and the subset selection problem. In *Proc. of 11th Int'l Conf. on Machine Learning*, pages 121–129, San Mateo, CA, 1994. Morgan Kaufmann.

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

[101] K. A. De Jong and W. M. Spears. Using genetic algorithms for concept learning. In *Proc. of 3rd Int'l Conf. on Genetic Algorithms*, pages 124–132, Fairfax, VA, 1989. George Mason University.

[102] J. S. Judd. On the complexity of loading shallow neural networks. *Journal of Complexity*, 4:177–192, 1988.

[103] L. P. Kaelbling. *Learning in Embedded Systems.* MIT Press, 1993.

[104] E. L. Kaplan and P. Meier. Nonparametric estimation from incomplete observations. *Journal of the American Statistical Association*, 53:457–481, 1958.

[105] Y. Kim and W. N. Street. CoIL challenge 2000: Choosing and explaining likely caravan insurance customers. Technical Report 2000–09, Sentient Machine Research and Leiden Institute of Advanced Computer Science, June 2000. http://www.wi.leidenuniv.nl/~putten/library/cc2000/.

[106] Y. Kim, W. N. Street, and F. Menczer. Feature selection in unsupervised learning via evolutionary search. In *Proc. of 6th ACM SIGKDD Int'l Conf. on Knowledge Discovery & Data Mining (KDD-00)*, pages 365–369, 2000.

[107] Y. Kim, W. N. Street, and F. Menczer. An evolutionary multi-objective local selection algorithm for customer targeting. In *Proc. of Congress on Evolutionary Computation (CEC-01)*, pages 759–766, 2001.

[108] K. Kira and L. A. Rendell. The feature selection problem: Traditional methods and a new algorithm. In *10th National Conference on Artificial Intelligence*, pages 129–134. MIT Press, 1992.

[109] K. Kira and L. A. Rendell. A practical approach to feature selection. In *Proc. of 9th Int'l Conf. on Machine Learning*, pages 249–256. Morgan Kaufmann, 1992.

[110] 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.

[111] J. Kittler. Feature selection and extraction. In Y. Fu, editor, *Handbook of Pattern Recognition and Image Processing*, New York, 1986. Academic Press.

[112] R. Kohavi. Feature subset selection as search with probabilistic estimates. In *AAAI Fall Symposium on Relevance*, pages 122–126, 1994.

[113] R. Kohavi and G. H. John. Wrappers for feature subset selection. *Artificial Intelligence*, 97(1–2):273–324, 1997.

[114] R. Kohavi and D. Sommerfield. Feature subset selection using the wrapper method: Overfitting and dynamic search space topology. In U. M. Fayyad and R. Uthurusamy, editors, *Proc. of 1st Int'l Conf. on Knowledge Discovery & Data Mining*, pages 192–197. AAAI Press, 1995.

[115] D. Koller and M. Sahami. Toward optimal feature selection. In L. Saitta, editor, *Proc. of 13th Int'l Conf. on Machine Learning*, pages 284–292, San Francisco, 1996. Morgan Kaufmann.

[116] I. Kononenko. Estimating attributes: Analysis and extension of RELIEF. In L. D. Raedt, editor, *Proc. of European Conf. on Machine Learning*, pages 171–182, Berlin, 1994. Springer-Verlag.

[117] I. Kononenko and S. J. Hong. Attribute selection for modeling. *Future Generation Computer Systems*, 13(2–3):181–195, 1997.

[118] P. Kontkanen, P. Myllymaki, and H. Tirri. Comparing Bayesian model class selection criteria by discrete finite mixtures. In D. Dowe, K. Korb, and J. Oliver, editors, *Proc. of Information, Statistics and Induction in Science Conference (ISIS'96)*, pages 364–374. World Scientific, Singapore, 1996.

[119] W. L. G. Koontz, K. Fukunaga, and P. M. Narendra. A branch and bound clustering algorithm. *IEEE Transactions on Computing*, 23:908–914, 1975.

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

[121] A. Krogh and J. Vedelsby. Neural network ensembles, cross validation, and active learning. In D. Touretzky G. Tesauro and T. Leen, editors, *Advances in Neural Information Processing Systems*, volume 7, pages 231–238, Cambridge, MA, 1995. MIT Press.

[122] M. Kubat, D. Flotzinger, and G. Pfurtscheller. Discovering patterns in EEG signals: Comparative study of a few methods. In *Proc. of 1993 European Conf. on Machine Learning*, pages 367–371. Springer-Verlag, 1993.

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

[124] L. I. Kuncheva and C. J. Whitaker. Measures of diversity in classier ensembles. *Machine Learning*, 2000. (submitted).

[125] P. Langeley, W. Iba, and K. Thompson. An analysis of Bayesian classifiers. In *Proc. of 10th National Conf. on Artificial Intelligence*, pages 223–228. AAAI Press and MIT Press, 1992.

[126] P. Langeley and S. Sage. Induction of selective Bayesian classifiers. In *Proc. of 10th Conf. on Uncertainty in Artificial Intelligence*, pages 113–117, Seattle, WA, 1994. AAAI Press.

[127] P. Langeley and S. Sage. Oblivious decision trees and abstract cases. In *Working Notes of the AAAI-94 Workshop on Case-Based Reasoning*, pages 113–117, Seattle, WA, 1994. AAAI Press.

[128] P. Langeley and S. Sage. Scaling to domains with many irrelevant features. In R. Greiner, editor, *Computational Learning Theory and Natural Learning Systems*, volume 4, Cambridge, MA, 1997. MIT Press.

[129] Y. LeCun, J. S. Denker, and S. A. Solla. Optimal brain damage. In D. S. Touretzky, editor, *Advances in Neural Information Processing Systems*, volume 2, pages 598–605, San Mateo, CA, 1990. Morgan Kaufmann.

[130] Y.-J. Lee, O. L. Mangasarian, and W. H. Wolberg. Breast cancer survival and chemotherapy: A support vector machine analysis. Technical Report Data Mining Institute Technical Report 99-10, University of Winconsin, Computer Sciences Department, 1999.

[131] C. X. Ling and C. Li. Data mining for direct marketing: Problems and solutions. In *Proc. of 4th ACM SIGKDD Int'l Conf. on Knowledge Discovery & Data Mining (KDD-98)*, pages 73–79, 1998.

[132] H. Liu and R. Setiono. Chi2: Feature selection and discretization of numeric attributes. In *Proc. of 7th Int'l Conf. on Tools with Artificial Intelligence*, pages 388–391, Washington D.C., 1995. Morgan Kaufmann.

[133] H. Liu and R. Setiono. Feature selection and classification - a probabilistic wrapper approach. In *Proc. of 9th Int'l Conf. on Industrial & Engineering Applications of AI and Expert Systems*, pages 419–424, Fukuoka, Japan, 1996.

[134] W. Z. Liu and A. P. White. The importance of attribute selection measures in decsion tree induction. *Machine Learning*, 15(1):24–41, 1994.

[135] O. L. Mangasarian. Mathematical programming in neural networks. *ORSA Journal on Computing*, 5:349–360, 1993.

[136] 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.

[137] T. Marill and D. M. Green. On the effectiveness of receptors in recognition systems. *IEEE Transactions on Information Theory*, 9:11–17, 1963.

[138] M. Meila and D. Heckerman. An experimental comparison of several clustering methods. Technical Report MSR-TR-98-06, Microsoft, Redmond, WA, 1998.

[139] F. Menczer and R. K. Belew. From complex environments to complex behaviors. *Adaptive Behavior*, 4:317–363, 1996.

[140] F. Menczer and R. K. Belew. Adaptive retrieval agents: Internalizing local context and scaling up to the web. *Machine Learning*, 39:203–242, 2000.

[141] F. Menczer, M. Degeratu, and W. N. Street. Efficient and scalable Pareto optimization by evolutionary local selection algorithms. *Evolutionary Computation*, 8(2):223–247, Summer 2000.

[142] 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 Intelligent Agents*. MIT Press, Cambridge, MA, 2000.

[143] A. J. Miller. *Subset Selection in Regression*. Chapman and Hall, 1990.

[144] J. Mingers. An empirical comparison of selection measures for decision-tree induction. *Machine Learning*, 3(4):319–342, 1989.

[145] T. Mitchell. Generalization as search. *Artificial Intelligence*, 18:203–226, 1982.

[146] T. Mitchell. *Machine Learning*. McGraw-Hill, New York, 1997.

[147] M. Modrzejewski. Feature selection using rough sets theory. In P. B. Brazdil, editor, *Proc. of European Conf. on Machine Learning*, pages 213–226, 1993.

[148] A. W. Moore and M. S. Lee. Efficient algorithms for minimizing cross validation error. In *Proc. of 11th Int'l Conf. on Machine Learning*, pages 190–198, New Brunswick, NJ, 1994. Morgan Kaufmann.

[149] M. C. Mozer. A focused back-propagation algorithm for temporal pattern recognition. *Complex Systems*, 3:349–381, 1989.

[150] S. Murthy and S. Salzberg. Lookahead and pathology in decision tree induction. In C. S. Mellish, editor, *Proc. of 14th Int'l Joint Conf. on Artificial Intelligence*, pages 1025–1031. Morgan Kaufmann, 1995.

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

[152] J. Neter, W. Wasserman, and M. H. Kutner. *Applied Linear Statistical Models.* Irwin, Homewood, IL, third edition, 1990.

[153] R. T. Ng and J. Han. Efficient and effective clustering methods for spatial data mining. In *Proc. of 20th Int'l Conf. on Very Large Databases (VLDB'94)*, pages 144–155, Santiago, Chile, September 1994.

[154] K. L. Oehler and R. M. Gray. Combining image compression and classification using vector quantization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17:461–473, 1995.

[155] D. Opitz. Feature selection for ensembles. In *16th National Conf. on Artificial Intelligence (AAAI)*, pages 379–384, Orlando, FL, 1999.

[156] D. Opitz and R. Maclin. Popular ensemble methods: An empirical study. *Journal of Artificial Intelligence Research*, 11:169–198, 1999.

[157] D. Opitz and J. Shavlik. Actively searching for an effective neural-network ensemble. *Connection Science*, 8(3/4):337–353, 1996.

[158] G. Pagallo and D. Haussler. Boolean feature discovery in empirical learning. *Machine Learning*, 5(1):71–100, 1990.

[159] Z. Pan, X. Liu, and O. Mejabi. A neural-fuzzy system for forecasting. *Journal of Computational Intelligence in Finance*, 5(1):7–15, 1997.

[160] Y-J Park and M-S Song. A genetic algorithm for clustering problems. In *Proc. of 3rd Annual Conf. on Genetic Programming*, pages 568–575, 1998.

[161] M. J. Pazzani. Searching for dependencies in Bayesian classifiers. In D. Fisher and H. Len, editors, *Proc. of 5th Int'l Workshop on Artificial Intelligence and Statistics.* Ft. Lauderdale, FL, 1995.

[162] P. Pudil, J. Novovičová, and J. Kittler. Floating search methods in feature selection. *Pattern Recognition Letters*, 15:1119–1125, 1994.

[163] W. F. Punch, E. D. Goodman, Min Pei, Lai Chia-Shun, P. Hovland, and R. Enbody. Further research on feature selection and classification using genetic algorithms. In *5th Int'l Conf. on Genetic Algorithms*, pages 557–564, Champaign Ill, 1993.

[164] J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, 1986.

[165] J. R. Quinlan. An empirical comparison of genetic and decision-tree classifiers. In *Proc. of 5th Int'l Conf. on Machine Learning*, pages 135–141. Morgan Kaufmann, 1988.

[166] J. R. Quinlan. *C4.5: Programs or Machine Learning*. Morgan Kaufmann, San Mateo, CA, 1993.

[167] J. R. Quinlan. Improved use of continuous attributes in c4.5. *Journal of Artificial Intelligence Research*, 4:77–90, 1996.

[168] J. R. Quinlan and R. M. Cameron-Jones. Oversearching and layered search in empirical learning. In *Proc. of 14th Int'l Joint Conf. on Artificial Intelligence*, pages 1019–1024. Morgan Kaufmann, 1995.

[169] V. V. Raghavan and K. Birchand. A clustering strategy based on a formalism of the reproductive process in a natural system. In *Proc. of 2nd Int'l Conf. on Informatioin Storage and Retrieval*, pages 10–22, 1979.

[170] M. Raymer, W. Punch, E. Goodman, P. Sanschagrin, and L. Kuhn. Simultaneous feature extraction and selection using a masking genetic algorithm. In *Proc. of 7th Int'l Conf. on Genetic Algorithms*, pages 561–567, San Francisco, 1997. Morgan Kaufmann.

[171] M. Richeldi and P. L. Lanzi. Performing effective feature selection by investigating the deep structure of the data. In *Proc. of 2nd Int'l Conf. on Knowledge Discovery & Data Mining*, pages 379–383, 1996.

[172] M. Riedmiller. Advanced supervised learning in multi-layer perceptrons - from backpropagation to adaptive learning algorithms. *International Journal of Computer Standards and Interfaces*, 16(5):265–278, 1994.

[173] J. Rissanen. Stochastic complexity and modeling. *The Annals of Statistics*, 14:1080–1100, 1986.

[174] P. E. Rossi, R. McCulloch, and G. Allenby. The value of household information in target marketing. *Marketing Science*, 15(3):321–340, 1996.

[175] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by error propagation. In D. E. Rumelhart and J. L. McClelland, editors, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, pages 318–362. MIT Press, Cambridge, MA, 1986.

[176] E. Saad, D. Prokhorov, and D. Wunsch. Comparative study of stock trend prediction using time delay, recurrent and probabilistic neural networks. *IEEE Transactions on Neural Networks*, 9(6):1456–1470, 1998.

[177] W. S. Sarle. Neural networks and statistical models. In *Proc. of 19th Annual SAS Users Group International Conference*, pages 1538–1550. SAS Institute, 1994.

[178] C. M. Schaffer and P. E. Green. Cluster-based market segmentation: Some further comparisons of alternative approaches. *Journal of the Market Research Society*, 40(2):155–163, April 1998.

[179] J. D. Schaffer. Multiple objective optimization with vector evaluated genetic algorithms. In J. J. Grefenstette, editor, *Proc. of Int'l Conf. on Genetic Algorithms*, pages 93–100, 1985.

[180] R. E. Schapire. The strength of weak learnability. *Machine Learning*, 5(2):197–227, 1990.

[181] J. Schmid and A. Weber. *Desktop Database Marketing.* NTC Business Books, 1998.

[182] S. Z. Selim and K. Al-Sultan. Simulated annealing algorithms for the clustering problem. *Pattern Recognition*, 10(24):1003–1008, 1991.

[183] R. Setiono and H. Liu. Neural-network feature selector. *IEEE Transactions on Neural Networks*, 8(3):654–662, 1997.

[184] J. W. Shavlik, R. J. Mooney, and G. G. Towell. Symbolic and neural learning algorithms: An experimental comparison. *Machine Learning*, 6(2):111–144, 1991.

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

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

[187] M. Singh and G. M. Provan. A comparison of induction algorithms for selective and non-selective Bayesian classifiers. In *Proc. of 12th Int'l Conf. on Machine Learning*, pages 497–505, Lake Tahoe, CA, 1995. Morgan Kaufmann.

[188] P. Smyth. Clustering using Monte Carlo cross-validation. In *Proc. of 2nd Int'l Conf. on Knowledge Discovery and Data Mining (KDD-96)*, pages 126–133, 1996.

[189] P. H. A. Sneath and R. R. Sokal. *Numerical Taxonomy*. Freeman, London, UK, 1973.

[190] N. Srinivas and K. Deb. Multiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary Computation*, 2(3):221–248, 1994.

[191] S. D. Stearns. On selecting features for pattern classifiers. In *Proc. of 3rd Int'l Conf. on Pattern Recognition*, pages 71–75, Coronado, CA, 1976.

[192] W. N. Street and Y. Kim. A streaming ensemble algorithm (SEA) for large-scale classification. In *Proc. of 7th ACM SIGKDD Int'l Conf. on Knowledge Discovery & Data Mining (KDD-01)*, 2001. 377–382.

[193] W. N. Street, O. L. Mangasarian, and W. H. Wolberg. An inductive learning approach to prognostic prediction. In A. Prieditis and S. Russell, editors, *Proc. of 12th Int'l Conf. on Machine Learning*, pages 522–530, San Francisco, 1995. Morgan Kaufmann.

[194] L. Talavera. Feature selection as a preprocessing step for hierarchical clustering. In *Proc. 16th Int'l Conf. on Machine Learning*, pages 389–397. Morgan Kaufmann, San Francisco, CA, 1999.

[195] T. Terano and Y. Ishino. Interactive knowledge discovery from marketing questionnaire using simulated breeding and inductive learning methods. In *Proc. of 2nd ACM SIGKDD Int'l Conf. on Knowledge Discovery & Data Mining (KDD-96)*, pages 279–282, Menlo Park, CA, 1996.

[196] T. Townsend-Weber and D. Kibler. Instance-based prediction of continuous values. In *Working Notes of the AAAI 94 Workshop on Case-Based Reasoning*, pages 30–35. AAAI Press, 1994.

[197] H. Vafaie and K. Dejong. Genetic algorithms as a tool for feature selection in machine learning. In *4th Int'l Conf. on Tools with Artificial Intelligence*, pages 200–203. IEEE Computer Society Press, 1992.

[198] H. Vafaie and K. Dejong. Robust feature selection algorithms. In *5th Int'l Conf. on Tools with Artificial Intelligence*, pages 356–363. IEEE Computer Society Press, 1993.

[199] H. Vafaie and K. Dejong. Improving a rule learning system using genetic algorithms. In *Machine Learning: A Multistrategy Approach*, pages 453–470. Morgan Kaufmann, 1994.

[200] H. Vafaie and I. Imam. Feature selection methods: Genetic algorithms vs. greedy-like search. In *Proc. of Int'l Conf. on Fuzzy and Intelligent Control Systems*, pages 453–470. Morgan Kaufmann, 1994.

[201] S. Vaithyanathan and B. Dom. Model selection in unsupervised learning with applications to document clustering. In *Proc. 16th Int'l Conf. on Machine Learning*, pages 433–443. Morgan Kaufmann, San Francisco, CA, 1999.

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

[203] Vladimir N. Vapnik. *The Nature of Statistical Learning Theory.* Springer Verlag, New York, 1995.

[204] S. A. Vere. Induction of concepts in the predicate calculus. In *Proc. of Int'l Joint Conf. on Artificial Intelligence -1975*, pages 281–287, 1975.

[205] C. Wallace and P. Freeman. Estimation and inference by compact coding. *Journal of the Royal Statistical Society (B)*, 49:240–265, 1987.

[206] M. Wedel and W. A. Kamakura. *Market Segmentation: Conceptual and Methodological Foundations*, chapter 6–7. Kluwer Academic Publishers, second edition, 1999.

[207] S. M. Weiss and C. A. Kulikowski. *Computer Systems That Learn.* Morgan Kaufmann, San Mateo, CA, 1991.

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

[209] R. L. Wilson and R. Sharda. Bankruptcy prediction using neural networks. *Decision Support Systems*, 11(3):545–557, 1994.

[210] D. H. Wolpert. Stacked generalization. *Neural Networks*, 5:241–259, 1992.

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

[212] X. Yao. Evolving artificial neural networks. *PIEEE: Proceedings of the IEEE*, 87(9):1423–1447, 1999.

[213] G. U. Yule. On the association of attributes in statistics. *Philosophical Transactions, Royal Society London Series A*, 194:257–319, 1900.

[214] T. Zhang, R. Ramakrishnan, and M. Livny. BIRCH: A new data clustering algorithm and its applications. *Data Mining and Knowledge Discovery*, 1(2):141–182, 1997.

[215] E. Zitzler. Evolutionary algorithm for multiobjective optimization: Methods and applications. Technical Report 30, Computer Engineering and Networks Laboratory, Swiss Federal Institute of Technology Zurich, Zurich, Switzerland, 1999.