A CLASS-SPECIFIC ENSEMBLE FEATURE SELECTION

APPROACH FOR CLASSIFICATION PROBLEMS

Except where reference is made to the work of others, the work described in this thesis is my own or was done in collaboration with my advisory committee. This thesis does not include proprietary or classified information.

_____

Caio Soares

Certificate of Approval:

_____	_____
Juan E. Gilbert, Chair Full Professor Computer Science and Software Engineering	Cheryl Seals Associate Professor Computer Science and Software Engineering
_____	_____
Gerry Dozier Associate Professor Computer Science and Software Engineering	George T. Flowers Dean Graduate School

A CLASS-SPECIFIC ENSEMBLE FEATURE SELECTION APPROACH FOR CLASSIFICATION PROBLEMS

Caio Soares

A Thesis

Submitted to

the Graduate Faculty of

Auburn University

in Partial Fulfillment of the

Requirements for the

Degree of

Master of Science

Auburn, Alabama
May 9, 2009

A CLASS-SPECIFIC ENSEMBLE FEATURE SELECTION

APPROACH FOR CLASSIFICATION PROBLEMS

Caio Soares

_____

Signature of Author

_____

Date of Graduation

VITA

Caio Vinicius Soares, son of Darwin Soares and Yara Panzone, was born March 18, 1983 in Sao Paulo, Brazil. In 2001, he graduated from Dr. Phillips High School in Orlando, Florida. He attended Berry College in Rome, Georgia, graduating Magna Cum Laude with Bachelor of Science degrees in Computer Science and Mathematics in 2004. The same year, he entered Graduate School in the Computer Science and Software Engineering Department at Auburn University in Auburn, Alabama.

THESIS ABSTRACT

A CLASS-SPECIFIC ENSEMBLE FEATURE SELECTION

APPROACH FOR CLASSIFICATION PROBLEMS

Caio Soares

Master of Science, May 9, 2009
(B.S. Computer Science, Berry College, 2004)
(B.S. Mathematics, Berry College, 2004)

58 Typed Pages

Directed by Juan E. Gilbert

This research proposes a new feature selection algorithm, Class-specific Ensemble Feature Selection (CEFS), which finds class-specific subsets of features optimal to each available classification in the dataset. Each subset is then combined with a classifier to create an ensemble feature selection model which is further used to predict unseen instances. CEFS attempts to provide the diversity and base classifier disagreement sought after in effective ensemble models by providing highly useful, yet highly exclusive feature subsets. Also, the use of a wrapper method gives each subset the chance to perform optimally under the respective base classifier. Preliminary experiments implementing this innovative approach suggest potential improvements of more than 10% over existing methods.

ACKNOWLEDGEMENTS

The author thanks the following:

Lacey Weaver for her unyielding love, support and encouragement.

Dr. Juan Gilbert and Dr. Gerry Dozier for letting me borrow their great minds and great ideas.

Yara, George, Bruna, Junior, Lysandra, and Duda for being the greatest family a person could ask for.

Style manual or journal used IEEE Computer Society Transactions.

Computer software used Microsoft Word 2003 and Microsoft Word 2007.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

Prediction and classification problems appear throughout our daily lives; physicians predict (diagnose) health problems, gamblers predict (pick) sporting events, stock traders predict (buy/sell) stock, and etc. The storage and processing of past known data gives humans and machines alike the ability to predict and classify unseen or future data. Now, add to it the fact that data storage has become increasingly easier and increasingly inexpensive. The result is a dramatic shift in the mindset of data storage; a shift which considered a one-megabyte database as being very large in 1989 to numerous multi-terabyte databases being mined regularly by the year 2000 [1].

Unfortunately, data processing has not kept up with data acquisition. That is, as computer and database technologies continue to improve, data accumulates in a speed significantly faster than the capacity of data processing [2]. Such data accumulation produces significantly large databases. These databases can be large due to a high number of instances[1], a high number of features[2], or a combination of both. An example of a high-instance database is Google index, which reported having indexed one billion unique URLs by 2000 and one trillion by 2008 [3]. On the other hand, microarray

[1] For the remainder of this work, instances may also be referred to as "cases", "vectors", "samples", or "observations".

[2] For the remainder of this work, features may also be referred to as "variables", "dimensions", "attributes", or "genes" (in case of microarray data).

datasets present a fitting example of high-dimensional datasets, often containing a small number of instances but each with thousands of attributes [4]. Although various methods for dealing with both do exist, this research will focus solely on the issue of high-dimensional data. More specifically, it will do so through the use of *feature selection*, a technique which can reduce the cost of recognition and often provide better classification accuracy by *reducing* the number of features [5]. Up to now, however, this technique has been almost always carried out on the entire dataset at once, assuming that the most salient features for learning will be uniform among all instances, regardless of classification.

So, assume dataset *D*, with instances classified as either *x* or *y*. The hypothesis investigated in this research is two-fold. First, it is believed that instances classified as *x* will have a different set of relevant[3] features than will instances classified as *y*, contradicting the aforementioned methodology, which would suggest that features relevant to the entire dataset must be relevant to instances classified *x* and instances classified as *y*. Moreover, under existing methods, features relevant to a single classification, but not to the entire dataset, may be deemed irrelevant, removed and thus unused at prediction time. Second, given the well documented benefits of ensemble methods [6] [7] [8], an ensemble system which takes advantage of class-specific features should provide the data diversity and classifier disagreement necessary for ensemble systems to succeed, thereby increasing prediction performance (better accuracy). Allowing each classification to have its own set of class-specific optimal features, and

[3] The formal definition of *relevance* as it pertains to features will be detailed in Section 2.2 of this work.

thus each its own ensemble classifier, will create an ensemble system which should allow for a more accurate classification of unseen instances. As an added benefit, new information may be learned as to the relevance of certain features on certain classifications, information which may have been otherwise unknown due to performing feature selection on all instances, and thus, all classifications at once. This would be of particularly good use in problems such as cancer prediction [9] [10], where certain features may be optimal in detecting certain types of cancer, but irrelevant in others.

1.1 Motivation

According to [11], a learning algorithm is good if it produces hypotheses that do a good job of predicting classifications of unseen examples. With that in mind, the primary goal of this research is to increase prediction accuracy, thereby creating a *better* learning algorithm. Note, however, that the focus here is not on the learning algorithm itself, but rather on the pre-processing methodology used to prepare the training data for optimal learning. In fact, research suggests that much of the power in classification comes not from the specific learning method, but from proper formulation of the problems and crafting the representation to make learning tractable [12]. In addition, the same research suggests that performance is roughly equivalent between numerous learning algorithms, across many domains. In other words, the preparation and representation of the data is of the utmost importance given that no single learning algorithm outperforms all others on every domain. In a nutshell, this represents the well-known theorems of "No Free Lunch" [13].

Although a dataset may be well prepared and represented, it may also fall victim to Belman's "curse of dimensionality", a problem which often plagues high-dimensional datasets. The curse of dimensionality refers to the exponential growth of hypervolume as a function of dimensionality [14]. That is, the more dimensions in a dataset, the more representation needed by the predicting model. For classification, this can mean that there are not enough data objects to allow the creation of a model that reliably assigns a class to all possible objects [15]. In the context of Neural Networks (NN), for example, the curse of dimensionality causes networks with irrelevant inputs to behave badly: the dimension of the input space is high, and the network uses almost all its resources to represent irrelevant portions of the space [16]. In [17], Silverman illustrates the difficulty of kernel estimation in high dimensions. As shown in Table 1, even at a small dimensionality of 10, roughly 840,000 samples are required to estimate the density at 0 with a given accuracy [17].

Table 1: Required Sample Size for given Number of Dimensions [17]

Dimensionality	Required Sample Size
1	4
2	19
5	786
7	10,700
10	842,000

In 2003, research conducted by Guyon and Elisseeff reported on gene expression datasets up to 60,000 variables wide, and text classification datasets 15,000 variables wide and nearly 1 million instances deep [18]. Those numbers continue to grow daily, as well as in other fields such as satellite imagery, hyperspectral imagery, financial data,

consumer data, and etc [19]. Improving prediction accuracy within problems of high data dimensionality is the primary motivation for using feature selection.

The key benefit of feature selection is that it directly targets the curse of dimensionality, since it can eliminate irrelevant and redundant features and reduce noise in the dataset. Moreover, as the dimension of the feature space increases so does the probability of over-fitting, and as shown by [20], feature selection provides a powerful means to avoid over-fitting. Feature selection can also lead to a more understandable model, allow the data to be more easily visualized, and with a reduction in dimensionality, can decrease the amount of time and memory required by learning algorithms [15]. The motivation for this research, however, goes deeper into the methodology of feature selection, a topic which will be further explored in the subsequent section.

1.2 Problem Description

Although there are a plethora of feature selection algorithms, all differing in some shape or form, they all share a commonality; they all attempt to reduce the number of features by selecting an *optimal* subset of the original feature set. This subset, however, almost always contains features which are assumed to be optimal to the *entire* dataset. This then leads to the assumption that the most optimal features to the dataset must be optimal for instances across all classifications. Therein lies the problem and thus the basis for this research.

The hypothesis is that by allowing the feature selection algorithm to focus its attention on the prediction accuracy of a single classification at a time, the algorithm will produce a separate subset of features optimal to each classification. These separate subsets can then be utilized to create separate classification models, thereby taking advantage of another technique proven to improve classification accuracy; ensemble learning (or in this case, ensemble feature selection). In summary, this research will attempt to improve on the problem of prediction accuracy by proposing a unique approach to performing ensemble feature selection

1.3 Goals and Contributions

The research conducted and detailed through this work will attempt to achieve the following comprehensive goals:

1. To design and develop a unique and effective approach to feature selection.

2. To examine the potential benefits of class-specific ensemble feature selection.

3. To enhance dataset understandability with the creation of class-specific optimal subsets as opposed to dataset-specific optimal subsets.

4. To test the effectiveness of class-specific ensemble feature selection on an existing classification problem.

The immediate contribution of this research will be in the introduction of a unique algorithm to the areas of feature selection, ensemble learning, and machine learning.

However, as dataset sizes continue to grow in every discipline, the proposed algorithm can further contribute in a much more widespread manner as it is data independent. Given its class-specific design, the algorithm may have even farther reaching implications to fields in science, medicine, vision, finance, etc.

1.4 Organization

The outline of this thesis first began with a brief introduction on the methods explored in this research. The remainder of this thesis is organized as described below.

Chapter 2 explains feature selection in detail. Following a brief introduction to feature selection, common misconceptions and the definition of relevance are provided. The use of feature selection in unsupervised learning is then discussed, concluding with thorough explanations of the different types of feature selection algorithms and existent work in ensemble feature selection.

Chapter 3 provides a thorough explanation of the Class-specific Ensemble Feature Selection (CEFS) algorithm proposed in this research. A detailed description of the suggested implementation is also provided.

Chapter 4 presents an initial implementation of the suggested algorithm, along with results attained from a preliminary experiment. Moreover, experiment parameters, analysis, and conclusions are discussed. Lastly, Chapter 5 concludes with final observations and analysis of the conducted research, and possible future work.

CHAPTER 2

FEATURE SELECTION

Many machine learning algorithms, including top-down induction of decision tree algorithms such as ID3 [21], C4.5 [22], and CART [23], and instance-based algorithms, such as IBL [24] [25], are known to lose prediction accuracy when faced with many features unnecessary for predicting the output, i.e. *irrelevant* features [26]. These are features which contain almost no useful information for the learning task at hand. For example, students' ID numbers are irrelevant for predicting students' GPAs [15]. On the other hand, algorithms such as Naïve-Bayes [27] [28] [29] are robust with respect to irrelevant features, but their performance may degrade quickly if correlated features, i.e. *redundant* features, are added [26]. Such features duplicate much or all of information contained in one or more other attributes. For instance, the purchase price of a product and the amount of sales tax paid contain much of the same information [15]. Feature selection targets those very problems as it is a technique which reduces the number of features and removes irrelevant, redundant or noisy data by selecting a subset of the original feature set [2]. Most feature selection methods follow a four step process: subset generation, subset evaluation, stopping criterion, and result validation (Figure 1) [30].

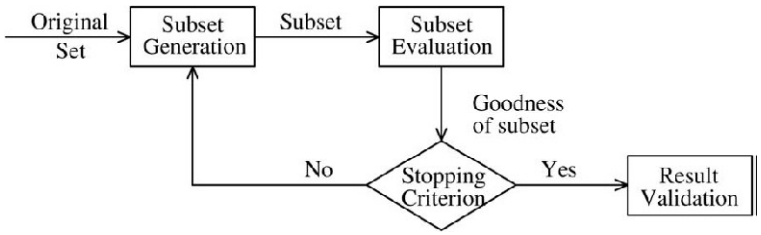

Figure 1: Feature Selection Process [30]

Beginning with subset generation, a selected search strategy produces candidate feature subsets. Each subset is then evaluated and compared to others according to a given evaluation criterion. The best subset is kept and this process is repeated until a stopping criterion is reached, at which point, the selected subset is validated using the pre-selected classifier. Formally, feature selection can be defined as follows [2]:

Definition 1 (feature selection) Let Y be the original set of features, with cardinality n. Let d represent the desired number of features in the selected subset X, $X \subseteq Y$. Let the feature selection criterion function for the set X be represented by $J(X)$. Without lost of generality, let us consider a higher value of J to indicate a better feature set. Formally, the problem of feature selection is to find a subset $X \subseteq Y$ such that $|X| = d$ and

$$J(X) = \max_{Z \subseteq Y, |Z|=d} J(Z).$$

Although optimal feature selection is typically intractable [26] and many problems related to it have been shown to be NP-hard [31] [32], some feature selection algorithms have proved to be significantly more stable than others [33] [34]. Furthermore, feature selection has been shown to yield benefits such as facilitating data visualization and data understanding, reducing measurement and storage requirements, reducing training and utilization times, and defying the curse of dimensionality to improve prediction performance [18]. As a result, feature selection can be found in many areas of research such as bioinformatics [35] [36] [37] [38], machine learning [35] [39] [40] [41], text classification/mining [42] [43] [44] [45] [46], case-based reasoning [47],

statistics [48] [49] [50], security [44] [51], pattern recognition [5] [36] [52] [53], data mining [54] [55] [56], and various others. In addition, numerous reviews and comparative studies on existing feature selection algorithms have been conducted [2] [30] [57] [58] [59].

Feature selection methods can be structured into four main factions: *filter*, *wrapper*, *embedded*, and *hybrid* methods [2] [38]. Filter and wrapper methods are most widely known and used. Embedded methods are also frequently used, but typically with the user having no knowledge that feature selection is being employed. Hybrid methods are fairly new on the scene, so they have not yet fully made their mark.

Filter methods rank each feature according to some univariate metric, and only the highest ranking features are used while the remaining low ranking features are eliminated [38]. Some commonly used metrics for filter methods include Information Gain, Chi-Squared [42], and Pearson Correlation Coefficient [60]. Since such metrics are used as the evaluation criterion to select an optimal subset of features, it is important to point out that filter methods do not involve the use of a learning algorithm as evaluation criterion, defining its main distinction from wrapper methods. As such, filter methods do not inherit bias from learning algorithms and are also more computationally efficient than wrapper methods [2].

Popularized by Kohavi and John in [26] wrapper methods utilize a learning algorithm to assess the *goodness* of a subset of features [18]. Whereas a filter method will use metrics to determine the usefulness of each feature, wrapper methods will use a learning method to determine the usefulness of a candidate set of features. At the end of

the process, not only will the algorithm produce an optimal set of features, but one which will perform most optimally under the aforementioned learning algorithm. Well known wrapper algorithms include Sequential Forward Selection, Sequential Backward Selection [5], and using a hill-climber to optimize feature selection [26]. Although wrapper methods have shown to be more computationally expensive than filter methods due to their use of a separate learning algorithm for evaluation, research has also shown that wrapper methods tend to give superior performance as feature subsets are better suited to the predetermined learning algorithm [2].

Embedded methods perform feature selection as part of the learning algorithm itself. That is, during the process of the learning algorithm, the algorithm decides which attributes to use and which to ignore [15]. Decision Trees are perhaps the most well known example of embedded methods [21] [22] [23].

To take advantage of the benefits of filter and wrapper models, and to avoid the pre-specification of a stopping criterion, hybrid methods have been proposed [61]. A typical hybrid method uses both an independent measure and a learning algorithm to evaluate feature subsets [2].

As such, this work will focus most of its attention on filter and wrapper algorithms, the latter, being implemented on the proposed algorithm. The remainder of this chapter is organized as follows. First, common misconceptions of the feature selection are addressed, followed by an in depth discussion on the subject of relevance as it pertains to feature selection. Next, a brief synopsis of the use of feature selection in unsupervised learning is given. Detailed descriptions of each type of feature selection

algorithm are then provided. Finally, previous research conducted in ensemble feature selection is summarized and explained, concluding with a few remarks regarding feature selection as described in this chapter.

2.1 Misconceptions

Techniques such as dimensionality reduction, feature extraction, feature construction, feature weighting, feature creation and others are often misused as synonymous to feature selection. It should be pointed out that feature selection strictly selects a subset of the existing feature set, without creation of new features [26] [62] [63]. However, to attain a better understanding of feature selection, a brief explanation of some of the aforementioned techniques is noteworthy.

The first two techniques, dimensionality reduction and feature extraction, go hand in hand as various algorithms are often interchangeably characterized under both. Tan *et al* comments that dimensionality reduction commonly uses techniques from linear algebra to project the data from a high-dimensional space into a lower-dimensional space [15]. Similarly, Jain *et al* describes feature extraction algorithms as algorithms which create *new features* based on transformations or combinations of the original feature set [5]. For example, Principal Component Analysis (PCA) finds new attributes which are linear combinations of the original attributes, are orthogonal to each other, and capture the maximum amount of variation in the data [15]. In other words, PCA is an orthogonal transformation of the coordinate system, which is obtained by projection onto the *principal components*, or *features*, of the data. Since a small number of principal

components are often sufficient, a reduction in dimensionality occurs [64]. Further reviews of PCA literature can also be found at [65] and [66].

Now consider another example; suppose you have a set of photographs, where each photograph is to be classified as to whether or not it contains a human face. By using a feature extraction algorithm, pixels can be transformed to provide higher-level features, such as edges and areas highly correlated with the presence of human faces. Other examples include applying a Fourier transform to times series data to identify underlying frequencies in order to detect certain periodic patterns [15] and using Independent Component Analysis (ICA) in conjunction with Support Vector Machines (SVM) in order to improve prediction accuracy in series forecasting [67].

Other techniques such as feature weighting [15] and feature construction [18] also provide similar distinctions, but for the brevity of this research, will not be discussed at this time. In summary, feature selection is not to be confused with other techniques which go beyond the original feature set, creating new features; that is, feature selection algorithms always yield a direct subset of the original feature set.

2.2 Definition of Relevance

Relevance, as it pertains to feature selection, is a bit of a loaded term. Defining it is not trivial nor is it widely agreed upon. In fact, numerous authors have provided a number of different definitions for relevance. A brief summary of some of these definitions will be provided in the subsequent paragraphs. Detailed explanations and further definitions of relevance can be found in works such as [26] by Kohavi and John,

[68] by Blum and Langley, [36] by Nilsson *et al*, and most recently in [69] by Bell and Wang.

Although the word relevance is often used casually and without formal definition in most feature selection studies, its definition is as important as its use. At the heart of this matter lies the question of relevance vs. usefulness. Selecting only the most relevant features will often produce suboptimal results, especially if the features are redundant. Conversely, a subset of useful features may leave out many redundant, yet relevant, features [18]. In other words, the relevance of a feature does not imply that it should be in the optimal feature subset, while irrelevance does not imply it should not be in the optimal feature subset [26]. For clarification purposes, provided next are some formal definitions acquired from [26], [36], and [68] which may aid in understanding the role of relevance in feature selection.

Definition 2 (conditional independence) A variable X_i is conditionally independent of a variable Y given (conditioned on) the set of variables $S \subset X$ iff it holds that

$$P\big(p(Y|X_i,S) = p(Y|S)\big) = 1.$$

This is denoted $Y \perp X_i|S$.

Conditional independence is a measure of *irrelevance*, but it is difficult to use as an operational definition since this measure depends on the conditioning set S [36]. The definitions of strong and weak relevance, defined next, will help refine what it means for a feature to be considered irrelevant.

Definition 3 (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

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

Definition 4 (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

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

In other words, a feature X is strongly relevant if the removal of X alone will yield a decrease in the performance of an optimal classifier. Conversely, a feature X is weakly relevant if it is not strongly relevant and there exists a subset of features, S, such that the performance of a classifier on S is worse than the performance on $S \cup \{X\}$. Thus, a feature is *relevant* if it is either weakly or strongly relevant, otherwise it is *irrelevant* [26]. Figure 2 provides a useful visual representation of the feature subset space divided into irrelevant, weakly relevant and strongly relevant feature subsets [26].

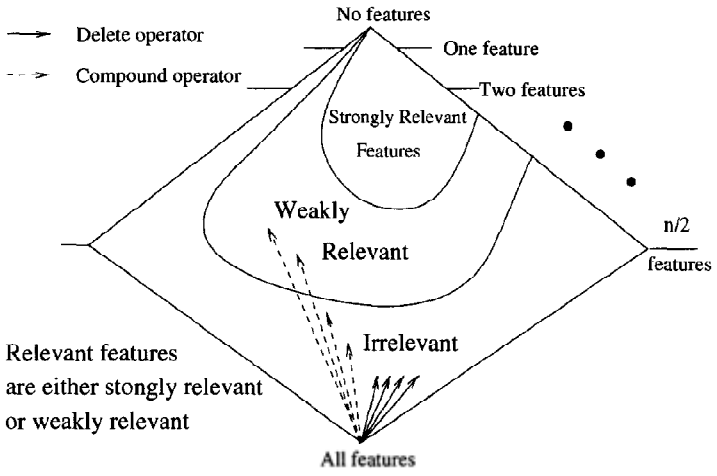

Figure 2: Feature Subset Space [26]

Recall, however, that a distinction between the relevance of a feature and the usefulness of a feature is still necessary. As such, the following definition derived from [70] will provide clarification.

Definition 5 (incremental usefulness) Given a sample of data S, a learning algorithm L, and a feature set A, feature x_i is incrementally useful to L with respect to A if the accuracy of the hypothesis that L produces using the feature set $\{x_i\} \cup A$ is better than the accuracy achieved using just the feature set A.

Accordingly, the definition of incremental usefulness will be the focal point for determining whether or not a feature should be included in the optimal feature subset. This is further apparent given the fact that the proposed algorithm later described in this research will employ a wrapper method, which tailors directly to the learning algorithm used to evaluate each feature subset. Therefore, features will be assessed according to their usefulness with respect to the learning algorithm rather than their overall relevance.

As a final illustration on the issue of relevance vs. usefulness, refer to an example provided in [26]. One of the artificial datasets (m-of-n-3-7-10) represents a symmetric target function, implying that all features should be ranked equally by any filtering method. However, Naïve-Bayes improves if a single feature (any one of them) is removed. Consequently, that is the precise motivation behind the selection of a wrapper method over a filter method in the implementation of the proposed algorithm in this research (Chapter 3). Please note, however, that relevance according to these definitions does not imply membership in the optimal feature subset, and that irrelevance does not imply that a feature cannot be in the optimal feature subset [26]. The underlying purpose among an of this is to design the algorithm in such a way that it will be allowed to choose usefulness over relevance.

2.3 Feature Selection in Unsupervised Learning

Although the use of feature selection in unsupervised learning will not be implemented in this research, a brief discussion of the subject is important given the amount of attention it has received [2] [71] [72] [73] [74]. Unsupervised learning, more specifically clustering [75] [76], groups instances based on the information describing the instance or based on their relationships. The goal is to obtain groups with instances similar or related to one another and different from instances in other groups. The greater the similarity (or homogeneity) within a group and the greater the difference between groups, the better the clustering [71].

Wrapper methods used in conjunction with clustering evaluate the goodness of a feature subset by the quality of the clusters resulted from applying the clustering algorithm on the candidate feature subset. A number of heuristic measures exist for estimating the quality of clustering results, such as cluster compactness, scatter separability, and maximum likelihood [2]. Further work on developing dependent criteria in feature selection for clustering can also be found in [54], [72], and [73]. Since filter methods don't depend on the use of a separate learning algorithm for evaluation, the methodology for using filter feature selection on unsupervised learning does not differ from that of supervised learning. Given the infancy of this research area, advances continue to occur.

2.4 Feature Selection Methods

Feature selection methods can be grouped into one of four groups: filter, wrapper, embedded, and hybrid methods [2] [38]. The subsequent sections will describe each method in detail.

2.4.1 Filter Methods

Coined by John, Kohavi and Pfleger in [77], filter methods have their name because they *filter* out irrelevant features before induction occurs. The process uses general characteristics of the training set to select some features and exclude others. Since filtering methods do not involve the use of a learning algorithm to evaluate candidate sets, they can be combined with any learning algorithm after the filtering is complete. Moreover, filter methods are a computationally effective form of data pre-processing, especially when compared to wrapper methods [38] [68].

Figure 3 describes a generalized form of a filter algorithm, provided by [2]. Given dataset D, begin with a given subset S_0 (an empty set, a full set, or any randomly selected subset) and search through the feature space using a particular search strategy. Each generated subset S is evaluated by an independent measure M and compared with the previous best. If better, it's regarded as the current best subset. The search iterates until a predefined stopping criterion is reached. The algorithm outputs the last current best subset S_{best} as the final feature subset [2].

Filter Algorithm

input: $D(F_0, F_1, ..., F_{n-1})$ // a training data set with N features
S_0 // a subset from which to start the search
δ // a stopping criterion

output: S_{best} // an optimal subset

```
01 begin
02     initialize: S_best = S_0;
03     γ_best = eval(S_0, D, M); // evaluate S_0 by an independent measure M
04     do begin
05         S = generate(D); // generate a subset for evaluation
06         γ = eval(S, D, M); // evaluate the current subset S by M
07         if (γ is better than γ_best)
08             γ_best = γ;
09             S_best = S;
10     end until (δ is reached);
11     return S_best;
12 end;
```

Figure 3: Generalized Filter Algorithm [2]

One of the simplest filtering processes is to evaluate each feature individually based on its correlation with the target function and then select the *k* features with the highest values. This method has been shown to achieve good results in text categorization tasks [78] [79], often used in combination with either a Naïve-Bayesian classifier or a nearest neighbor classifier [68]. Other widely used metrics include Information Gain, Odds Ratio, Log Probability Ratio [42], FOCUS [80], RELIEF [81] [82], Potential Difference [83], Pearson Correlation Coefficient [60], etc. RELIEF, for instance, assigns a "relevance" weight to each feature, which represents the relevance of the feature to the target concept. It samples instances randomly and updates the relevance values based on the difference between the selected instance and the nearest instances of the same and opposite class ("near-hit" and "near-miss") [26]. The algorithm does require the problem to only contain two classes however. Another example, FOCUS, looks for minimal combinations of attributes that perfectly

discriminate among the classes. It begins by looking at each feature individually, goes on to pairs of features, triples, and so forth, stopping only when it finds a combination which generates pure partitions of the training set.

There are some drawbacks to filter methods however. First, most filtering methods require the pre-selection of a set number of features to be chosen. If the number is too high, irrelevant features will be kept and accuracy may suffer. If the number is too low, useful features may not be selected, once again affecting accuracy. Some remedies to this problem include using a hold-out set to test for the best *k*, or to use a hill-climber or some other evolutionary computation algorithm to find an optimal *k*. These solutions, however, will negatively impact what is perhaps the biggest benefit of a filtering algorithm, its relative speed. A second drawback is that filtering algorithms may miss features that would otherwise be useful to the learning algorithm which will predict unseen instances. Since the algorithm bases its selection purely on metrics, it may miss features deemed useful by some learning algorithms and perhaps irrelevant in others. This is why wrapper methods are widely known to outperform filter methods (in terms of prediction accuracy) [2] [38].

2.4.2 Wrapper Methods

Wrapper methods occur outside the basic learning algorithm, but also use said learning algorithm as a subroutine, rather than just as a post-processor. For this reason, John, Kohavi, and Pfleger [77] refer to them as *wrapper* methods. Each candidate subset is evaluated by running the selected data through the learning algorithm and using the

estimated accuracy of the resulting classifier as its evaluation metric. As aforementioned, the biggest benefit to using wrapper methods is their tendency to outperform (prediction accuracy) their filter and embedded counterparts. The general argument is that the classifier which will use the feature subset should provide a better estimate of accuracy than a separate metric that may have an entirely different bias [68].

Figure 4 provides the pseudocode for a generalized wrapper algorithm [2], which is quite similar to the generalized filter algorithm (Figure 3 – Section 2.4.1) except that it uses a predefined learning algorithm *A* instead of an independent measure *M* for the evaluation of each candidate subset. For each generated subset *S*, the algorithm evaluates its goodness by applying the learning algorithm to the data with subset *S* and evaluating the accuracy. Thus, different learning algorithms may produce different feature selection results. Varying the search strategies via the function generate(*D*) and learning algorithms (*A*) can result in different wrapper algorithms. Since learning algorithms are used to control the selection of feature subsets, the wrapper model tends to give superior performance as feature subsets found are better suited to the predetermined learning algorithm. As a result, it's also more computationally expensive than a filter method [2].

Wrapper Algorithm

input: $D(F_0, F_1, ..., F_{n-1})$ // a training data set with N features
S_0 // a subset from which to start the search
δ // a stopping criterion

output: S_{best} // an optimal subset

```
01 begin
02     initialize: S_best = S_0;
03     γ_best = eval(S_0, D, A); // evaluate S_0 by a mining algorithm A
04     do begin
05         S = generate(D); // generate a subset for evaluation
06         γ = eval(S, D, A); // evaluate the current subset S by A
07         if (γ is better than γ_best)
08             γ_best = γ;
09             S_best = S;
10     end until (δ is reached);
11     return S_best;
12 end;
```

Figure 4: Generalized Wrapper Algorithm [2]

Similar to many wrapper method variations such as the brute force method, branch and bound, sequential backward/forward search, the sequential floating search method, etc. Variations can also be found in the learning algorithm. Neural networks, Bayesian networks, and the SVM are often applied on different wrapper problems [84]. Research done in [56] also discusses overfitting and dynamic search in regards to wrapper methods.

2.4.3 Embedded Methods

Much like wrapper methods, embedded methods interact directly with a specific learning algorithm. In other words, the feature selection algorithm is built (*embedded*) into the classifier model itself rather than using the classifier to evaluate candidate feature sets. Moreover, these methods have the advantages that they interact directly with the classifier while also being less computationally expensive than wrapper methods [38]. A

generalized form of this technique is not provided here because embedded feature selection methods may be highly different in implementation depending on the learning algorithms encompassing them (e.g. decision tree vs. SVM).

Recursive partitioning methods for induction such as decision trees [21] [22] [23], for instance, employ a greedy search through the space of decision trees, at each stage using an evaluation function to select the feature which has the best ability to discriminate among the classes. They partition the data based on this feature and repeat the process on each subset, extending the tree downward until no further discrimination is possible. Other embedded methods include Separate-and-Conquer for learning decision lists [85] and Support Vector Machines of Recursive Feature Elimination [35].

2.4.4 Hybrid Methods

Hybrid feature selection algorithms combine the use of filter and wrapper methods in an attempt to exploit the benefits of both. By combining both techniques at different stages, hybrids are able to take advantage of the speed of a filter method and the accuracy of a wrapper method [37] [61]. Other implementations may also include the use of an embedded method, such as a decision tree, instead of a wrapper method [86] [87].

Similar to the generalized algorithms previously provided, Figure 5 describes the pseudocode for a generalized hybrid algorithm [2]. The algorithm first uses a filter method to find the best subset for a given predetermined cardinality and then uses the learning algorithm to select the final best subset among the best subsets across different cardinalities. The algorithm begins with a given subset S_0 (typically an empty set in

sequential forward selection) and iterates to find the best subsets at each increasing cardinality. In each round for a best subset with cardinality c, it searches through all possible subsets of cardinality $c + 1$ by adding one feature from the remaining features. Each newly generated subset S with cardinality $c + 1$ is evaluated by an independent measure M and compared with the previous best. If S is better, it becomes the current best subset S'_{best} at level $c + 1$. At the end of each iteration, a learning algorithm A is applied on S'_{best} at level $c + 1$ and the quality of the mined result is compared with that from the best subset at level c. If S'_{best} is better, the algorithm continues to find the best subset at the next level; otherwise, it stops and outputs the current best subset as the final best subset. The quality of results from a learning algorithm provides a natural stopping criterion in this model [2].

Hybrid Algorithm

input: $D(F_0, F_1, ..., F_{n-1})$ // a training data set with N features
S_0 // a subset from which to start the search
output: S_{best} // an optimal subset

```
begin
    initialize: S_best = S_0;
    c_0 = card(S_0); // calculate the cardinality of S_0
    γ_best = eval(S_0, D, M); // evaluate S_0 by an independent measure M
    θ_best = eval(S_0, D, A); // evaluate S_0 by a mining algorithm A
    for c = c_0 + 1 to N begin
        for i = 0 to N − c begin
            S = S_best ∪ {F_j}; // generate a subset with cardinality c for evaluation
            γ = eval(S, D, M); // evaluate the current subset S by M
            if (γ is better than γ_best)
                γ_best = γ;
                S'_best = S;
        end;
        θ = eval(S'_best, D, A); // evaluate S'_best by A
        if (θ is better than θ_best);
            S_best = S'_best;
            θ_best = θ;
        else;
            break and return S_best;
    end;
    return S_best;
end;
```

Figure 5: Generalized Hybrid Algorithm [2]

For example, [84] begins by pre-processing the dataset by using two different filter methods, F-score and Information Gain. The separate feature subsets are then combined and processed through a Sequential Floating Wrapper method, which yields the final feature subset. A Support Vector Machine (SVM) then utilizes the resulting feature subset to compute the classification accuracy.

2.5 Feature Selection in Ensemble Methods

Feature selection methods discussed up to this point employ the use of a single classifier. An ensemble system, on the other hand, is composed of a set of multiple classifiers and performs classification by selecting from the predictions made by each of the classifiers [15]. Since wide research has shown that ensemble systems are often more accurate than any of the individual classifiers of the system alone [6] [7] [8], it is only natural that ensemble systems and feature selection would be combined at some point.

2.5.1 Ensemble Methods

The main goal of an ensemble is to construct multiple classifiers from the original data and then aggregate their predictions when classifying unknown instances. Figure 6 shows a basic view of an ensemble method [15]. As depicted, three main steps exist: training set generation, learning, and integration. Step 1 begins with the original training set D. From this training set, t data subsets are created (D_1, D_2, …, D_t). Bagging and boosting are common ways to accomplish this step [15]. Then in Step 2, t base classifiers are generated (C_1, C_2, …, C_t). These classifiers may all be the same, all different, or

contain any combination of the same or different classifiers. Each classifier C_i is trained using the subset D_i. Finally in Step 3, the prediction of each classifier is combined in a predetermined way to produce the resulting classification.

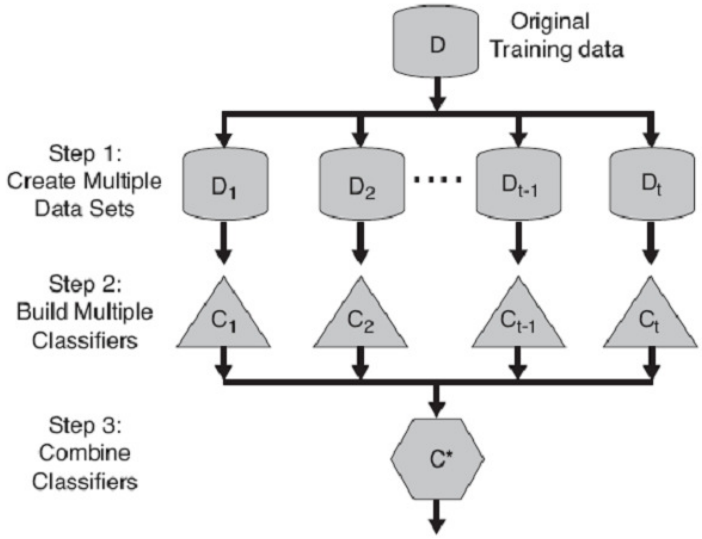

Figure 6: A logical view of the ensemble learning method [15]

Two primary approaches exist to the integration phase: combination and selection. In the combination approach, the base classifiers produce their class predictions and the final outcome is composed using those predictions. In the selection approach, one of the classifiers is selected and the final prediction is the one produced by it [88]. The simplest and most common combination method is voting, also known as majority voting. In voting, the classification predicted by a base classifier is counted as a vote for that particular class value. The class value with the most votes becomes the final classification [88]. A simple and popular selection method is cross-validation majority (CVM) [89], which estimates the accuracy of each base classifier using cross-validation

and selects the classifier with the highest accuracy. Although CVM is a selection method which chooses one classifier for the whole data space, more sophisticated selection methods which estimate local accuracy [90] or meta-level classifiers do exist [91]. Perhaps the most commonly used integration techniques are voting [6], simple and weighted averaging, and a posteriori [92] [93].

According to [94], the main objective when building the base classifiers is to maximize the coverage of the data, which is the percentage of the instances which at least one base classifier can classify correctly. Reaching coverage greater than the accuracy of the best classifier, however, requires diversity among the base classifiers [8] [92] [93]. Although research is always ongoing on new approaches to increase diversity, some methods include training on different subsets of the training set, using different learning algorithms, injecting randomness, and training on different sets of input features [95]. The latter is where ensemble feature selection can be successfully applied.

2.5.2 Ensemble Feature Selection

An effective way of generating a diverse, yet accurate, ensemble of base classifiers is to use ensemble feature selection [88]. To recall, theoretical and empirical research has shown that an efficient ensemble should consist not only of high accuracy classifiers, but classifiers which also err in different parts of the input space [6]. Providing different feature subsets allow base classifiers to make their classification errors in different subareas of the instance space. While feature selection algorithms attempt to find an optimal feature subset for the learning algorithm, ensemble feature

selection has the additional goal of finding a set of feature subsets which will promote disagreement among the base classifiers [88] [96]. This is perhaps the most important point in understanding the motivation and goals of the research presented in this work. The approach proposed in Chapter 3 will attempt to provide the aforementioned disagreement among the base classifiers, but with a set of highly optimal class-specific feature subsets.

Various successful attempts have been made at implementing ensemble feature selection. In [97], Ho proposes a technique called Random Subspace Method (RSM), which randomly selects a predetermined number of features from the original feature set. This is repeated *n* times to create *n* feature subsets which are used to generate the *n* base classifiers used in the ensemble. Ho goes on to show that RSM ensembles can reach effective results presumably because the lack of accuracy in the ensemble members is compensated for by the diversity created by the RSM [97]. In another implementation, Optiz begins by using RSM to generate an initial population of feature subsets, which he then uses and optimizes in his Genetic Algorithm (GA). Optiz uses GAs in conjunction with Neural Networks (NN) as a wrapper method to find optimal feature subsets, which he then uses as training data to build an ensemble model of NNs. Optiz comments that the initial population, acquired through RSM, was surprisingly good and produced better ensembles on average than the popular and power ensemble approaches of bagging and boosting [96]. Other ensemble feature selection studies have included implementations such as a Hill-Climber and Bayesian Classifiers with cross-validation integration [88], GA and *k* Nearest Neighbor classifiers [98] [99], sequential-search-based strategies and

Bayesian Classifiers [100], and comparative ensemble feature selection studies such as [101].

Perhaps the most related work to the algorithm proposed in this research is the work done by Vale *et al* in [102]. In that study, the authors implement a filter method with an emphasis on class-based feature selection to generate the feature subsets used by each base classifier. The use of a filter method is perhaps the most evident shortcoming of the implementation as wrapper methods have often shown to provide more accurate results. In addition, the authors implement a combination based ensemble integration technique, although the specific details of the combination technique are not provided. Depending on the specific combination technique, further improvement can be seen here as well.

In summary, Chapter 2 has provided an in depth description of feature selection. First, common misconceptions and various definitions of relevance were discussed. This was followed by the uses of feature selection in unsupervised learning and explanations of the different types of feature selection techniques. Lastly, the use of feature selection in combination with ensemble methods was reviewed.

CHAPTER 3

CLASS-SPECIFIC ENSEMBLE FEATURE SELECTION (CEFS)

A thorough summary of the intricacies of feature selection was presented in Chapter 2. This included an overview of some of the benefits and drawbacks of certain methods and the latest work being done in the field. The algorithm proposed in this research, named Class-specific Ensemble Feature Selection (CEFS), will seek to exploit some of those short comings, as well as, build upon the most recent successful findings.

The CEFS algorithm will focus mainly on improving prediction accuracy on classification problems. As an added benefit, the class-specific design will select features optimal to each separate classification, thereby providing more information and understanding in regards to the most useful features to each classification and to the model.

Presented next is the design of CEFS (Figure 7). First, assume dataset *D*, which contains *n* different classifications. CEFS will use a wrapper method such as Sequential Backward Selection (SBS) [47] or a Genetic Algorithm (GA) [96] to find *n* optimal feature subsets. That is, the feature selection algorithm will be run *n* times, each time searching for the subset of features which will maximize the classification accuracy, not of the entire dataset *D*, but of one of the *n* classifications, thus producing *n* optimal *class-specific* subsets. This is done by evaluating the prediction accuracy of the classifier on a

single classification at a time, instead of the prediction accuracy of the classifier on the entire test set (which groups all classifications together). To clarify, the training set and test set do not ever change, they still contain all of the instances as though the algorithm were going to predict as normal. The facet which changes is the evaluation criterion. By using only the prediction accuracy of a single classification at a time, the algorithm is able to find an optimal set of class-specific features.

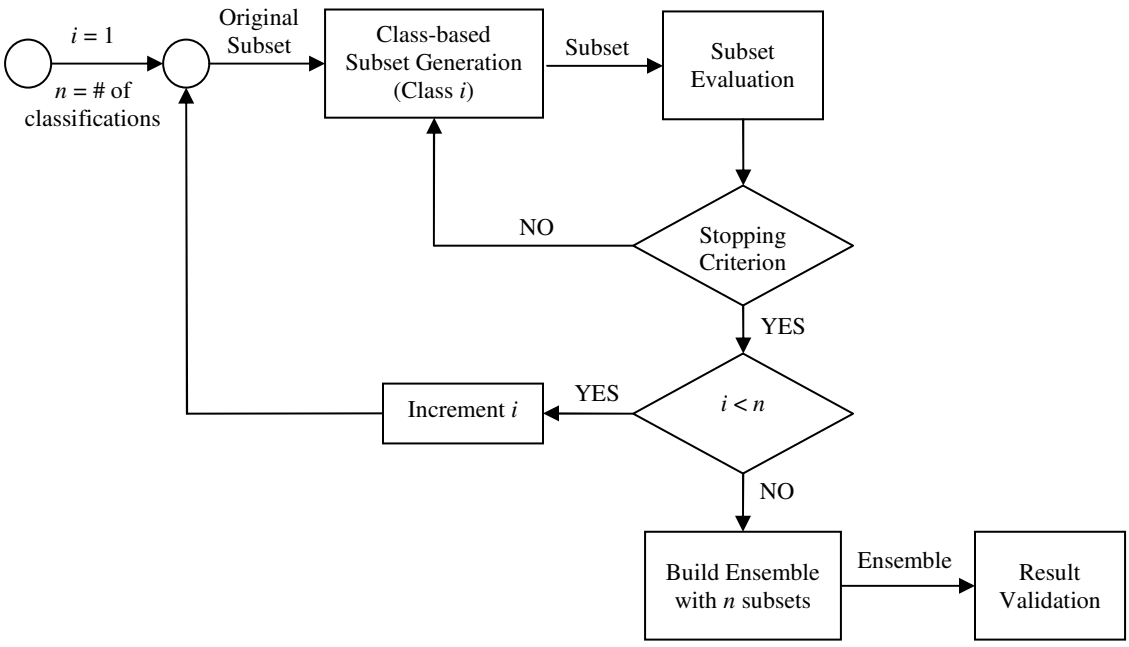

Figure 7: CEFS Procedure

The classifier used for the wrapper method evaluation is a Naïve Bayesian Classifier (NBC) [28] [29], which is perfectly suited for this research, as lazy learners tend to benefit greatly from feature selection [103]. Then, each class-specific subset will be used to build a separate base classifier, creating *n* classifiers in an ensemble feature

selection model. Finally, the ensemble will use a combination technique to predict the outputs of the unseen instances.

The algorithm itself is similar to the one presented in [102], but with two main improvements. First, a wrapper method is used for feature selection instead of a filter method. Since the features will be selected using the classifier as its evaluation criteria, this should produce features more useful to predict each class, which most importantly, should lead to a higher accuracy as has been typically shown to occur with the use of wrapper methods over filter methods [2]. Refer to Section 2.2 for a more in depth discussion of the issue of relevance vs. usefulness in regards to feature selection. Using a wrapper method over a filter method will increase the overall running time of the algorithm, however, this is an acceptable drawback if it will improve prediction accuracy. The second distinction is the integration strategy used for the ensemble machine. The research presented in [102] uses a fusion-based integration strategy, although the details on the specific implementation of this strategy are not provided. The research proposed in this work will not implement a specific ensemble integration technique, but rather explore the potential accuracy if an appropriate technique was selected.

CHAPTER 4

IMPLEMENTATION

In order to gage the usefulness and effectiveness of the algorithm and to test the hypothesis aforementioned in this text, an implementation of a Class-specific Ensemble Feature Selection (CEFS) is necessary. This Chapter provides preliminary results of an initial implementation of CEFS. Moreover, this initial implementation is tested on a limited amount of data to show a proof of concept. Further experimentation will be necessary to fully test the impact the proposed algorithm can have on prediction accuracy. The remainder of the Chapter will be arranged as follows. First, a brief description of the proposed hypothesis is provided. Next, experimental data, tools, and procedure are discussed. Finally, preliminary results, findings and conclusions are detailed.

4.1 Proposed Hypothesis

To reiterate the proposed hypothesis from Chapter 1, the hypothesis examined in this research is two-fold. First, is the theory that instances classified as *x* will have a different set of useful features than will instances classified as *y*, contradicting the aforementioned methodology, which suggests that features useful to the entire dataset must be useful to instances classified *x* and instances classified as *y* and vice-versa.

Second, utilizing class-specific feature subsets to assemble and train ensemble base classifiers will create a system which will further improve prediction accuracy over existing models.

4.2 Preliminary Experiment

An initial preliminary experiment was conducted in small scale to show, as a proof of concept, that the proposed approach is successful in confirming the aforementioned hypothesis and achieving the suggested goals.

4.2.1 Data

The dataset used in this experiment was acquired from the UCI Machine Learning Repository [104]. The breast cancer dataset was chosen, containing 286 instances with two possible classifications: recurrence (85) and no recurrence (201). Furthermore, each instance contained nine unique features: age, menopause, tumor-size, inv-nodes, node-caps, deg-malig, breast side, breast-quad, and irradiat. A Leave-One-Out cross-validation scheme, which is detailed in Section 4.2.3, was used to partition the data.

4.2.2 Tools

The dataset used in this experiment was stored and accessed using Microsoft Access 2007. The CEFS algorithm was written using Java JDK 1.6 as the programming language of choice and jGRASP as the IDE.

4.2.3 Procedure

Given the relatively small number of instances and imbalance between classes in the dataset (29.7% recurrence vs. 70.3% no recurrence), a leave-one-out cross-validation scheme was selected. In a leave-one-out approach, each test set contains only one instance from the overall dataset. This is repeated until all instances have been used as a test set. The accuracy is then the average of the number of instances correctly predicted over the total number of instances in the dataset. This approach has the advantage of utilizing as much data as possible for training (i.e., $n - 1$ instances where n is the number of instances in the original dataset). In addition, test sets are mutually exclusive and they effectively cover the entire dataset [15].

The CEFS algorithm implemented in this experiment uses a Sequential Backward Selection algorithm (SBS) [5] in conjunction with a Naïve-Bayesian Classifier (NBC) to create the feature selection wrapper method. The only portion of CEFS not implemented was the ensemble integration technique, for two reasons. First, by not implementing an integration technique and simply measuring whether either of the classifiers correctly classified the instance, we are able to attain an upper bound on performance. Second, further research and experimentation are needed in order to choose the most appropriate integration methods to be used in this system.

In terms of metrics, the overall accuracy of CEFS and of a baseline model, also using SBS and NBC, were measured. In addition to overall accuracy, several other metrics were recorded to support the hypothesis proposed in this thesis. These metrics will be further detailed in the subsequent section.

4.2.4 Experiment Results

Several comparative metrics were taken for this experiment. The first was to compare the optimal feature subsets acquired from the original baseline algorithm to CEFS. Table 2 gives the optimal feature subsets acquired by each algorithm. Although the dataset is considered small on a dimensional scale, the results can serve to illustrate several points. To begin, the only feature useful to both class-specific subsets was feature 8 – breast quadrant. Interestingly enough though, this feature was not deemed useful in terms of the optimal subset for the entire dataset. In contrast, features 5 and 6 were the only ones selected by the baseline algorithm, yet neither of them is deemed useful for instances classified as recurrence. Feature 8 was the only feature deemed useful for recurrence instances, yet this feature was not deemed at all useful under the baseline algorithm. The results from this dataset alone support the idea of feature dataset exclusivity, which shows that features useful for one classification may be different from other classifications, and may be lost at prediction time if using the baseline type of feature selection.

Table 2: Optimal Feature Subsets under Breast Cancer Dataset

Feature Subset	1	2	3	4	5	6	7	8	9
Entire Dataset (baseline)					X	X			
No-recurrence specific (CEFS)	X		X		X	X	X	X	X
Recurrence specific (CEFS)								X	

(1-age, 2-menopause, 3-tumor size, 4-inv nodes, 5-node caps, 6-deg malig, 7-breast side, 8-breast quad, 9-irradiat)

The next metric, depicted in Figure 8, measured the percentage of times each classifier agreed or disagreed on their prediction. This is important, because the

ensemble system accurately predicted about 80% of the instances when the classifiers agreed, which was 73% of the time.

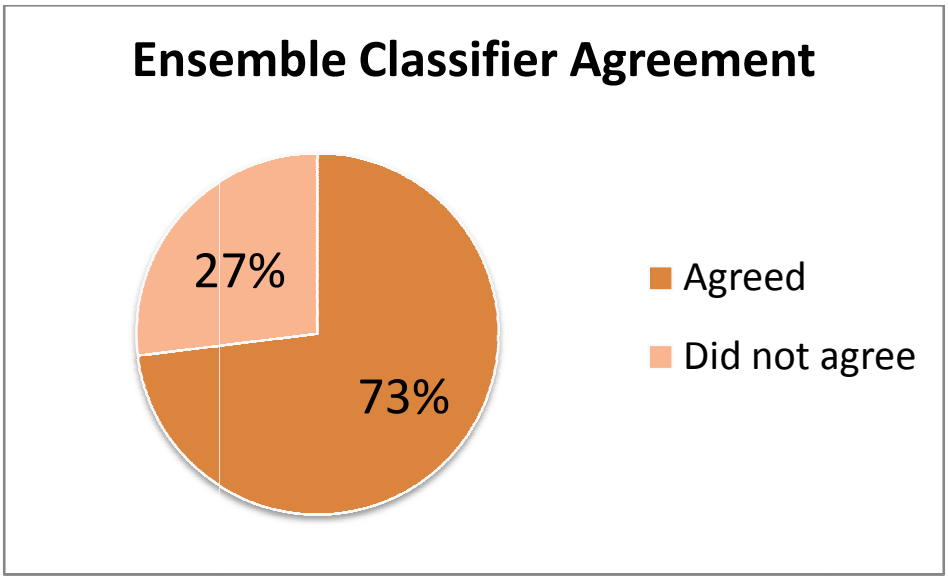

Figure 8: Ensemble Classifier Agreement

According to Figure 8, the other 27% of the time, the classifiers disagreed. The prediction accuracy when the classifiers disagree is dependent on which type of ensemble integration method is used. Since no ensemble integration method was implemented, this figure is not available. However, if the integration technique selected were to allow the ensemble machine to correctly predict 100% of the instances when the classifiers disagreed, then the algorithm would reach an overall accuracy of 85.3% (giving us an *upper bound* on accuracy). If at the worst, the integration technique predicted as good as when the classifiers agree, that would still yield an overall accuracy of roughly 80%. Figure 9 displays the prediction accuracy of an NBC without any feature selection, the baseline algorithm aforementioned, the case where disagreeing classifiers predicted as

well as agreeing classifiers and the upper bound on prediction accuracy (when disagreeing classifiers predict correctly 100% of the time).

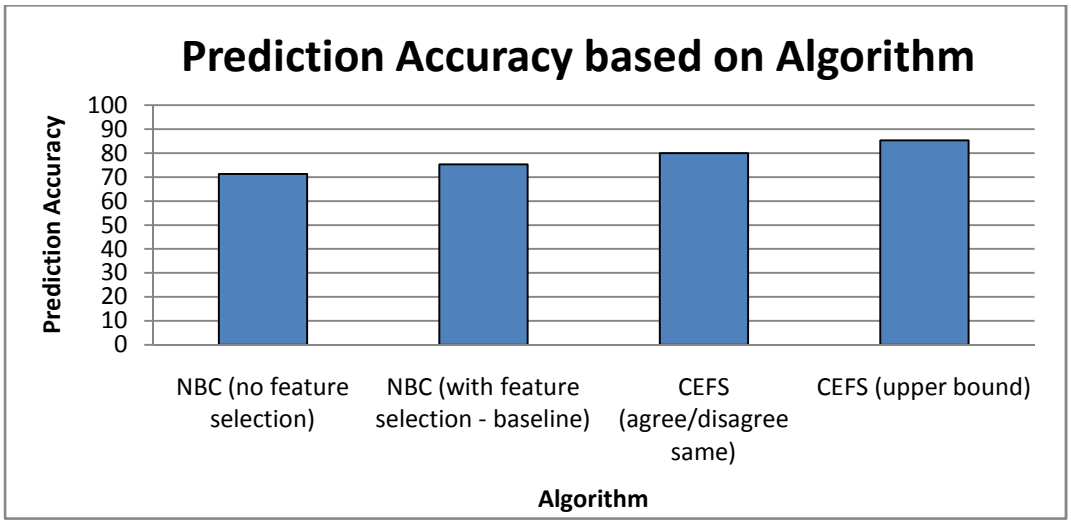

Figure 9: Prediction Accuracy based on Algorithm

As shown by Figure 9, choosing an effective integration technique for the ensemble system can provide significantly better accuracy than existent methods. Moreover, this figure shows that the opportunity for improvement is quite possible for the CEFS proposed approach.

CHAPTER 5

CONCLUSION

Given the motivation and objectives presented in Chapter 1, a unique ensemble feature selection algorithm is proposed and presented. This algorithm utilizes a wrapper method to build an ensemble of models, each with feature subsets optimized for performance with respect to a separate classification and under a specific base classifier. The proposed method seeks to outperform (in terms of prediction accuracy) the state of the art by providing an approach which will bring sought after diversity and disagreement to the ensemble model while supplying the same model with feature subsets containing highly useful features to the base classifiers themselves. Preliminary Results show promise in terms of possible increases in prediction accuracy, with potential improvements of more than 10% over exiting methods.

In terms of future work, this research can evolve in a variety of different ways. First, a thorough comparative study on how performance is affected by different wrapper methods, higher and lower number of classifications, and different ensemble integration techniques, should be conducted. Second, it would be worthwhile to investigate an ensemble of models with feature subsets derived from both filter and wrapper methods. Third, a comparative study of different evolutionary computation algorithms utilized as wrapper methods within ensemble feature selection models might provide valuable

information as to better performing wrapper methods. Finally, given that ensemble feature selection is still a relatively new area to feature selection, and class-specific selection even newer, fresh and innovative combinations may provide even better performing algorithms.

BIBLIOGRAPHY

[1] G. Piatetsky-Shapiro, "Knowledge Discovery in Databases: 10 years later," *SIGKDD Explorations*, vol. 1, no. 2, pp. 59-61, Jan. 2000.

[2] H. Liu and L. Yu, "Toward Integrating Feature Selection Algorithms for Classification and Clustering," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 4, pp. 491-502, Apr. 2005.

[3] J. Alpert and N. Hajaj. (2008, Jul.) The Official Google Blog: We knew the web was big.... [Online]. http://googleblog.blogspot.com/2008/07/we-knew-web-was-big.html

[4] C. Soares, L. Montgomery, K. Rouse, and J. E. Gilbert, "Automating Classification using General Regression Neural Networks," in *Proceedings of the 2008 Seventh International Conference on Machine Learning and Applications*, San Diego, CA, 2008, pp. 508-513.

[5] A. Jain and D. Zongker, "Feature Selection: Evaluation, Application, and Small Sample Performance," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 2, pp. 153-158, 1997.

[6] E. Bauer and R. Kohavi, "An Empirical Comparison of Voting Classification Algorithms: Bagging, Boosting, and Variants," *Machine Learning*, vol. 36, no. 1, 2, pp. 105-139, 1999.

[7] T. G. Dietterich, "Ensemble Learning Methods," in *Handbook of Brain Theory and Neural Networks*, M. A. Arbib, Ed. MIT Press, 2001.

[8] L. Hansen and P. Salamon, "Neural Network Ensembles," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, pp. 993-1001, 1990.

[9] T. R. Golub, et al., "Molecular Classification of Cancer: Class Discovery and Class Prediction by Gene Expression Monitoring," *Science*, vol. 286, no. 5439, pp. 531-537, 1999.

[10] U. Alon, et al., "Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays," *PNAS*, vol. 96, no. 12, pp. 6745-6750, 1999.

[11] S. J. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*. Upper Saddle River, NJ: Pearson Education, 2003.

[12] P. Langley and H. A. Simon, "Applications of machine learning and rule induction," *Communications of ACM*, vol. 38, no. 11, pp. 55-64, 1995.

[13] D. H. Wolpert and W. G. Macready, "No free lunch theorems for search," Santa Fe Institute, Santa Fe, CA, Technical Report SFI-TR-95-02-010, 1995.

[14] R. Bellman, *Adaptive Control Processes: A Guided Tour*. Princeton University Press, 1961.

[15] P.-N. Tan, M. Steinbach, and V. Kumar, *Introduction to Data Mining*. Addison-Wesley, 2006.

[16] J. Sinkkonen and W. S. Sarle. (2002, Oct.) Neural Network FAQ, part 2 of 7: Learning. [Online]. ftp://ftp.sas.com/pub/neural/FAQ2.html

[17] B. W. Silverman, *Density Estimation for Statistics and Data Analysis*, 1st ed. Chapman and Hall/CRC Press, 1986.

[18] I. Guyon and A. Elisseeff, "An Introduction to Variable and Feature Selection," *Journal of Machine Learning Research*, vol. 3, pp. 1157-1182, 2003.

[19] D. L. Donoho, "High-dimensional data analysis: The curses and blessings of dimensionality," in *Aide-Memoire of a Lecture at AMS Conference on Math Challenges of the 21st Century*, 2000.

[20] A. Y. Ng, "Feature selection, L1 vs. L2 regularization, and rotational invariance," in *International Conference on Machine Learning*, 2004.

[21] J. R. Quinlan, "Induction of decision trees," *Machine Learning*, vol. 1, pp. 81-106, 1986.

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

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

[24] B. V. Dasarathy, *Nearest Neighbor (NN) Norms: NN Pattern Classification Techniques*. Los Alamitos, CA: IEEE Computer Society Press, 1990.

[25] D. W. Aha, D. Kibler, and M. K. Albert, "Instance Based learning algorithms," *Machine Learning*, vol. 6, no. 1, pp. 37-66, 1991.

[26] R. Kohavi and G. H. John, "Wrappers for feature subset selection," *Artificial Intelligence: Special Issue on relevance*, vol. 97, no. 1 - 2, pp. 273-324, 1997.

[27] P. Langley, W. Iba, and K. Thompson, "An analysis of Bayesian classifiers," in *Proceedings of the Tenth National Conference on Artificial Intelligence*, 1992, pp. 223-228.

[28] R. Duda and P. Hart, *Pattern Classification and Scene Analysis*. Wiley, 1973.

[29] I. J. Good, *The Estimation of Probabilities: An Essay on Modern Bayesian Methods*. MIT Press, 1965.

[30] M. Dash and H. Liu, "Feature selection for classification," *Intelligent Data Analysis*, vol. 1, no. 1 - 4, pp. 131-156, 1997.

[31] A. L. Blum and R. L. Rivest, "Training a 3-Node Neural Networks is NP-Complete," *Neural Networks*, vol. 5, pp. 117-127, 1992.

[32] E. Amaldi and V. Kann, "On the approximation of minimizing non zero variables or unsatisfied relations in linear systems," *Theoretical Computer Science*, vol. 209, pp. 237-260, 1998.

[33] A. Kalousis, J. Prados, and M. Hilario, "Stability of Feature Selection Algorithms," in *Proceedings of the Fifth IEEE International Conference on Data Mining*, Houston, TX, 2005.

[34] A. Kalousis, J. Prados, and M. Hilario, "Stability of feature selection algorithms: a study on high-dimensional spaces," *Knowledge and Information Systems*, vol. 12, no. 1, pp. 95-116, 2006.

[35] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik, "Gene selection for cancer classification using support vector machines," *Machine Learning*, vol. 46, no. 1 - 3, pp. 389-422, 2002.

[36] R. Nilsson, J. M. Pena, J. Bjorkegren, and J. Tegner, "Consistent Feature Selection for Pattern Recognition," *Journal of Machine Learning Research*, vol. 8, pp. 589-612, 2007.

[37] E. Xing, M. Jordan, and R. Karp, "Feature Selection for High-Dimensional Genomic Microarray Data," in *Proceedings of the 15th International Conference on Machine Learning*, 2001, pp. 601-608.

[38] F. K. Ahmad, N. M. Norwawi, S. Deris, and N. H. Othman, "A review of feature selection techniques via gene expression profiles," *International Symposium on Information Technology*, vol. 2, pp. 1-7, Aug. 2008.

[39] P. Langley, "Selection of relevant features in Machine Learning," in *AAAI Fall Symposium on Relevance*, 1994, pp. 140-144.

[40] R. Caruana and D. Freitag, "Greedy attribute selection," in *Machine Learning: Proceedings of the Eleventh International Conference*, 1994, pp. 28-36.

[41] D. Koller and M. Sahami, "Toward optimal feature selection," in *Proceedings of the 13th International Conference on Machine Learning*, San Francisco, CA, 1996, pp. 284-292.

[42] G. Forman, "An Extensive Empirical Study of Feature Selection Metrics for Text Classification," *Journal of Machine Learning Research*, vol. 3, pp. 1289-1305, 2003.

[43] H. Lu, et al., "The Effects of Domain Knowledge Relations on Domain Text Classification," in *Proceedings of the 27th Chinese Control Conference*, Kumming, Yunnan, China, 2008, pp. 460-463.

[44] Q. Chen, "Feature Selection for the Topic-Based Mixture Model in Factored Classification," in *Proceedings of the 2006 International Conference on Computational Intelligence and Security*, 2006, pp. 39-44.

[45] E. Gabrilovich and S. Markovitch, "Text categorization with many redundant features: using aggressive feature selection to make SVMs competitive with C4.5," in *Proceedings of the 21st International Conference on Machine Learning*, 2004, pp. 41-48.

[46] A. Dasgupta, P. Drineas, B. Harb, V. Josifovski, and M. W. Mahoney, "Feature Selection Methods for Text Classification," in *Proceedings of the 13th SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Jose, CA, 2007, pp. 230-239.

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

[48] M. Ben-Bassat, *Use of distance measures, information measures and error bounds in feature evaluation*, 2nd ed., P. R. Krishnaiah and L. N. Kanal, Eds. North-Holland Publishing Company, 1982.

[49] A. Miller, *Subset Selection in Regression*, 2nd ed. Chapman & Hall/CRC, 2002.

[50] N. R. Draper and H. Smith, *Applied Regression Analysis*, 2nd ed. John Wiley and Sons, 1981.

[51] J. Doak, "An evaluation of feature selection methods and their application to computer security," University of California at Davis, Technical Report CSE-92-18, 1992.

[52] P. Pudil, J. Novovicova, and J. Kittler, "Floating search methods in feature selection," *Pattern Recognition Letters*, vol. 15, pp. 1119-1125, 1994.

[53] M. Kudo and J. Sklansky, "Comparison of algorithms that select features for pattern classifiers," *Patter Recognition*, vol. 33, pp. 25-41, 2000.

[54] Y. Kim, W. N. Street, and F. Menczer, "Feature selection in unsupervised learning via evolutionary search," in *Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2000, pp. 365-369.

[55] Y. Kim, W. N. Street, and F. Menczer, "An evolutionary multi-objective local selection algorithm for customer targeting," in *Proceedings of Congress on Evolutionary Computation*, 2001, pp. 759-766.

[56] R. Kohavi and D. Sommerfield, "Feature subset selection using the wrapper model: Overfitting and dynamic search space topology," in *The First International Conference on Knowledge Discovery and Data Mining*, 1995, pp. 192-197.

[57] D. W. Aha and R. L. Bankert, "A comparative evaluation of sequential feature selection algorithms," in *Proceedings of the Fifth International Workshop on Artificial Intelligence and Statistics*, Ft. Lauderdale, FL, 1995, pp. 1-7.

[58] H. Liu and H. Motola, *Feature selection for Knowledge Discovery and Data Mining*. Springer, 1998.

[59] A. Arauzo-Azofra and J. M. Benitez, "Empirical Study of Feature Selection Methods in Classification," in *Proceedings of the Eighth International Conference on Hybrid Intelligent Systems*, 2008, pp. 584-589.

[60] S. B. Cho and H. H. Won, "Machine Learning in DNA Microarray Analysis for Cancer Classification," in *Proceedings of the First Asia-Pacific Bioinformatics Conference*, 2003.

[61] S. Das, "Filters, Wrappers and a Boosting-Based Hybrid for Feature Selection," in *Proceedings of the 18th International Conference on Machine Learning*, 2001, pp. 74-81.

[62] J. Kittler, *Feature Selection and Extraction*. Academic Press, 1986.

[63] L. Rendell and R. Seshu, "Learning hard concepts through constructive induction: framework and rationale," *Computational Intelligence*, vol. 6, no. 4, pp. 247-270, Nov. 1990.

[64] B. Scholkopf and A. J. Smola, *Learning with Kernels*. Cambridge, MA: MIT Press, 2002.

[65] I. T. Jolliffe, *Principal Component Analysis*. New York, NY: Springer-Verlag, 1986.

[66] K. I. Diamantaras and S. Y. Kung, *Principal Component Neural Networks*. New York: Wiley, 1996.

[67] L. J. Cao and W. K. Chong, "Feature Extraction in Support Vector Machine: A comparison of PCA, KPCA, and ICA," in *Proceedings of the 9th International Conference on Neural Information Processing*, 2002, pp. 1001-1005.

[68] A. L. Blum and P. Langley, "Selection of Relevant Features and Examples in Machine Learning," *Artificial Intelligence*, vol. 97, pp. 245-271, 1997.

[69] D. A. Bell and H. Wang, "A formalism for relevance and its application in feature subset selection," *Machine Learning*, vol. 41, pp. 175-195, 2000.

[70] R. A. Caruana and D. Freitag, "How useful is relevance?," in *Working Notes of the AAAI Fall Symposium on Relevance*, New Orleans, LA, 1994, pp. 25-29.

[71] M. Steinbach, L. Ertoz, and V. Kumar, "The challenges of clustering high-dimensional data," *New Vistas in Statistical Physics: Applications in Econophysics, Bioinformatics, and Pattern Recognition*, 2003.

[72] M. Dash and H. Liu, "Feature Selection for Clustering," in *Proceedings of the Fourth Pacific Asia Conference on Knowledge Discovery and Data Mining*, 2000, pp. 110-121.

[73] J. G. Dy and C. E. Brodley, "Feature Subset Selection and Order Identification for Unsupervised Learning," in *Proceedings of the 17th International Conference on Machine Learning*, 2000, pp. 247-254.

[74] L. Talavera, "Feature Selection as a Preprocessing Step for Hierarchical Clustering," in *Proceedings of the International Conference on Machine Learning*, 1999, pp. 389-397.

[75] A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data*. Prentice Hall, 1988.

[76] L. Kaufman and P. J. Rousseeuw, *Finding Groups in Data: an Introduction to Cluster Analysis*. John Wiley and Sons, 1990.

[77] G. H. John, R. Kohavi, and K. Pfleger, "Irrelevant features and the subset selection problem," in *Proceedings of the Eleventh International Conference on Machine Learning*, New Brunswick, NJ, 1994, pp. 121-129.

[78] D. D. Lewis, "Representation and learning in information retrival.," Department of Computer Science, University of Massachusetts, Amherst, Doctoral dissertation and Technical Report UM-CS-1991-093, 1992.

[79] D. D. Lewis, "Feature selection and feature extraction for text categorization," in *Proceedings of Speech and Natural Language Workshop*, San Francisco, CA, 1992, pp. 212-217.

[80] H. Almuallim and T. G. Dietterich, "Learning with many irrelevant features," in *Proceedings of the Ninth National Conference on Artificial Intelligence*, San Jose, CA, 1991, pp. 547-552.

[81] K. Kira and L. A. Rendell, "The feature selection problem: Traditional methods and a new algorithm," in *Tenth National Conference on Artificial Intelligence*, 1992, pp. 129-134.

[82] I. Kononenko, "Estimating attributes: Analysis and extensions of RELIEF," *Machine Learning: ECML-94*, vol. 784/1994, pp. 171-182, 1994.

[83] G. Liu, L. Dong, S. Yuan, Y. Liu, and Y. Li, "New Feature Selection Algorithm based on Potential Difference," in *Proceedings of the 2007 IEEE International Conference on Mechatronics and Automation*, Harbin, China, 2007, pp. 566-570.

[84] H.-H. Hsu, C.-W. Hsieh, and M.-D. Lu, "A Hybrid Feature Selection Mechanism," in *Eighth International Conference on Intelligent Systems Design and Applications*, 2008, pp. 271-276.

[85] G. Pagallo and D. Haussler, "Boolean feature discovery in empirical learning," *Machine Learning*, vol. 5, pp. 71-99, 1990.

[86] D. Lei, Y. Xiaochun, and X. Jun, "Optimizing Traffic Classification Using Hybrid Feature Selection," in *Proceedings of the Ninth International Conference on Web-Age Information Management*, 2008, pp. 520-525.

[87] M. Baglioni, B. Furletti, and F. Turini, "DrC4.5: Improving C4.5 by means of prior knowledge," in *Proceedings of the 2005 ACM Symposium on Applied Computing*, Santa Fe, CA, 2005, pp. 474-481.

[88] A. Tsymbal, S. Puuronen, and D. Patterson, "Ensemble feature selection with the simple Bayesian classification," *Information Fusion*, vol. 4, pp. 87-100, 2003.

[89] R. Kohavi, "A study of cross-validation and bootstrap for accuracy estimation and model selection," in *Proceedings of the International Joint Conference on Artificial Intelligence*, 1995.

[90] C. Merz, "Dynamical selection of learning algorithms," in *Learnin from Data, Artificial Intelligence and Statistics*, D. Fisher and H. .-J. Lenz, Eds. NY: Springer-Verlag, 1996.

[91] M. Koppel and S. P. Engelson, "Integrating multiple classifiers by finding their areas of expertise," in *AAAI-96 Workshop On Integrating Multiple Learning Models*, 1996, pp. 53-58.

[92] A. Krogh and J. Vedelsby, "Neural Network ensembles, cross validation, and active learning," in *Advances in Neural Information Processing Systems*, D. Touretzky and T. Leen, Eds. Cambridge, MA: MIT Press, 1995, vol. 7, pp. 231-238.

[93] K. Tumer and J. Ghosh, "Error correlation and error reduction in ensemble classifiers," *Connection Science, Special Issue on Combining Artificial Neural Networks: Ensemble Approaches*, vol. 9, no. 3, 4, pp. 385-404, 1996.

[94] C. Brodley and T. Lane, "Creating and exploiting coverage and diversity," in *AAAI-96 Workshop on Integrating Multiple Learned Models*, 1996, pp. 8-14.

[95] L. Asker and R. Maclin, "Ensembles as a sequence of classifiers," in *Proceedings of the 15th International Joint Conference on Artificial Intelligence*, San Francisco, CA, 1997, pp. 860-865.

[96] D. W. Opitz, "Feature Selection for Ensembles," in *Proceedings of 16th National Conference on Artificial Intelligence*, 1999, pp. 379-384.

[97] T. K. Ho, "The random subspace method for constructing decision forests," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8, pp. 832-844, 1998.

[98] C. Guerra-Salcedo and D. Whitley, "Genetic Approach for Feature Selection for Ensemble Creation," in *Proceedings of the Genetic and Evolutionary Computation Conference*, Orlando, FL, 1999, pp. 236-243.

[99] C. Guerra-Salcedo and D. Whitley, "Feature Selection Mechanisms for Ensemble Creation: A Genetic Search Perspective,," Data Mining with Evolutionary Algorithms: Research Directions. Papers from the AAAI Workshop., 1999.

[100] A. Tsymbal, P. Cunningham, M. Pechenizkiy, and S. Puuronen, "Search Strategies for Ensemble Feature Selection in Medical Diagnostics," in *Proceedings of the 16th IEEE Symposium on Computer-Based Medical Systems*, 2003, pp. 124-129.

[101] L. E. A. Santana, D. F. d. Oliveira, A. M. P. Canuto, and M. C. P. d. Souto, "A Comparative Analysis of Feature Selection Methods for Ensembles with Different Combination Methods," in .

[102] K. M. O. Vale, F. G. Dias, A. M. P. Canuto, and M. C. P. Souto, "A Class-Based Feature Selection Method for Ensemble Systems," in *Proceedings of the Eighth International Conference on Hybrid Intelligent Systems*, 2008, pp. 596-601.

[103] P. Cunningham and J. Carney, "Diversity versus quality in classification ensembles based on feature selection," in *Proceedings of the 11th European Conference on Machine Learning*, 2000, pp. 109-116.

[104] T. G. Dietterich, "Machine Learning Research: four current directions," *AI Magazine*, vol. 18, no. 4, pp. 97-136, 1997.