Classification in the Presence of Ordered Classes and Weighted Evaluative Attributes

Forrest Justin Osterman
Louisiana State University and Agricultural and Mechanical College, forrestosterman@hotmail.com

CLASSIFICATION IN THE PRESENCE OF ORDERED CLASSES AND WEIGHTED EVALUATIVE ATTRIBUTES

A Dissertation

Submitted to the Graduate Faculty of the
Louisiana State University and
Agricultural and Mechanical College
in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

in

The Department of Computer Science

by
Forrest J. Osterman
B.S., Louisiana State University, 2000
May 2015

This work is dedicated to Walter Otto Osterman, Jr., who inspired me to make this journey.

ACKNOWLEDGEMENTS

I would like to express my gratitude to my major professor Dr. Evangelos Triantaphyllou for his guidance, support, and encouragement during the course of my Ph.D. program. Dr. Triantaphyllou has a true passion for research, and he has shown me, above all, to define the problem correctly, and enjoy the solution process. I would also like to thank Dr. Jian Zhang, Dr. Rahul Shah, and Dr. Helmut Schneider for their insights and expertise, and for serving on my graduate committee. I wish to recognize the many professors in the Computer Science department who have shared their knowledge and experience with me. I would also like to Ms. Maggie Edwards for her invaluable help and encouragement. Finally, I would like to thank my wonderful fiancé and my family, for their support.

TABLE OF CONTENTS

ABSTRACT

We are interested in an important family of problems in the interface of the Multi-Attribute Decision-Making and Data Mining fields. This is a special case of the general classification problem, in which records describing entities of interest have been expressed in terms of a number of evaluative attributes. These attributes are associated with weights of importance, and both the data and the classes are ordinal. Our goal is to use historical records and the corresponding decisions to best estimate the class values of new data points in a way consistent with prior classification decisions, without knowledge of the weights of the evaluative attributes.

We study three variants of this problem. The first is when all decisions are consistent with a single set of attribute weights (called the separable case.) The second is when all decisions are consistent, but involve two sets of attribute weights corresponding to two decision makers, who determine the classification of the data together (called the two-plane separable case.) The third is when there is some inconsistency in the set of weights that must be accounted for (called the non-separable case.) Furthermore, we examine 2-class problems and also multiple class problems. We propose the Ordinal Boundary method, which has a significant advantage over traditional approaches in multi-class problems. Linear programming (optimization) based approaches provide a promising avenue for dealing with these problems effectively. We present computational results that support this argument.

1. INTRODUCTION

We examine a set of problems which lie in the intersection of two exciting and continually developing areas of study: Multi-Attribute Decision-Making (MADM), also known as Multi-Criteria Decision-Making (MCDM), and Machine Learning/Data Mining. We would like to apply the philosophy of Machine Learning/Data Mining approaches, which are typically concerned with finding and modeling patterns in large data sets, to the MADM domain, which is concerned with evaluating and comparing potential actions based on available attribute information. One needs to consider these two fields in order to develop effective solution approaches to solve a type of problems that lie in the interface of these two areas.

Our goal is to make new classification decisions for an MADM problem which are consistent with historical decisions. On the surface, this seems ideally suited for traditional Machine Learning/Data Mining approaches, as inferring classification models is a primary topic of Machine Learning/Data Mining. However, traditional Data Mining models suffer from some disadvantages in this application. They are designed to be flexible and do not take advantage of all the knowledge we have about this application. The generalized nature that gives these models their flexibility can also be a performance limitation. As the number of classification categories and attributes becomes larger, the performance of even the best suited traditional Machine Learning/Data Mining methods degrades significantly.

In a typical MADM problem, the decision maker is presented with a set of *alternatives*, each of which represents a solution or potential course of action that can be taken. These alternatives are expressed in terms of a set of *evaluative attributes*, each of which represents an aspect of the problem to be considered. These attributes are useful for comparing alternatives with each other. The decision maker also has (or assigns) a corresponding set of *weights*, which express the relative importance of the attributes. While each alternative can be expressed as a vector of values in terms of these attributes, the weights of the attributes are applicable universally to all alternatives.

A simple example of an MADM problem in this context is the task of reviewing loan applications. The decision maker, in this case a loan officer, may use the credit score, income, employment status, and debt-to-credit ratio of each applicant as the evaluative attributes in order to make a decision about a loan extension. Each applicant can be considered as an alternative solution to the loan problem. As classes in this case one may consider groups defined as follows: "Applicant is highly trustworthy for credit," "Applicant is somewhat trustworthy," "Applicant may default on the loan," and "Applicant would most likely default on the loan."

Table 1-1 shows this type of information in the form of a *decision matrix*. There are m solutions (alternatives) expressed in rows of n-dimensional vectors. These n dimensions correspond one-to-one to the n evaluative attributes. Each of the x_{ij} entries in Table 1-1 corresponds to the performance value of alternative A_i in terms of attribute a_j. Each w_j is the relative weight of importance of evaluative attribute a_j.

Table 1-1. Structure of a typical decision matrix.

Weight	***(w_1)***	***(w_2)***	***…***	***(w_n)***
		Attribute		
Alternative	a_1	a_2	…	a_n
A_1	x_{11}	x_{12}	$\cdots$	x_{1n}
A_2	x_{21}	x_{22}	$\cdots$	x_{2n}
$\cdots$	…	…	$\cdots$	…
A_m	x_{m1}	x_{m2}	$\cdots$	x_{mn}

There are several well-studied MADM problem formulations which are referred to as "problematics" (Roy, 1996). These problematics represent different ways to view MADM problems and the different types of results that a corresponding approach will provide. Three of the reference problematics are the *choice problematic*, the *sorting problematic,* and the *ranking problematic*.

a) The *choice problematic* focuses on selecting the smallest possible subset of "the best" alternatives from the given set of alternatives. An example: deciding on a job applicant to hire from a pool of interviewees.

b) The goal of the *sorting problematic* is to assign each alternative to a predefined ordinal category, or *class*, where it is grouped together with similar alternatives. An example: Rating credit applications into one of several ordered risk categories, such as "low risk," "medium risk," and "high risk."

c) The goal of the *ranking problematic* is to provide a complete or partial ordering between alternatives. An example: Setting priorities for funding in multiple competing projects.

We focus on the sorting problematic, which is commonly referred to in the Machine Learning/Data Mining field as a *classification* process (Tan, et al., 2006). Sorting problems in the MADM field are often ordered, because they depend on ordinal criteria (attributes) (Zopounidis & Doumpos, 2002), while Machine Learning/Data Mining classification problems may use either nominal or ordinal classes. We are interested in addressing the following situation: We have a decision matrix of some alternatives described in terms of some evaluative attributes. We also have the corresponding ordered classifications of these alternatives. However, we do not have the weights of the attributes used to derive these ordered classifications. Next, a new alternative is given. We would like to make a decision somehow consistent with the original decision maker's decisions.

In some situations, the need to reverse engineer these decisions will arise. A decision maker may not be available to query about their processes. They may have retired, or may be working for a business competitor. It is useful to have tools that can determine how their decisions were made from their input data and their past decisions alone. The results can then be used to make future determinations, or to review and modify the older determinations.

We discuss how to build a fast and accurate optimization-based approach to determine the ranges of possible weights for these problems. Our approach takes advantage of the ordinal nature of the classification categories (or classes), and the fact that a single set of weights is applied consistently to all alternatives. This is the *separable* case. We also discuss the case in which there is no possible single set of weights that can be used with a linear model to explain the classifications of all previous alternatives simultaneously.

This is the *non-separable* case. We study these two types of problems with two classes and also with more than two classes. We compare our method with traditional Data Mining and MADM methods, and perform experiments on some generated and real world data. This dissertation is organized as follows. Chapter 2 provides a description of the problem. Chapter 3 provides a review of the standard MADM models, as well as other classification and ranking literature. Chapter 4 provides a description of the approaches with simple examples. Chapter 5 discusses the data sets, testing methods, and the results of computational experiments and analyzes these results. Chapter 6 provides concluding remarks and goals for further research.

2. PROBLEM DESCRIPTION

In the problem addressed in this work, one has a history of past classifications, and the attributes for each alternative, but not the attribute weights used or the definitions of the class boundaries. The goal is to estimate these attribute weights and class boundaries in a manner as consistent as possible with the historical data in order to make future classifications.

Attributes for MADM problems may be presented in a variety of scales and may be a mix of cost and benefit measures. They need to be compared in order to make a classification. Each attribute can be normalized so that they appear on the same scale, as long as it is a ratio type attribute, i.e., both differences and ratios between attributes are meaningful (Tan, et al., 2006). We are interested in problems which mainly have ratio type attributes.

However, this will not express how relevant each attribute is to the classification. A powerful method for evaluating attributes of different relevance is to use weights. Assigning a higher weight to an attribute shows preference for that attribute over those with lower weights. A common approach is the *Weighted Sum Model* (Fishburn, 1967). The best alternative is decided to be the one that satisfies the additive expression:

$$\text{Maximize } P(A_i) = \sum_{j=1}^{n} x_{ij}\, w_j, \qquad \text{for } i = 1, 2, \ldots, m. \qquad (2\text{-}1)$$

This model assigns an overall preference value ($P(A_i)$) to each alternative ($A_i,\ i = 1,2,\ldots,m$), which is the sum of the products of the relative preferences of the alternatives

(x_{ij}) times the weights (w_j) of the evaluative attributes (a_j). This expression assumes that all of the attributes are of the benefit type. We leave the investigation of a mix of benefit and cost attributes for future work.

The following simple example demonstrates the process that the decision maker follows to arrive at the original classifications. Table 2-1 shows a small subset of data from a 2-attribute, 2-class problem, where the attributes fall within the range of 0-1.

Table 2-1. Example problem data.

Alternative	a_1	a_2
A_1	0.6	0.2
A_2	0.8	0.3
A_3	0.4	0.5
A_4	0.5	0.4

First, the decision maker must choose weights to express the relative importance of each attribute. In this example, the decision maker has chosen weights of 0.4 and 0.6 for attributes a_1 and a_2, respectively. Once the weights are expressed numerically, the decision maker can use a model such as the WSM, given as expression (2-1), to assign a preference value to each alternative, as shown in Table 2-2.

Table 2-2. Preference values of example problem.

Weight	(0.4)	(0.6)	
Alternative	a_1	a_2	$P(A_i)$
A_1	0.6	0.2	0.36
A_2	0.8	0.3	0.50
A_3	0.4	0.5	0.46
A_4	0.5	0.4	0.44

Calculating $P(A_i)$ for Alternative $A_1 \dots A_4$ according to (2-1):

$$P(A_1) = \sum x_{1j}w_j = 0.6w_1 + 0.2w_2 = 0.6(0.4) + 0.2(0.6) = 0.36$$

$$P(A_2) = \sum x_{2j}w_j = 0.8w_1 + 0.3w_2 = 0.8(0.4) + 0.3(0.6) = 0.50$$

$$P(A_3) = \sum x_{3j}w_j = 0.4w_1 + 0.5w_2 = 0.4(0.4) + 0.5(0.6) = 0.46$$

$$P(A_4) = \sum x_{4j}w_j = 0.5w_1 + 0.4w_2 = 0.5(0.4) + 0.4(0.6) = 0.44$$

These preference values can be used to address all three of the previously mentioned problematics, however, we are concerned with sorting (classification.) To classify these alternatives, in addition to the preference values, the decision maker must choose a threshold preference value, $P(\theta)$.

For a 2-class problem, the classification function is:

If $P(A_i) < P(\theta)$ $\Rightarrow$ Class C_1, otherwise

$\Rightarrow$ Class C_2.

For this example, the decision maker has chosen a threshold value $P(\theta)$ of 0.45. The alternatives are thus classified as shown in Table 2-3.

Table 2-3. Classifications of example data.

Weight	(0.4)	(0.6)		
Alternative	a_1	a_2	$P(A_i)$	Class
A_1	0.6	0.2	0.36	C_1
A_2	0.8	0.3	0.50	C_2
A_3	0.4	0.5	0.46	C_2
A_4	0.5	0.4	0.44	C_1

Finally, we consider the situation where the weights and threshold preference values used by the original decision maker are unavailable. The proposed methods must work only from the input attributes (shown again in Table 2-4) and the corresponding decisions for that data in order to find a consistent classification for the new alternative A_5.

Table 2-4. Example problem data.

Alternative	a_1	a_2	Class
A_1	0.6	0.2	C_1
A_2	0.8	0.3	C_2
A_3	0.4	0.5	C_2
A_4	0.5	0.4	C_1
A_5	0.5	0.7	?

We would like to determine whether the alternatives A_1 to A_4 have been processed under the same set of attribute weights in order to derive their class values, and what class value for the new alternative, A_5, is most consistent with the available training data (i.e., the top part of Table 2-4.) We also consider the cases where the alternatives cannot be linearly separated using a single weight vector. This includes the case where there is some degree of variation in the weights the alternatives were processed under, and the case where there are multiple decision makers making the classifications. We discuss the 2-class and multi-class cases for these scenarios. After the appropriate literature is reviewed, some methods for dealing with these problems are discussed in Chapter 4.

3. LITERATURE REVIEW

MADM methods for the sorting problematic have many practical real world applications. Some examples include credit risk assessment (Zopounidis & Doumpos, 2002), business failure prediction (Dimitras, et al., 1999), bond credit ratings (Bugera, 2008), evaluation of insurance companies (Pardalos, et al., 1997), supplier selection and evaluation (Araz & Ozkarahan, 2007), and customer satisfaction (Dutka, 1995).

Because of this multitude of useful applications, models for classification and sorting have been the subject of recent attention in both the MADM and Data Mining disciplines. They share a common goal in trying to fit a model to a problem. In Data Mining, the problem is typically described by previous data and modeled "automatically" without further input. MADM approaches are developed to operate around interaction with a decision maker, and use their input to build, review, and improve the model. However, the growing complexity of some of these models, as they are continually extended, makes it difficult to elicit them directly from the decision maker and difficult for those without extensive MADM expertise to interpret. This has prompted the research for some "automatic" model inference in the MADM field as well (Mousseau, 2000).

In recent years, the earlier statistical models used for MADM classification, such as linear discriminate analysis (Fisher, 1936) and logit/probit analysis (McFadden, 1974), have been replaced with *criteria aggregation* methods (Zopounidis & Doumpos, 2002). These methods combine input attributes (criteria) according to a model that reflects the decision maker's preference. The most widely used types of criteria aggregation methods are the outranking relation and the utility function.

Outranking relation methods, such as those in the ELECTRE family of methods, use binary relations between attributes to assess an outranking degree of one alternative over another (Figueria, 2005). When employed in classification/sorting problems, these outranking relations are considered with respect to reference alternatives that represent class boundaries. Because they are complex and time-consuming to construct directly through expert interaction, there has been some work to infer ELECTRE models from assignment examples (Mousseau & Slowinski, 1998). However, ELECTRE models have some serious shortcomings with ranking inconsistencies (Wang & Triantaphyllou, 2008).

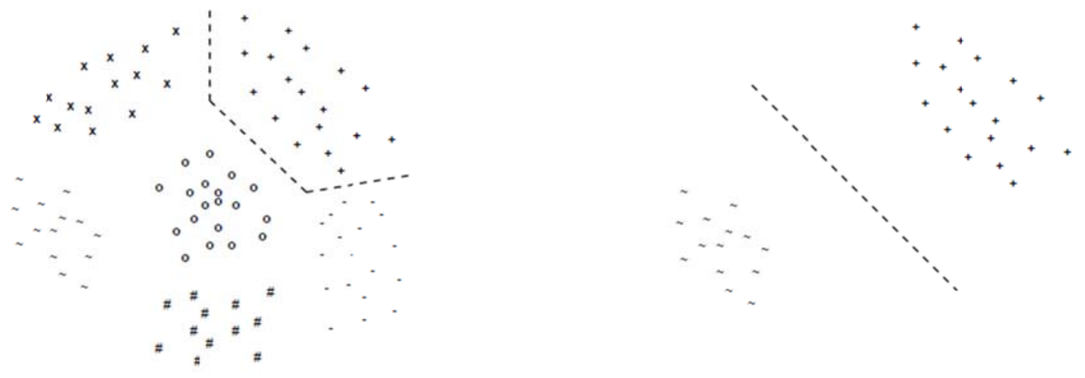
Additive utility function methods such as UTA and UTADIS (Jacquet-Lagreze, 1982), are widely used criteria aggregation methods (Jacquet-Lagreze & Siskos, 2001). These methods use a composition of utility functions, each of which expresses an overall preference value for an alternative. Methods for fitting some quadratic utility functions for classifying credit ratings of bonds are discussed in (Bugera, 2008).

In addition to MADM inspired methods, there are a large number of Machine Learning/ Data Mining methods which are applicable to this type of problem. Neural Networks (NN) have been employed with some success (Malakooti & Zhou, 1994). However, the resulting NN models are often hard to relate to real-world preferences. Support Vector Machines (SVM) have recently been used for many classification problems, and there has been some investigation into ordered extensions to the SVM model (Kim & Ahn, 2011), but without significant improvement in classification accuracy.

Rather than argue for a complex MADM model, we examine the potential of the Weighted Sum Model. The "Multi-Criteria Analysis Manual," published by The Department for Communities and Local Government in London (Dodgson, 2009), lists the following criteria (on page 20) for choosing "MCDA" models (used here synonymously with MADM/MCDM) for practitioners: *"internal consistency and logical soundness, transparency, ease of use, data requirements not inconsistent with the importance of the issue being considered, realistic time and manpower resource requirements for the analysis process, the ability to provide an audit trail, and software availability, where needed."* Based on these criteria, this manual recommends some methods based on the WSM to address several of the presented case studies of classification-oriented MADM problems. They refer to the WSM as the *"basic model of rational choice under uncertainty…the most broadly accepted normative model of rational choice."* This sentiment is reinforced in (Triantaphyllou, 2000) and (Pokorny, 2011), which both refer to the WSM as *"probably the most widely used* [decision-making] *method."*

Our method infers this MADM model using data of past decisions, and makes use of some assumptions that will be true where this model was previously employed. The assumption of interest is that the weights used in the model were applied consistently to all training examples. We can use training examples from every class to simultaneously constrain the estimation of all of the boundaries dividing each pair of classes. Even in cases where there is some inconsistency in the way the weights were applied, we can estimate a single set of weights which most consistently describes how the original classification was done. This property is not exploited by utility functions, which estimate borders independently, nor is it addressed by outranking relations or current Machine Learning/Data Mining methods.

When dealing with multi-class classification problems, a common approach of traditional Data Mining methods is to apply an iterative classification scheme, referred to as a *meta-classifier,* to convert an *n*-class problem into a number of 2-class problems. This approach would then combine the resulting classifications from these 2-class problems in order to make a classification for the *n*-class problem. The two traditional meta-classification methods are referred to as the One-vs-One (or pairwise comparison) approach, and the One-vs-All (or One-vs-The-Rest) approach (Tan, et al., 2006). The One-vs-One approach is to create $\binom{n}{2}$ classifiers by breaking the data set up into all possible pairs of classes. The results are then combined in a voting scheme, or by some method of probabilistic combination (Hastie & Tibshirani, 1998).



(a) ONE-VS-ALL
c classifiers, each separates one class from all other classes. Here: + against all other classes.

(b) ONE-VS-ONE
$c(c-1)/2$ classifiers, one for each pair of classes. Here: + against $\sim$.

Figure 3-1. Traditional meta-classification schemes.
Adapted from (Fürnkranz, 2004).

The One-vs-All approach is to train *n* classifiers, wherein class C_1 is represented by the training data for the *i*th class, and class C_2 is represented by combining the training data from all the remaining classes (Tan, et al., 2006). Many have chosen One-vs-One as the preferred scheme for better accuracy based on empirical results. However, these results have been contested with the objection that the underlying classification techniques were not properly parameterized (Rifkin and Klautau, 2004). Some specific benefits to rule-based learners when using the One-vs-One scheme are shown in (Furnkranz, 2002). Perhaps the most beneficial improvement to the One-vs-One meta-classification scheme is the method of combining output probabilities with a method titled Pairwise Coupling (Wu & Lin, 2004). When these schemes were empirically tested using SVMs as the base learner, Pairwise Coupling was the only clear winner (Duan & Keerthi, 2005).

The work in (Frank & Hall, 2001) proposes an "Ordinal Classification" method to capitalize on ordinal information with a new meta-classification scheme. This approach breaks an n-class problem into n-1 binary problems in an alteration of the One-vs-All approach. The classifiers in this approach are built by dividing the training data between classes in keeping with their order, for example, a 3-class problem (classes C_1, C_2 and C_3, where $C_1 < C_2 < C_3$) would use two binary classifiers trained on examples organized as follows: (C_1 vs C_2&C_3) and (C_1&C_2 vs C_3). This method showed some improvement in accuracy in several data sets in comparison with the One-vs-All approach, using C4.5 trees as the underlying classifier. However, it was not tested against the One-vs-One scheme, or with less flexible base classifiers.

The authors of (Frank & Kramer, 2004) introduced another meta-classification scheme which employed a tree structure, referred to as nested dichotomies. The tree nodes are randomly split, from the full data set at the root, into decreasing sized subsets of classes.

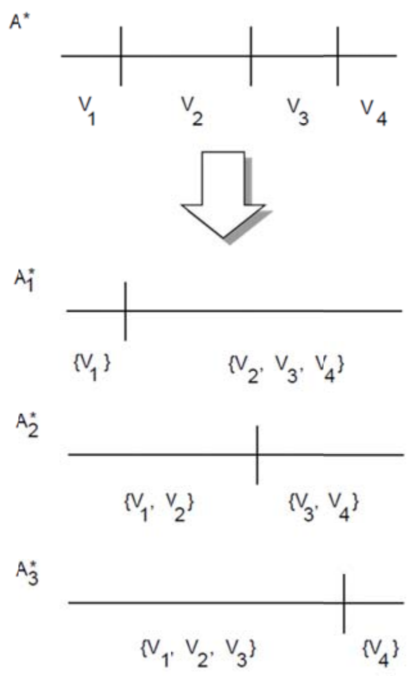


Figure 3-2. A simple ordinal meta-classification scheme.
Adapted from (Frank & Hall, 2001).

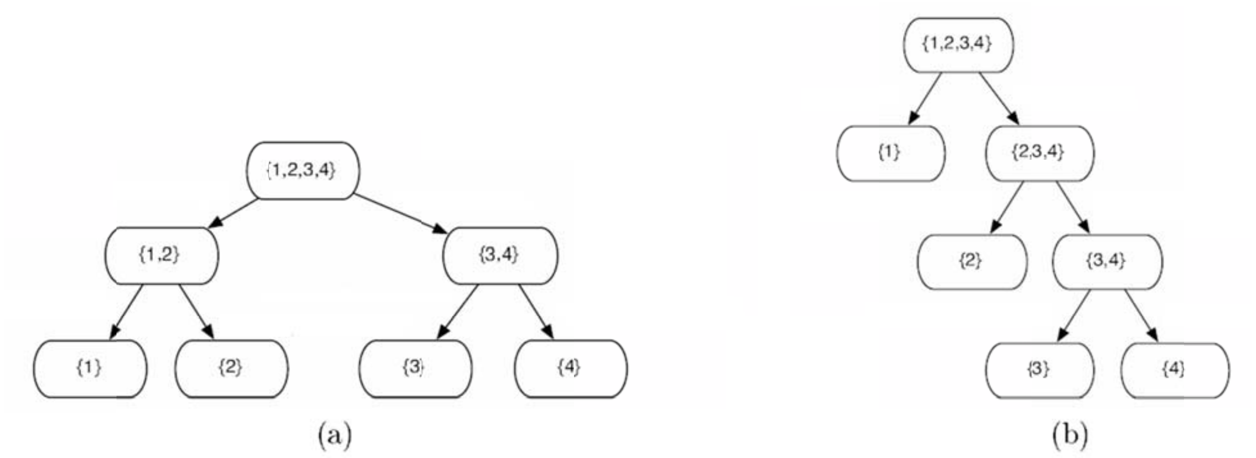


Figure 3-3. Two examples of "nested dichotomy" meta-classifiers.
Adapted from (Frank & Kramer, 2004).

When applied to an ordered problem, they are restricted to ordered splits similar to (Frank & Hall, 2001). The probability distributions along the branches are then averaged to make a class prediction. This process is repeated several times, so that a forest of these randomly generated trees are used for the final classification.

In a study to determine the value that order adds to these methods (Huhn & Hullermeier, 2009), the meta-classifier built from nested dichotomies and Frank's ordinal meta-classifier were compared, using the pairwise meta-classifier as a performance baseline, with a variety of underlying base learners. The pairwise scheme was chosen because in an ordinal context it is not directly disadvantaged, as all pairs of classes are trivially ordered. The results of (Huhn & Hullermeier, 2009) showed several interesting characteristics of these methods: there was some significant advantage when using the ordinal meta-classifiers with flexible base learners (in this case a C4.5 decision tree) in comparison with the pairwise meta-classifier, on what the authors describe as "truly" ordinal data. The advantage is lessened or removed when moving to a less flexible model, such as a linear regression model or a Support Vector Machine with a linear kernel. It may be that the inherent constraints in these models, which attempt to fit a linear problem, are better suited for ordered data (Huhn & Hullermeier, 2009). It is therefore important to pay close attention to these particular Machine Learning/Data Mining methods when benchmarking a new method.

4. METHOD DESCRIPTION

We propose a new method, which we call the *Ordinal Boundary* method, for using historical decisions as training data to estimate weights and threshold preference values produced by the original decision maker, and we describe how to use this information to make new classifications. Both separable problems and non-separable problems are addressed, starting with the 2-class case and extending these approaches for the multi-class case. We also consider a special case of the non-separable problems called the two-plane separable case, in which we have multiple decision makers.

4.1 Separable Problems

The goal of the proposed Ordinal Boundary method is to approximate the model and parameters used by the decision maker as outlined in Chapter 2. We first consider the separable case, in which the historical data is linearly separable. We attempt to fit a linear model, the WSM (given as expression (2-1)), in which a threshold preference value and a weight vector correspond to a linear boundary which divides the attribute space into classification categories (classes). Although the decision maker may not have used this specific model, or may not have been perfectly consistent, the hope is that this model can closely approximate the decision-making process.

4.1.1 Single Dimension Problems

The simplest separable case is a data set containing alternatives with only one attribute (dimension), an example of which is shown in Table 4-1. For clarity, the attribute values in these examples are generated such that they have values in the range of [0,1].

Table 4-1. A single attribute data set.

Alternative	a_1	Class
A_1	0.5	C_1
A_2	0.1	C_1
A_3	0.8	C_2
A_4	1.0	C_2
A_5	0.4	?
A_6	0.6	?

In a single dimension, the boundary between class C_1 and class C_2 is represented by one point in the attribute space. This boundary is referred to here as θ, and in multiple dimensions, the overall preference value of every point on θ is referred to as $P(\theta)$. We use a scale from 0 to 1, such that $\sum_{j=1}^{n} w_j = 1$, to estimate the attribute weights. Thus, $w = 1$ for one attribute problems, $P(\theta) = \theta$, and $P(A_i) = \sum_{j=1}^{n} x_{ij} w_j = x_{i1} w_1 = x_{i1} 1 = x_{i1}$.

The classes in our problems are *ordered*, meaning the preference values for every alternative in class C_1 are less than the preference values for every alternative in class C_2. Because the WSM is being used to approximate the model, this implies the following:

$$\forall A_i \in \text{class } C_1, \text{and } \forall A_k \in \text{class } C_2 \mid \sum_{j=1}^{n} x_{ij}\, w_j \ < \ \sum_{j=1}^{n} x_{kj}\, w_j$$

However, there are an infinite number of feasible boundary planes that will divide the alternatives in class C_1 from those in class C_2. To choose the boundary that best represents our assumptions, we find the two extreme cases. That is, we find the boundary that corresponds to the lowest possible preference value, and the boundary that corresponds to the highest possible preference value, for the data set of interest.

Specifically, we construct an LP (linear programming) model to find the feasible range of $P(\theta)$. That is, the minimum possible preference value of θ, denoted as $P(\theta^{min})$, and the maximum possible preference value of θ, denoted as $P(\theta^{max})$. $P(\theta^{min})$ cannot be lower than the preference value of any class C_1 data point. Likewise, $P(\theta^{max})$ cannot be higher than the preference value of any class C_2 data point. The largest possible difference in these values (that does not violate these constraints) will result in the lowest possible $P(\theta^{min})$ and the highest possible $P(\theta^{max})$ that separates the training examples from class C_1 and class C_2. Thus, the objective value in this formulation maximizes the difference in these two values. In one dimension, there are no weight vectors to estimate, so this is a trivial problem. The formulation is as follows:

Objective function: Maximize $\left(P(\theta^{max}) - P(\theta^{min})\right)$ (4-1)

Such that the following constraints are satisfied:

$\forall A_i \in$ class $C_1 \mid \sum_{j=1}^{n} x_{ij} < P(\theta^{min})$

$\forall A_i \in$ class $C_1 \mid \sum_{j=1}^{n} x_{ij} < P(\theta^{max})$

$\forall A_k \in$ class $C_2 \mid \sum_{j=1}^{n} x_{kj} \geq P(\theta^{min})$

$\forall A_k \in$ class $C_2 \mid \sum_{j=1}^{n} x_{kj} \geq P(\theta^{max})$

and $P(\theta^{min}),\ P(\theta^{max}) \geq 0$.

For this example data, the problem constraints would be:

$0.5 < P(\theta^{min})$	$0.1 < P(\theta^{min})$	$1.0 \geq P(\theta^{min})$	$0.8 \geq P(\theta^{min})$
$0.5 < P(\theta^{max})$	$0.1 < P(\theta^{max})$	$1.0 \geq P(\theta^{max})$	$0.8 \geq P(\theta^{max})$

Table 4-2. Solution values for example data set.

$P(\theta^{min})$	$P(\theta^{max})$
.5	.8

Figure 4-1 shows the example problem from Table 4-1 and the solution values found using formulation (4-1). We can classify alternative A_5 as class C_1, because its preference value, $P(A_5)$, is below the lowest possible preference value, $P(\theta^{min})$, for the boundary represented by θ^{min}. However, the preference value for alternative A_6, $P(A_6)$, lies above $P(\theta^{min})$ and below $P(\theta^{max})$. This puts it into the *uncertainty region,* the region in the attribute space which is not covered by training examples.

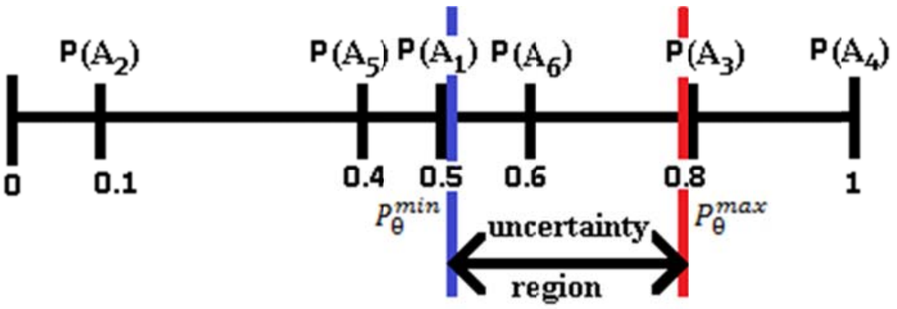


Figure 4-1. Single dimension representation of the example problem.

When new alternatives fall into this region, a simple classification approach is to determine whether the preference value for an alternative is closer to $P(\theta^{min})$ or to $P(\theta^{max})$. In one dimension, one can take the absolute value of the difference in preference values to find which difference is smaller. Figure 4-2 shows these distances for the example in Table 4-1. Here, $d_1 < d_2$, so $P(A_6)$ is closer to $P(\theta^{min})$ and thus A_6 is classified as class C_1.

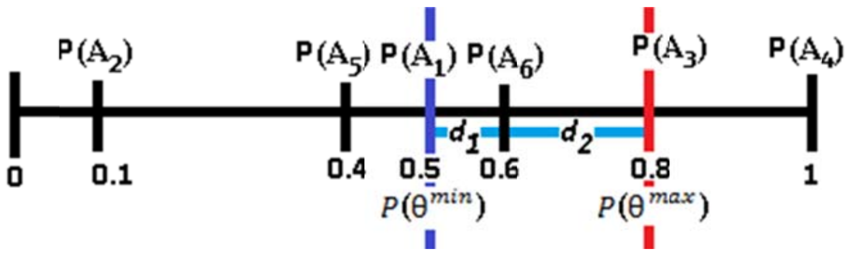


Figure 4-2. Using d_1 and d_2 to classify A_6.

$d_1 = | P(\theta^{min}) - P(A_6) | = 0.1 \qquad d_2 = | P(\theta^{max}) - P(A_6) | = 0.2$

The constraints given in formulation (4-1) use "<" so that the preference values of the data points must fall below the preference values of our minimum values. Data points from both class C_1 and class C_2 cannot be allowed to fall directly on the boundary, as this would create the possibility of an ambiguity when $P(\theta^{max})$ - $P(\theta^{min}) = 0$, so we chose to require that class C_1 data points fall strictly below this boundary. Because the linear solver library (LPSolve) cannot handle this type of constraint, the proposed models use constraints with "≤", and a small constant value ε is chosen to find approximate equivalents to the constraints in (4-1) and further formulations. A method for choosing the value of ε is discussed in Section 4.1.4.

4.1.2 Multiple Attribute Problems

Next we consider problems with two or more attributes. One must now approximate the weights used as well as the preference values for the θ^{min} and θ^{max} class boundaries. A boundary's overall preference value, $P(\theta)$, and a weight vector, w, correspond to a line (or hyper-plane) which divides the attribute space into classification categories. The equation of this boundary line for a problem with two attributes is as follows, where (θ_1, θ_2) is any point on θ, with the overall preference value $P(\theta)$.

$$P(\theta) = \theta_1 w_1 + \theta_2 w_2$$

As in the single dimension case, we construct an LP that estimates the two extremes of θ, θ^{min} and θ^{max}, from their preference values, $P(\theta^{min})$ and $P(\theta^{max})$. Because a single formulation is being used to estimate both the θ^{min} and θ^{max} simultaneously, we must employ a separate weight vector for each boundary. This is because these two boundaries may correspond to different sets of weights. In a problem with two dimensions, the boundaries are defined as:

$P(\theta^{min}) = \theta_1 w_1^{min} + \theta_2 w_2^{min}$, where (θ_1, θ_2) is any point on the boundary θ^{min}.

$P(\theta^{max}) = \theta_1 w_1^{max} + \theta_2 w_2^{max}$, where (θ_1, θ_2) is any point on the boundary θ^{max}.

Before presenting the complete formulation, we examine some examples using a simpler formulation which estimates only θ^{min}.

Objective function: Minimize $P(\theta^{min})$ (4-2)

Such that the following constraints are satisfied:

$$\forall\, A_i \in \text{class } C_1 \mid \sum_{j=1}^{n} x_{ij}\, w_j^{min} < P(\theta^{min})$$

$\sum_{j=1}^{n} w_j^{min} = 1$

$P(\theta^{min}) \geq 0$

and $w_j^{min} \geq 0$, for $j = 1, \dots, n$.

The objective of this formulation is to minimize the value of $P(\theta^{min})$ in a problem with only class C_1 training examples. This demonstrates how the constraints and objective function will "move" the θ^{min} boundary in a geometric representation, in order to clarify the interactions of these constraints in more complicated formulations.

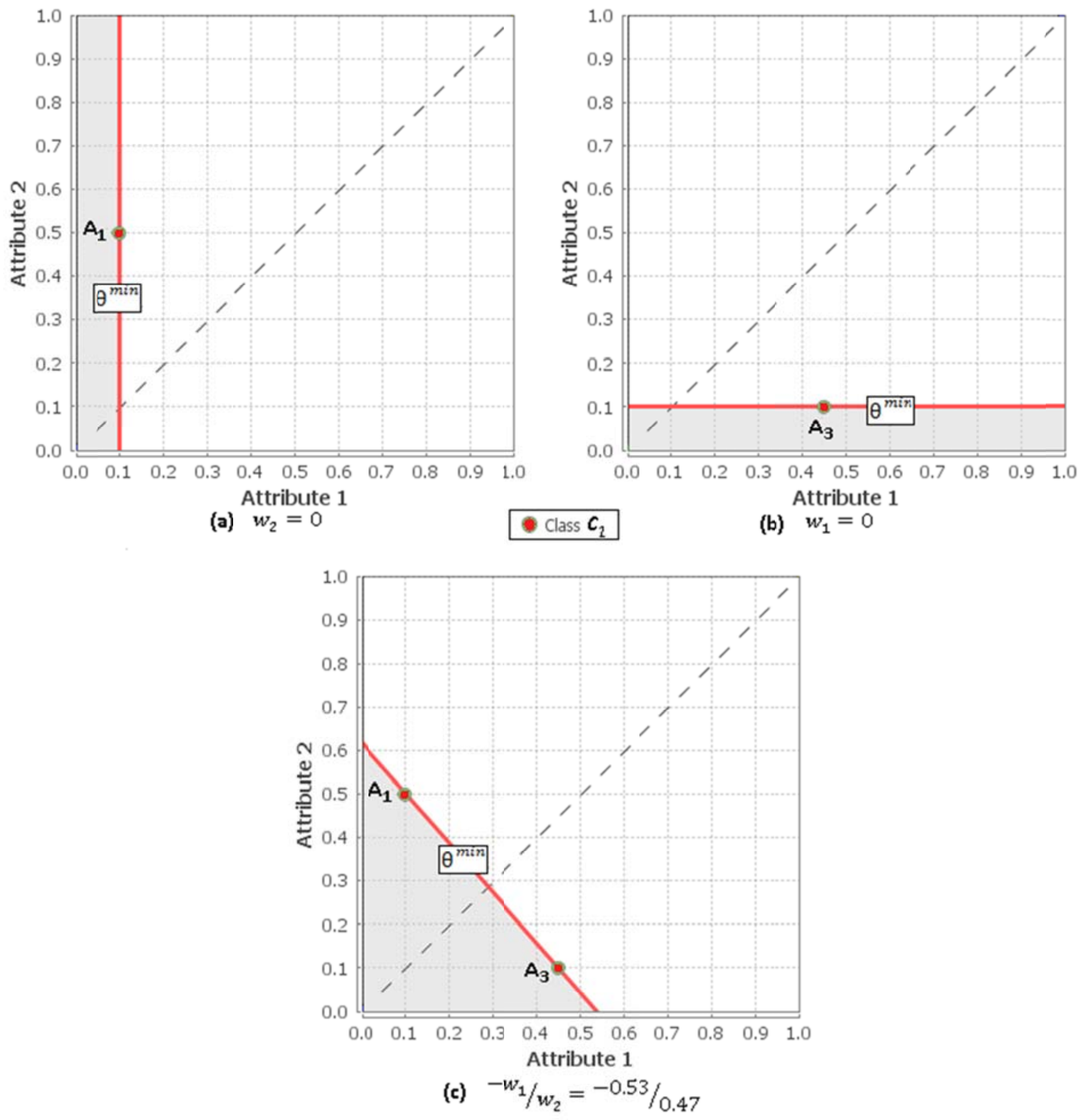


Figure 4-3. Finding the minimum border on simple cases.

Figure 4-3(a) and Figure 4-3(b) show the resulting θ^{min} boundary line with the single training data point A_1. In two dimensions, the slope of θ^{min} is $^{-w_1}/_{w_2}$. Because the weights are always positive, the slope of the boundaries will be negative. The only exception is when one of the weights is 0, as shown in Figure 4-3(a), where $w_2 = 0$ and Figure 4-3(b), where $w_1 = 0$. This objective function of formulation (4-2) moves the "minimum" boundary, θ^{min}, to the position where it has the lowest possible preference value while still covering all class C_1 data points, as shown in Figure 4-3(c).

Formulation (4-3) estimates both the "minimum" and "maximum" boundaries between class C_1 and class C_2 with the addition of weight vectors.

Objective function: Maximize $\left(P(\theta^{max}) - P\left(\theta^{min}\right)\right)$ (4-3)

Such that the following constraints are satisfied:

$\forall\, A_i \in \text{class } C_1 \,|\sum_{j=1}^{n} x_{ij}\, w_j^{min} < P\left(\theta^{min}\right)$

$\forall\, A_i \in \text{class } C_1 \,|\sum_{j=1}^{n} x_{ij}\, w_j^{max} < P(\theta^{max})$

$\forall\, A_k \in \text{class } C_2 \,|\sum_{j=1}^{n} x_{kj}\, w_j^{min} \geq P\left(\theta^{min}\right)$

$\forall\, A_k \in \text{class } C_2 \,|\sum_{j=1}^{n} x_{kj}\, w_j^{max} \geq P(\theta^{max})$

$\sum_{j=1}^{n} w_j^{min} = 1,$ $\qquad \sum_{j=1}^{n} w_j^{max} = 1$

$P\left(\theta^{min}\right),\ P(\theta^{max}) \geq 0$

and $w_j^{min}, w_j^{max} \geq 0$, for $j = 1, \dots, n$.

If this formulation has no feasible solution for a particular data set, this data set falls into the non-separable case, which is addressed in Section 4.4. Once the feasible range of $P(\theta)$ is discovered, an estimation of θ can be made for use in classification. One can be certain the new data points which lie under the θ^{min} boundary are in class C_1, as this boundary cannot be lower without violating the constraints in our formulation. Likewise, one can be certain that new data points which lie above the θ^{max} boundary are in class C_2. These borders are shown in Figure 4-4 for the example given below.

Table 4-3 shows an example data set with two attributes. For clarity, the attribute values in these examples are generated such that they have values in [0,1].

Table 4-3. Example 2-attribute, 2-class problem data.

Alternative	a_1	a_2	Class
A_1	0.1	0.5	C_1
A_2	0.3	0.1	C_1
A_3	0.45	0.1	C_1
A_4	0.4	0.7	C_2
A_5	0.6	0.75	C_2
A_6	0.8	0.4	C_2
A_7	0.1	0.15	?
A_8	0.1	0.6	?

For the example data set in Table 4-3, the constraints are:

$$0.1w_1^{min} + 0.5w_2^{min} < P(\theta^{min}) \qquad 0.1w_1^{max} + 0.5w_2^{max} < P(\theta^{max})$$

$$0.3w_1^{min} + 0.1w_2^{min} < P(\theta^{min}) \qquad 0.3w_1^{max} + 0.1w_2^{max} < P(\theta^{max})$$

$$0.45w_1^{min} + 0.1w_2^{min} < P(\theta^{min}) \qquad 0.45w_1^{max} + 0.1w_2^{max} < P(\theta^{max})$$

$$0.4w_1^{min} + 0.7w_2^{min} \geq P(\theta^{min}) \qquad 0.4w_1^{max} + 0.7w_2^{max} \geq P(\theta^{max})$$

$0.6w_1^{min} + 0.75w_2^{min} \geq P(\theta^{min})$ $\qquad$ $0.6w_1^{max} + 0.75w_2^{max} \geq P(\theta^{max})$

$0.8w_1^{min} + 0.4w_2^{min} \geq P(\theta^{min})$ $\qquad$ $0.8w_1^{max} + 0.4w_2^{max} \geq P(\theta^{max})$

$$w_1^{min} + w_2^{min} = 1, \quad w_1^{max} + w_2^{max} = \mathbf{1}, \quad P(\theta^{min}), P(\theta^{max}) \geq 0$$

and $w_1^{min}, w_2^{min}, w_1^{max}, w_2^{max} \geq 0$.

Table 4-4. Solution values for example data set.

w_1^{min}	w_2^{min}	w_1^{max}	w_2^{max}	$P(\theta^{min})$	$P(\theta^{max})$
.53	.47	.43	.57	.29	.57

The solution values for this problem are listed in Table 4-4. Figure 4-4(a) shows these boundaries and the uncertainty region between them.

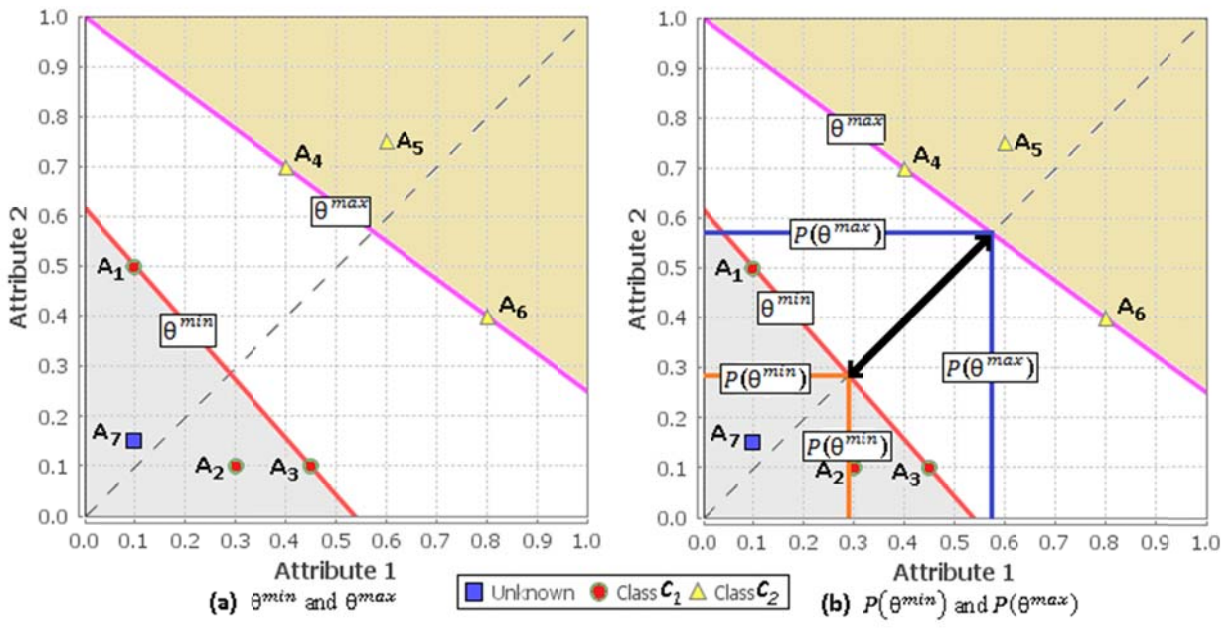


Figure 4-4. Boundaries for example data set.

Figure 4-4(b) shows how the preference values of θ^{min} and θ^{max} are represented geometrically. The preference value, $P(\theta)$, is equal to the value of θ_1 (and θ_2) when $\theta_1 = \theta_2$, which is the data point where the boundary θ crosses the *attribute*~1~ = *attribute*~2~ plane. This is because (if calculated using the WSM), $P(\theta)$ is a projection of the boundary θ onto this plane. Figure 4-4(b) also shows a geometric representation of the objective function, which maximizes the distance along this plane between $P(\theta^{min})$ and $P(\theta^{max})$.

Figure 4-5(a) shows that A_7 falls below the minimum possible class boundary, and we can safely classify it as class C_1. However, a new data point (unclassified alternative) may lie in the uncertainty region, above θ^{min} and below θ^{max}. An example is shown in Figure 4-5(a), where A_8 is between the two feasible class boundaries, thus we cannot be sure which class it belongs to.

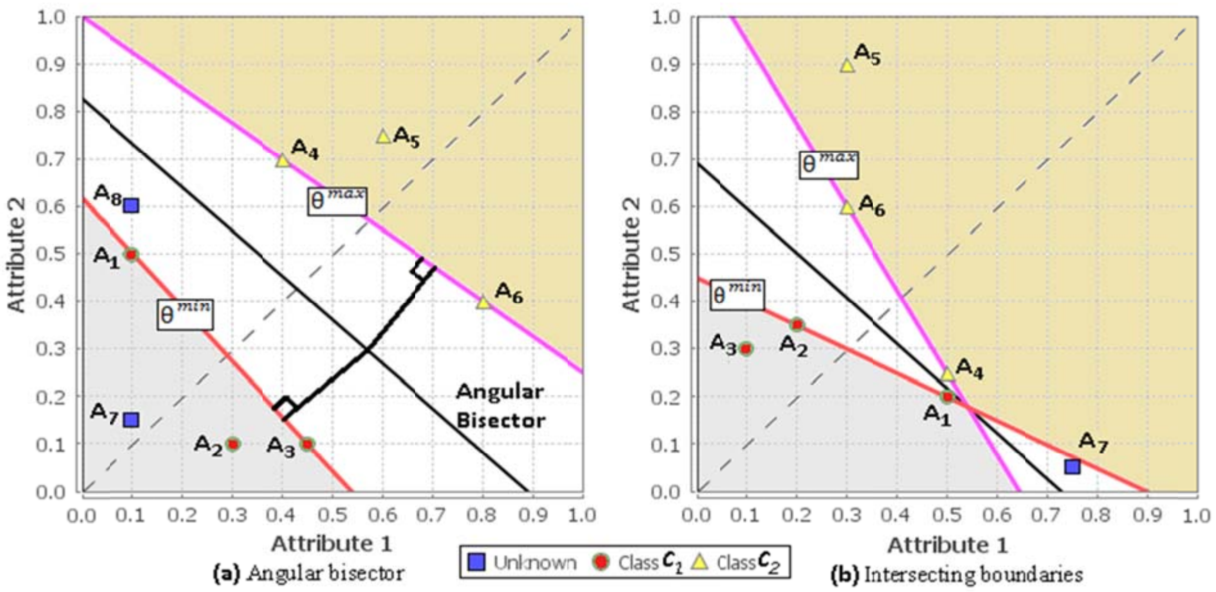


Figure 4-5. Angular bisector of intersecting and non-intersecting boundaries.

A classification for such an alternative can be made by judging which class region of the space it is closest to. To facilitate this classification process, we formulate a new boundary using the angular bisector of the boundary lines for θ^{min} and θ^{max}. The angular bisector is the locus of points whose perpendicular distance is equidistant from two boundaries as shown in Figure 4-5(b).

Figure 4-5(b) uses a different set of example data points to demonstrate the advantage of using the angular bisector for classifying data points in the uncertainty region. Unlike the first example, in this new example θ^{min} and θ^{max} intersect, and the unclassified data point A_7 is in an unpopulated area of the space. The lack of training data points in this area can result in the θ^{min} and θ^{max} borders intersecting, as shown in this example. In such cases, a data point may be both below θ^{min} and above θ^{max} simultaneously, and thus the border that it is closest to will not represent the region it is closest to. If A_7 was classified based on which border it was closer to, it would be in class C_1. However, because the angular bisector of these two solutions is used as the boundary between classes, A_7 is classified as part of the region that it is actually closer to, which is class C_2.

In general, an algebraic method for the expression of the angular bisectors of two lines L_1 and L_2, where $L_{1:}\ a_1x + b_1y + c_1 = 0$ and L_2: $a_2x + b_2y + c_2 = 0$, is the solution to the equation:

$$\left|\frac{a_1x + b_1y + c_1}{\sqrt{a_1^2 + b_1^2}}\right| = \left|\frac{a_2x + b_2y + c_2}{\sqrt{a_2^2 + b_2^2}}\right|$$

Of the two angular bisectors, the one of interest is the one shown in Figure 4-5(a), which has a negative slope. Because the example data uses benefit criteria with weights in the range [0,1], the boundaries will never have a positive slope. Thus one must solve:

$$\frac{a_1x + b_1y + c_1}{\sqrt{a_1^2 + b_1^2}} = -\frac{a_2x + b_2y + c_2}{\sqrt{a_2^2 + b_2^2}}$$

In two dimensions, the boundary lines are defined as:

$P(\theta^{min}) = \theta_1 w_1^{min} + \theta_2 w_2^{min}$, where (θ_1, θ_2) is any point on the boundary θ^{min}.

$P(\theta^{max}) = \theta_1 w_1^{max} + \theta_2 w_2^{max}$, where (θ_1, θ_2) is any point on the boundary θ^{max}.

To find the angular bisector of interest for the example in Table 4-3, substitute:

$a_1 = w_1^{min}$, $b_1 = w_2^{min}$, and $c_1 = -P(\theta^{min})$

$a_2 = w_1^{max}$, $b_2 = w_2^{max}$, and $c_2 = -P(\theta^{max})$, which yields:

$$\frac{0.53x + 0.47y - 0.29}{\sqrt{0.53^2 + 0.47^2}} + \frac{0.43x + 0.57y - 0.57}{\sqrt{0.43^2 + 0.57^2}} = 0$$

This simplifies to: $1.35x + 1.46y - 1.21 = 0$

To scale the *w* vector on a range from 0 to 1, normalize this equation s.t. $w_1 + w_2 = 1$:

$$\frac{1.35x + 1.46y - 1.21}{(1.35 + 1.46)} = 0$$

This simplifies to: $0.48x + 0.52y - 0.43 = 0$ or $0.48x + 0.52y = 0.43$

According to this equation of the bisecting boundary line, $w_1 = 0.48$ and $w_2 = 0.52$, and $P_\theta = 0.43$. Now that the w vector for the bisecting boundary is available, one can use it to calculate the value of (A_8) :

$$P(A_8) = 0.48(0.1) + 0.52\,(0.6) = 0.34$$

Note that $P(A_8) < P(\theta)$, thus we classify A_8 as being in class C_1.

Table 4-5. Bisecting solution.

w_1	w_2	P_θ	$P(A_8)$
0.48	0.52	0.43	0.34

The angular bisector classification approach can be extended to n dimensions, in which case, the bisector is the plane which is the locus of all the points equidistant from the hyper-planes represented by the minimum and maximum solutions. One would solve this equation:

$$\frac{a_1x \ + b_1y + \ldots + n_1z + p_1}{\sqrt{a_1^2 + b_1^2 + \ldots + n_1^2}} = -\frac{a_2x \ + b_2y + \ldots + n_2z + p_2}{\sqrt{a_2^2 + b_2^2 + \ldots + n_2^2}}$$

4.1.3 Boundary Interference

To find the extremes of the possible $P(\theta)$ values, formulation (4-3) effectively moves the θ^{min} boundary line as close to the region of class C_1 (and thus as far from class C_2) as possible, and moves the θ^{max} boundary as close to the region of class C_2 as possible. As illustrated in Section 4.1.2, $P(\theta^{min})$ is the value of attribute$_1$ (and attribute$_2$) where θ^{min} crosses the attribute$_1$=attribute$_2$ plane. Thus formulations that minimize $P(\theta^{min})$ will move the θ^{min} boundary as far along this plane towards the origin as possible, as this will result in the lowest value for $P(\theta^{min})$. However, training examples from class C_2 may interfere with the θ^{min} boundary, and likewise, training examples from class C_1 can interfere with the θ^{max} boundary. These types of interference are shown in Figure 4-6.

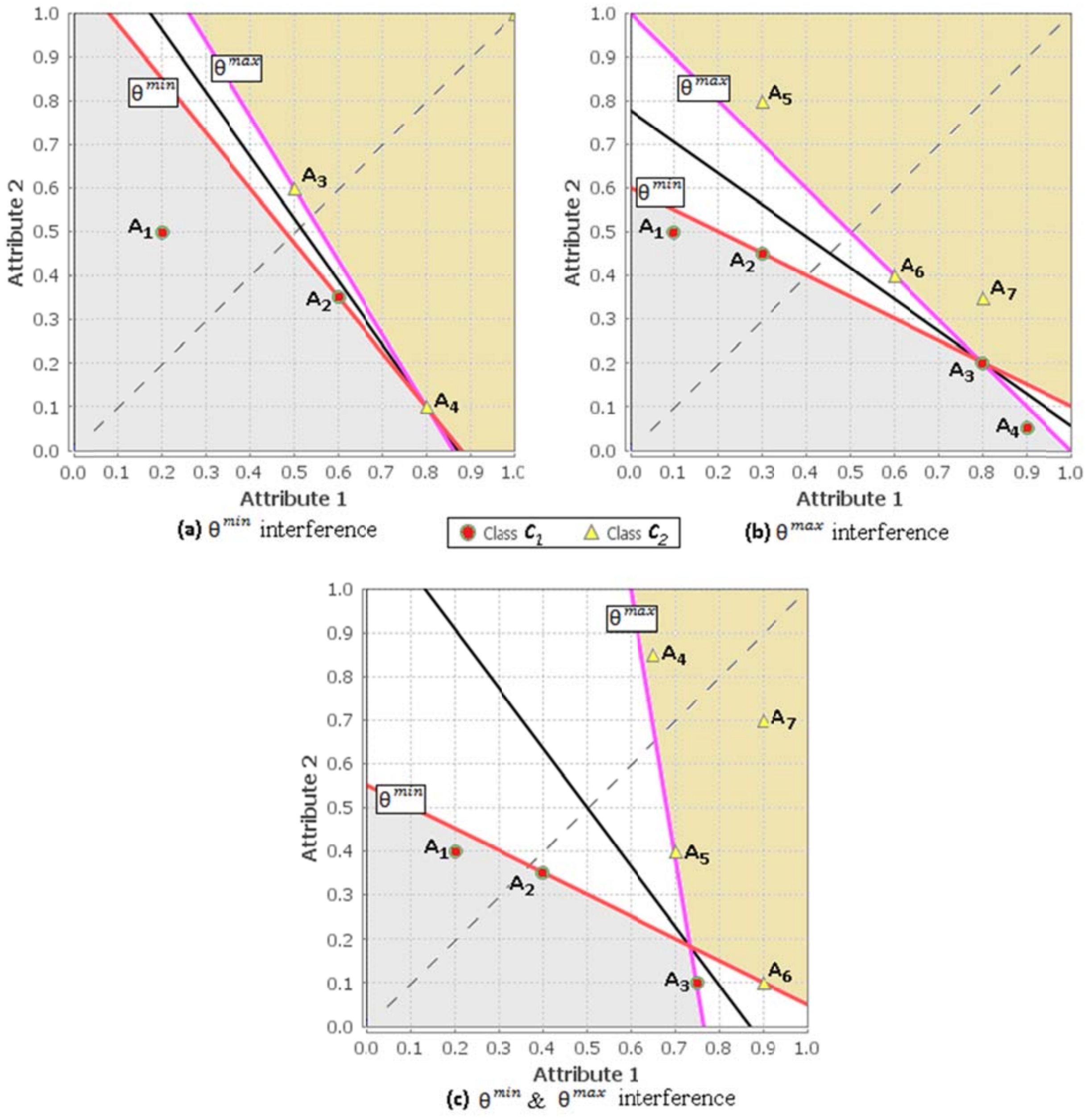


Figure 4-6. Types of boundary interference.

Figure 4-6(a) shows A_4 interfering with the solution corresponding to the θ^{min} boundary. Without A_4 present, the θ^{min} boundary would be as close as possible to A_1 and A_2. Figure 4-6(b) shows A_3 interfering with the solution corresponding to the θ^{max} boundary in a similar fashion. Figure 4-6(c) shows both A_3 and A_6 interfering with the solution. In all of these cases, the bisector evenly divides the uncertainty region between class C_1 and class C_2, such that data points above it will be classified as class C_2 and data points below it will be classified as class C_1.

4.1.4 Choosing a value for ε

As mentioned in Section 4.1.1, a small value for ε must be chosen to allow for the use of "<" constraints in these formulations. To find a sufficiently small value of ε, we restrict the minimum and maximum boundaries to a single weight vector, which effectively forces them to be parallel. This is very similar to the way a basic Support Vector Machine (SVM) is constructed, but here the distance along the $\text{attribute}_1 = \text{attribute}_2$ (in 2-dimension problems) plane is maximized rather than the perpendicular distance between the support vectors in an SVM.

Objective function: Maximize δ (4-4)

$\forall A_i \in \text{class } C_1 \mid \sum_{j=1}^{n} x_{ij} w_j \leq P(\theta) - \delta$

$\forall A_k \in \text{class } C_2 \mid \sum_{j=1}^{n} x_{kj} w_j \geq P(\theta) + \delta$

$\sum_{j=1}^{n} w_j = 1$

$P(\theta) \geq 0$

and $w_j \geq 0$, for $j = 1, \ldots, n$.

From the solution, we select ε such that $\varepsilon < \delta$, where δ is half the difference in the preference values corresponding to the boundary lines L_1 and L_2, defined as:

$$L_1 : \sum_{j=1}^{n} \theta_j w_j = P(\theta) - \delta$$

$$L_2 : \sum_{j=1}^{n} \theta_j w_j = P(\theta) + \delta$$

Figure 4-7(a) shows an example problem solved with formulation (4-4). Figure 4-7(b) shows the uncertainty region for the same example when solved with the formulation (4-3), where the boundary lines θ^{max} and θ^{min} have independent weight vectors and thus no such parallel constraint.

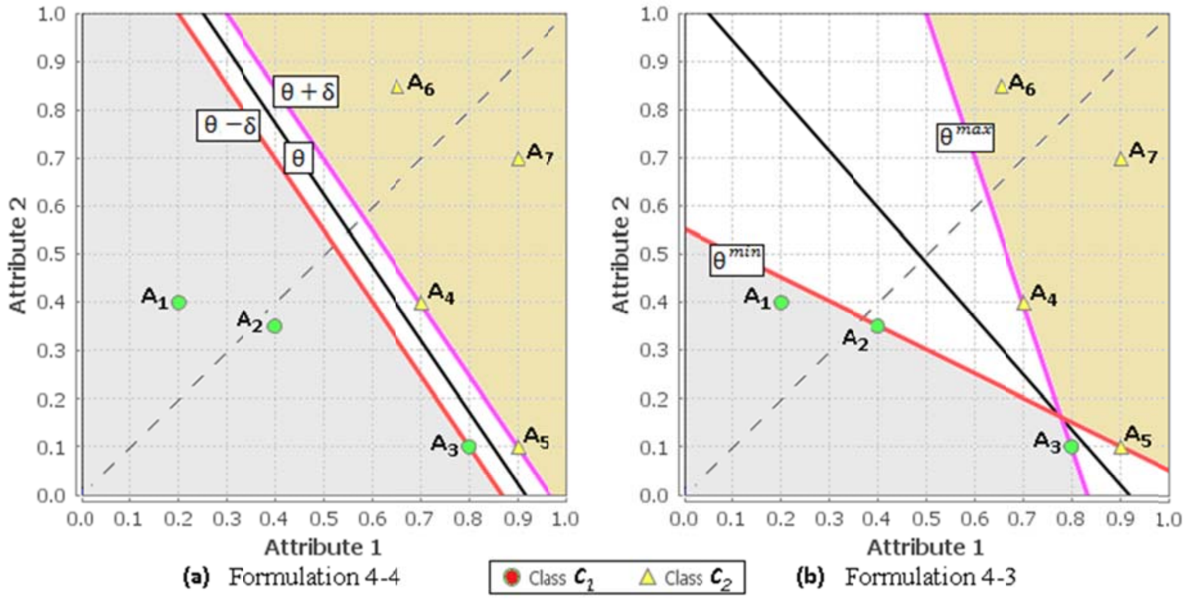


Figure 4-7. Parallel boundaries for finding δ. "θ − δ" and "θ + δ" are being used to represent the L_1 and L_2 boundaries as defined above, respectively.

Formulation (4-4) can also be used as a classification method. However, because formulation (4-4) uses a single weight vector, the region between the L_1 and L_2 boundaries in Figure 4-7(a) is much smaller than the uncertainty region indicated by the solution for formulation (4-3) shown in Figure 4-7(b). The actual θ^{min} boundary has a much lower preference value than the $\theta - \delta$ boundary, and the θ^{max} boundary has a much higher preference value than the $\theta + \delta$ boundary. If the analyst is interested in whether a new data point is in the uncertainty region as well as how it is ultimately classified, this is a weakness of using a method like formulation (4-3). A solution given by formulation (4-3) accurately reflects the maximum possible range in preference values and the corresponding boundaries. Otherwise, this formulation may be an alternative approach for classification of separable data sets when combined with the method described in Section 4.1.3. Experiments to this effect are discussed in Chapter 5.

4.1.5 Leveraging order in multi-class MADM problems

In the literature review, both the textbook meta-classification schemes for dealing with multi-class problems and the ordinal meta-classification schemes from recent research were described. However, all of those methods ignore useful information about any MADM problem in which the WSM or a similar model was used for decision-making.

A consistently applied model calculates the preference values for all given alternatives using the same weight vector. As a result, in any data set produced by a decision maker employing such a model, all the boundaries between classes have the same coefficients for their corresponding boundary lines, (or hyper-planes in *n* dimensions), and thus these boundaries are parallel. This allows one to use information given by a training data point between class C_1 and class C_2 to restrict the possible range of boundaries between class C_2 and class C_3, class C_3 and class C_4, etc., which is how one may take advantage of the ordinal properties in the classes to further restrict the range of potential values for $P(\theta)$.

The modified version of the proposed Ordinal Boundary method uses an optimization formulation that estimates a single pair of weight vectors, w^{min} and w^{max}, to estimate the minimum and maximum boundaries between all classes simultaneously. This is an extension of (4-1) with similar constraints, but with $\theta_1, \ldots, \theta_{m-1}$ pairs of min and max boundaries. The boundaries $\theta_1^{min}, \theta_2^{min}, \ldots, \theta_{m-1}^{min}$ all use the weight vector w^{min}, while the boundaries $\theta_1^{max}, \theta_2^{max}, \ldots, \theta_{m-1}^{max}$ all use the weight vector w^{max}.

However, using a single pair of weight vectors may make it impossible to maximize the difference in the preference values between all pairs of boundaries simultaneously. The goal for the new objective function is to separate the boundaries as much as possible while not violating this restriction on the weight vectors. To this end, the objective function maximizes the sum of these distances. The constraints ensure that the minimum boundary slopes are the same, as the θ^{min} all share the same weight vector, w^{min} (and likewise the maximum boundary slopes are the same, as all of the θ^{max} all share the weight vector w^{max}). There are other possible approaches for constructing this objective function, such as maximizing the minimum difference in preference values for each pair of boundaries, or using an approach such as formulation (4-4), which creates boundaries similar to the support vectors of a Support Vector Machine. These alternatives are discussed further in Section 4.1.6, and their effectiveness is tested in Chapter 5.

The formulation for a 3-class problem is given in formulation (4-5).

Objective function: Maximize $\sum_{p=1}^{3}(P(\theta_p^{max}) - P(\theta_p^{min}))$ (4-5)

Such that the following constraints are satisfied:

$\forall A_i \in$ class $C_1 \mid \sum_{j=1}^{n} x_{ij}\, w_j^{min} < P(\theta_1^{min})$

$\forall A_i \in$ class $C_1 \mid \sum_{j=1}^{n} x_{ij}\, w_j^{max} < P(\theta_1^{max})$

$\forall A_k \in$ class $C_2 \mid \sum_{j=1}^{n} x_{kj}\, w_j^{min} \geq P(\theta_1^{min})$

$\forall A_k \in$ class $C_2 \mid \sum_{j=1}^{n} x_{kj}\, w_j^{max} \geq P(\theta_1^{max})$

$\forall A_k \in$ class $C_2 \mid \sum_{j=1}^{n} x_{kj}\, w_j^{min} < P(\theta_2^{min})$

$\forall A_k \in$ class $C_2 \mid \sum_{j=1}^{n} x_{kj}\, w_j^{max} < P(\theta_2^{max})$

$\forall A_m \in$ class $C_3 \mid \sum_{j=1}^{n} x_{mj}\, w_j^{min} \geq P(\theta_2^{min})$

$\forall A_m \in$ class $C_3 \mid \sum_{j=1}^{n} x_{mj}\, w_j^{max} \geq P(\theta_2^{max})$

$\sum_{j=1}^{n} w_j^{min} = 1, \qquad \sum_{j=1}^{n} w_j^{max} = 1$

$P(\theta_p^{max}), P(\theta_p^{min}) \geq 0$

and $w_j^{min}, w_j^{max} \geq 0$, for $j = 1, \ldots, n$.

As shown in Figure 4-8(a), after the estimation of these weight vectors and pairs of boundaries, one can employ the same approach as in the 2-class case to find the bisector between each pair of ordered classes. This provides the estimation of the decision maker's threshold preference value and single weight vector. To classify a new alternative A_i, this weight vector can be used to find the preference value, $P(A_i)$. Then one can use a binary search to find which pair of bisectors that the preference value $P(A_i)$ falls between. The class value corresponding to the region between these two boundaries is then assigned to the new alternative.

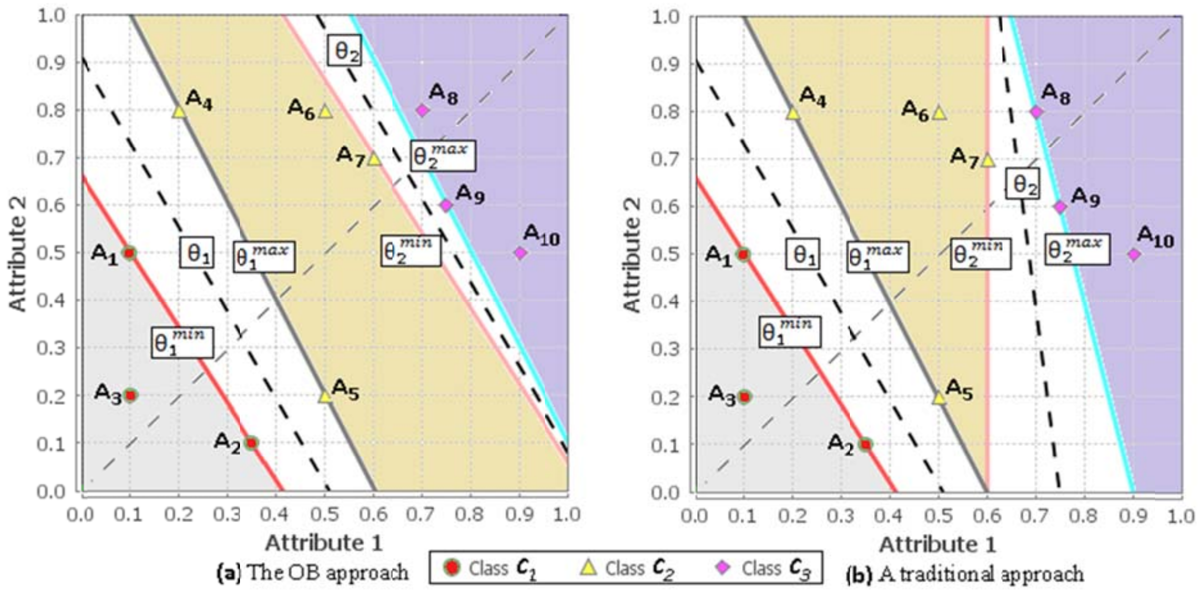


Figure 4-8. An example of a three-class data set with two pairs of boundaries.

Figure 4-8 also shows an example of the advantage gained from estimating all of the boundaries between classes simultaneously, instead of estimating each border independently of the others. When estimating these boundaries simultaneously according to formulation (4-4), all of the minimum boundaries will be parallel, because they share the same weight vector and thus the same slope (in two dimensions.) Similarly, the maximum boundaries will also be parallel. Because a consistent linear classification process will result in parallel boundaries, restricting the estimation process to boundaries that are parallel may increase the accuracy of the estimation process. This was confirmed in the computational tests described in Chapter 5.

This can be seen in Figure 4-8(a), where θ_1^{min} and θ_2^{min} are parallel, and θ_1^{max} and θ_2^{max} are parallel. In cases such as this example, using this restriction causes information from class C_1 and class C_2 data points to affect the estimation of the θ_2 boundary. In the example shown in Figure 4-8(a), A_2 is restricting the position of the θ_2^{min} boundary. In contrast, Figure 4-8(b) shows the combined results of two binary classifiers built using the 2-class formulation (4-3), in the manner a typical binary meta-classification scheme would be applied.

It should be noted that Figure 4-8(b) shows the boundaries of these two independent binary classifiers on a single graph, for convenience. The first binary classifier finds the border labeled as θ_1 using data points only from class C_1 and class C_2 for training. The second is labeled as θ_2 using data points only from class C_2 and class C_3 for training. It should also be noted that although the θ_1^{min} and θ_1^{max} boundaries estimated by the proposed Ordinal Boundary method (shown in Figure 4-8(a)) are identical to the θ_1^{min} and θ_1^{max} boundaries estimated by the first binary classifier (shown in Figure 4-8(b)), this will not always be the case.

Figure 4-8(a) shows that the possible range of the θ_2 boundary is much more restricted than it is in the approach shown in Figure 4-8(b). Given that the original data was created in a consistent manner, where the same weights were applied to all alternatives, the θ_2^{min} shown in Figure 4-8(b) is not a valid boundary. This is because the θ_2^{min} boundary in Figure 4-8(b) has a w vector of [1,0]. It can be seen in Figure 4-8(b) that no boundary between class C_1 and class C_2 could have such a w vector, as there is no possible vertical boundary between the data points in Figure 4-8(a). As a result, the

$P(\theta_2^{min})$ value shown in Figure 4-8(b) is outside the range of possible $P(\theta_2)$ values. Because the proposed Ordinal Boundary method incorporates these restrictions where traditional multi-class meta-classification schemes do not, it has a potential advantage when the historical data was indeed classified in a consistent manner.

4.1.6 Alternative formulations

As discussed in Section 4.1.5, there are several alternative objective functions that can be used for the proposed Ordinal Boundary Method for multi-class problems. One possibility is a version of formulation (4-5) that maximizes the minimum difference in the boundary preference values instead of the sum of those differences. A 3-class version is given in formulation (4-6), and problems with larger numbers of classes can be developed in a similar manner.

Objective function: Maximize Y (4-6)

Such that the following constraints are satisfied:

$\forall A_i \in \text{class } C_1 \mid \sum_{j=1}^{n} x_{ij}\, w_j^{min} < P(\theta_1^{min})$

$\forall A_i \in \text{class } C_1 \mid \sum_{j=1}^{n} x_{ij}\, w_j^{max} < P(\theta_1^{max})$

$\forall A_k \in \text{class } C_2 \mid \sum_{j=1}^{n} x_{kj}\, w_j^{min} \geq P(\theta_1^{min})$

$\forall A_k \in \text{class } C_2 \mid \sum_{j=1}^{n} x_{kj}\, w_j^{max} \geq P(\theta_1^{max})$

$\forall A_k \in \text{class } C_2 \mid \sum_{j=1}^{n} x_{kj}\, w_j^{min} < P(\theta_2^{min})$

$\forall A_k \in \text{class } C_2 \mid \sum_{j=1}^{n} x_{kj}\, w_j^{max} < P(\theta_2^{max})$

$\forall A_m \in \text{class } C_3 \mid \sum_{j=1}^{n} x_{mj}\, w_j^{min} \geq P(\theta_2^{min})$

$\forall A_m \in \text{class } C_3 \mid \sum_{j=1}^{n} x_{mj}\, w_j^{max} \geq P(\theta_2^{max})$

$Y < P(\theta_p^{max}) - P(\theta_p^{min}), \quad$ for $p = 1, 2$

$\sum_{j=1}^{n} w_j^{min} = 1, \quad \sum_{j=1}^{n} w_j^{max} = 1$

$P(\theta_p^{max}), P(\theta_p^{min}) \geq 0,$ for $p = 1, 2$

and $w_j^{min}, w_j^{max} \geq 0,$ for $j = 1, \dots, n.$

Another is a multi-class version of formulation (4-4). This is similar to a SVM, but rather than maximizing the perpendicular distance between support vectors, it maximizes the distance along the attribute$_1$=attribute$_2$ plane (in a 2-attribute problem), as the preference values are projections along this plane. We chose to use an objective function using the sum of differences in preference values, similar to formulation (4-5). The max-min approach shown in formulation (4-6) could also be used. The 3-class version is given in formulation (4-7), and problems with larger numbers of classes can be developed in a similar manner. These alternative objective functions were tested in Section 5.3.2.

Objective function: Maximize $\sum_{p=1}^{2} \delta_p$ (4-7)

$\forall A_i \in$ class $C_1 \mid \sum_{j=1}^{n} x_{ij} w_j \leq P(\theta_1) - \delta_1$

$\forall A_k \in$ class $C_2 \mid \sum_{j=1}^{n} x_{kj} w_j \geq P(\theta_1) + \delta_1$

$\forall A_k \in$ class $C_2 \mid \sum_{j=1}^{n} x_{kj} w_j \leq P(\theta_2) - \delta_2$

$\forall A_m \in$ class $C_3 \mid \sum_{j=1}^{n} x_{mj} w_j \geq P(\theta_2) + \delta_2$

$\sum_{j=1}^{n} w_j = 1, \ P(\theta) \geq 0$

and $w_j \geq 0,$ for $j = 1, \dots, n.$

4.2 Multiple decision maker problems

A special case to consider is one in which there are multiple decision makers determining the classification of each record in a data set. Instead of estimating a single weight vector w and set of threshold preference values $P(\theta)$, one must now consider a weight vector and a set of threshold values for each decision maker. We explore the simplest version of this problem, which has 2 decision makers, 2 classes, and 2 attributes. This is equivalent to finding two boundary lines to divide class C_1 and class C_2 instead of one. These problems fall into one of two cases.

In Case I, the decision to classify a data point as class C_2 instead of class C_1 is the result of an agreement between both decision makers that the record should be classified so, shown in Figure 4-9(a). Thus, the data point must lie above the boundary lines for both decision makers to be classified as class C_2. In Case II, the decision to place a data point in class C_2 instead of class C_1 is the result of at least one decision makers classifying the data point so, shown in Figure 4-9(b). Thus, the data point must lie above the boundary lines for at least one decision maker. Without prior knowledge of which case may apply to a particular data set, one can attempt to find a feasible solution for each one separately.

Consider Case I. Class C_1 data points fall into a concave region consisting of the sub-regions labeled I, II, and III in Figure 4-9(a). If a solution is feasible, there must exist two subsets of class C_1 data points, A^1 and A^2, such that each are linearly separable from the data points in class C_2, and $A^1 \cup A^2 =$ class C_1. Once these subsets are known, one can apply the approach for finding a single boundary line.

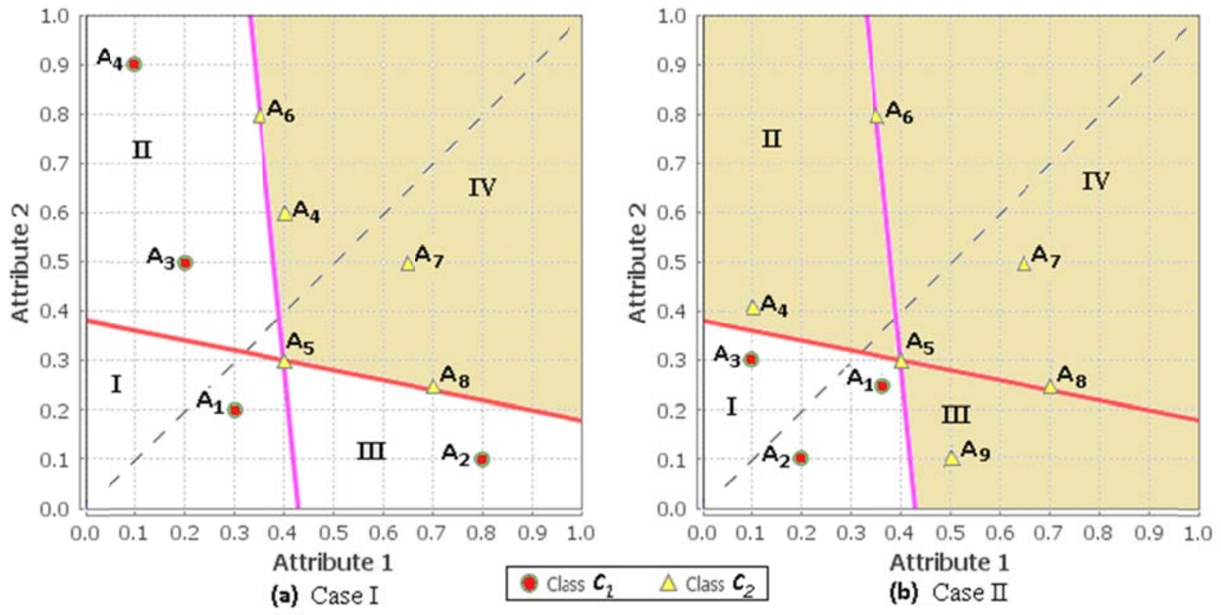


Figure 4-9. Examples of Case I and Case II.

To find these subsets, start with $A^1 =$ class C_1 , $A^2 = \{\}$ and iteratively move data points from A^1 to A^2. The goal is for A^1 to include all data points in region II, and for A^2 to include all data points in region III. Data points in region I are covered by both lines, and can be included in either subset. One can decide what data points to move by observing that all data points in region II have greater values in Attribute$_2$ than Attribute$_1$ in comparison with region III. Therefore, by moving data points in A^2 with the highest values in Attribute$_1$ into A^1 first, all region III data points will be moved before any region II data points are moved.

To find a 2-plane solution (in two dimensions this will be a 2-line solution) for two classes one proceeds as follows. First, divide class C_1 data points into two subsets A^1 and A^2, such that: $A^1 =$ class C_1, $A^2 = \{\}$. Then, construct two linear programs (LPs), LP$_1$ and LP$_2$, using formulation (4-4), such that LP$_1$ uses as training A^1 and class C_2 data points (Figure 4-10(b)), and LP$_2$ uses as training A^2 and class C_2 data points (Figure 4-10(c)).

Once an attempt to solve these two LPs is made, one can use the results to make some determinations. First, if both are solvable, a 2-plane separable solution has been found (Figure 4-10(d)). On the first iteration, A^2 is empty, so if both LPs are solvable this will show that a 2-plane separable approach was not needed, as this will be a linearly separable solution with a single boundary line between A^1 and class C_2 data points.

If both LPs are unsolvable, there is no solution. This is not possible on the first iteration, as A^2 is empty, so LP_2 is trivially solvable, but on subsequent iterations, if a situation arises where LP_2 becomes unsolvable before LP_1 becomes solvable, there is no possible 2-plane solution. If only LP_2 is solvable, which will be the case in the first iteration, one needs to move some of the data points from region III into A^2. Therefore, choose the A^1 data point (or data points) with the highest Attribute_1 value, move it to A^2, and repeat the attempt to solve both LPs on the new training sets.

As soon as both LPs are solvable, a feasible 2-plane solution for the training set has been found. This process can be continued until LP_1 is unsolvable to find the complete set of ways to divide class C_1 data points into A^1 and A^2. This process can also be performed from the opposite direction, i.e., by moving data points with the highest Attribute_2 value from A^1 into A^2, essentially moving region I data points into A^2 instead of region III data points. Finally, Case II can be approached using the same steps. In this case, one would start with $A^1 =$ class C_2 instead of class C_1, sorted as before, and move data points into A^2 in the same manner.

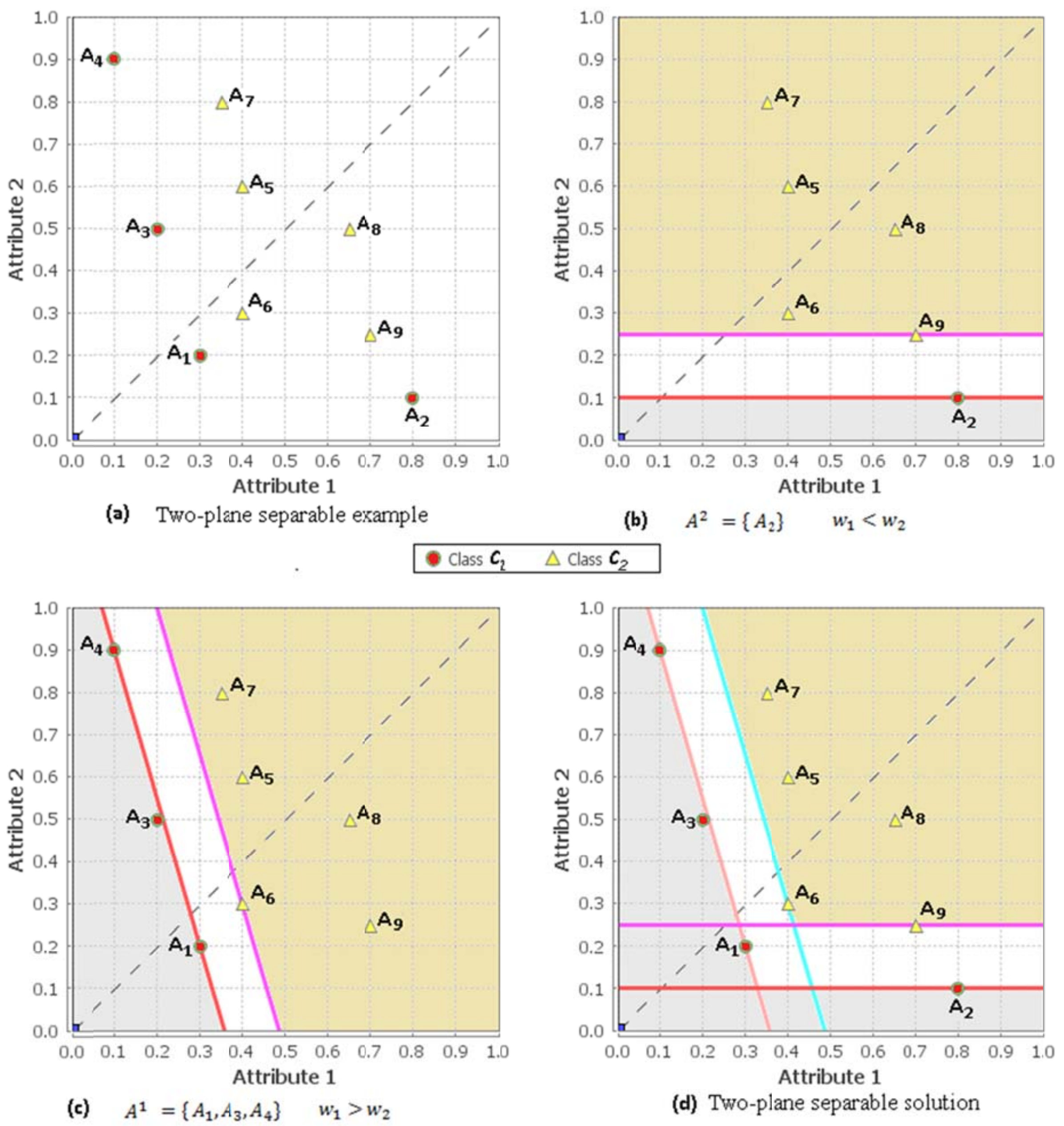


Figure 4-10. 2-plane separable problem example.

In order to reduce unnecessary iterations, one can guess at a starting point with which to subdivide class C_1. A good starting point would be a class C_2 data point that is close to the intersection of the two boundary lines. To this end, one can find the class C_2 data point with the lowest preference value according to the LP solution for (4-3.) Then, one can find the line which intersects this data point in class C_2 and is perpendicular to the boundary line which it lies on.

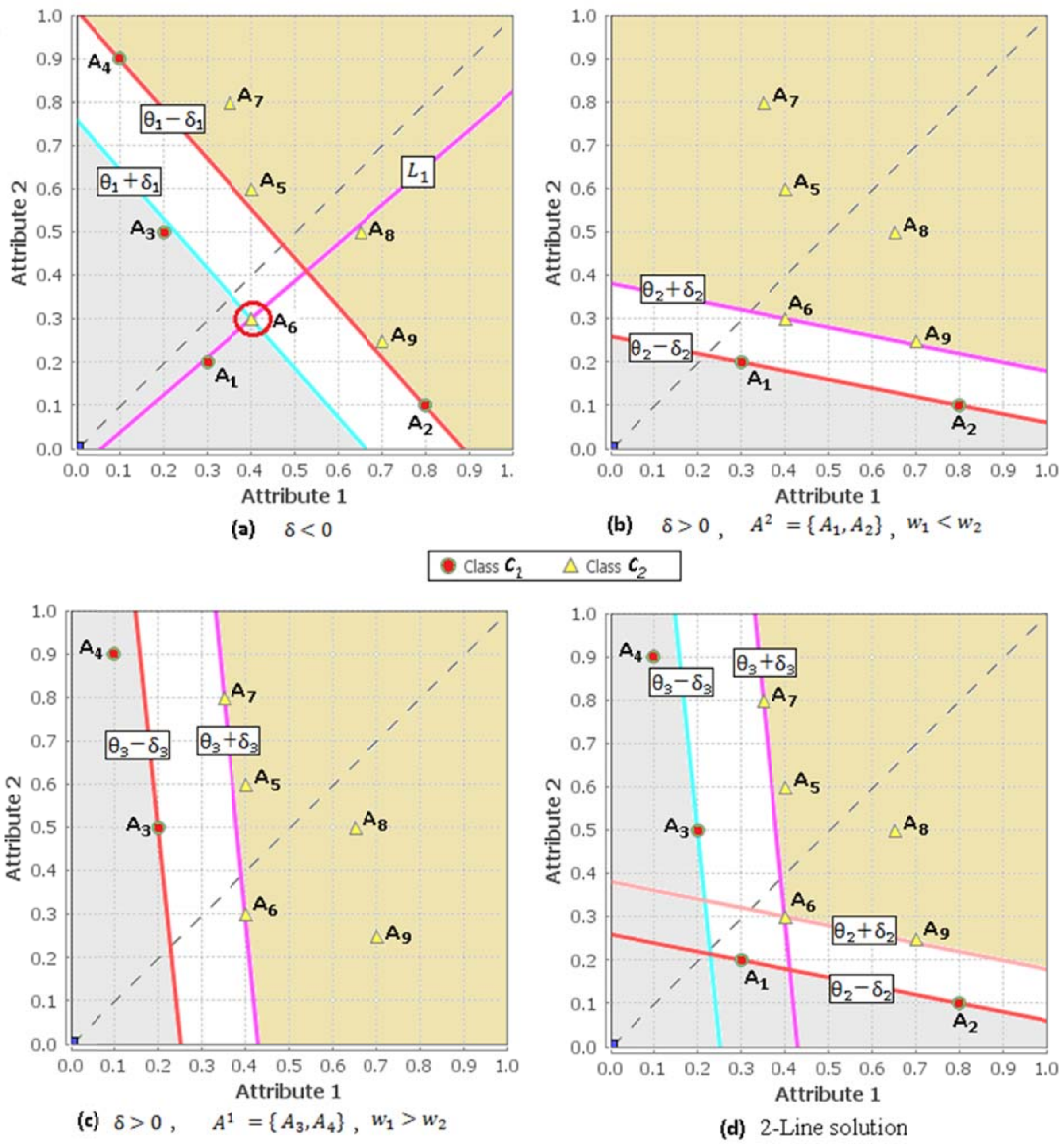


Figure 4-11. 2-plane separable problem revised approach.

This choice gives a good starting guess because, when using formulation (4-4) on a non-separable problem, having the lowest preference value means it is the most deeply positioned into the class C_1 region of all class C_2 data points. In many test cases created from randomized data, this starting point provided an immediate solution.

The revised approach using this is as follows. First, solve the LP (4-4) for finding the value of δ on class C_1 and class C_2 data points, as shown in Figure 4-11(a). If δ is positive, this is a single plane solution that requires no further steps. Otherwise, find the class C_2 data point with the lowest preference value using the w vector from this solution, shown in Figure 4-11(a), and create a boundary through this class C_2 data point, perpendicular to the $P(\theta) + \delta$ line (Figure 4-11(a)). Then, divide C_1 data points into two subsets A^1 and A^2 depending on whether they fall above or below this line.

Now one can follow the previously detailed steps for iteratively moving data points from one subset of class C_1 data points to the other. In this case, it will not be known from which subset to remove data points until an attempt is made to solve LP_1 and LP_2. Once it is determined which LP is unsolvable, one can move data points from the corresponding subset into the other as before until either both LPs are solvable, or neither is. Some experiments on this approach are detailed in Chapter 5.

4.3 Removing non-binding constraints

In this section, we discuss methods for removing many of the non-binding constraints for two-class and multi-class problems.

4.3.1 2-class problems

Although the original problem formulation (4-3) uses every data point in a data set as the basis for a constraint, many of these constraints may not affect the solution. It is only useful to create constraints from data points which could affect the values of $P(\theta^{min})$ or (θ^{max}). As shown in Section 4.1.3, constraints based on data points from either class may affect the values of both $P(\theta^{min})$ and $P(\theta^{max})$.

Without solving formulation (4-3), the data points that are affecting the θ^{min} or θ^{max} boundaries for the optimal solution are not known. However, one can be sure of a subset of points in each class which cannot affect these values. This subset of data points will have lower (for class C_1) or higher (for class C_2) preference values in comparison with at least one other data point in that class for any feasible solution. Thus, no boundary which includes a data point from this subset can correspond to a feasible solution, as it would exclude at least one class C_1 data point. This subset of points falls inside a convex region of class C_1 points, where the outside corners of this region are made up of the potentially relevant data points which make the points inside the region non-binding.

These outer data points can be called "maximal" data points (for class C_1), as they will have preference values equal to the $P(\theta^{min}) - \varepsilon$ for some solution to formulation (4-3) that is feasible with respect to class C_1 data points. This will be the highest preference value possible for a class C_1 data point according to this solution. Class C_2 data points may make all potential solutions that correspond to a particular maximal data point infeasible, but this may not be obvious without first running the linear program.

Similarly, the outer corners of the lower region formed by class C_2 data points can be called "minimal" data points, and will have preference values equal to $P(\theta^{max})$ for some solution to formulation (4-3) that is feasible with respect to class C_2 data points. All inner class C_2 data points will have a higher preference value than the minimal data points for class C_2.

Figure 4-12 (a) shows the convex hulls for class C_1 and class C_2, with the inner data points A_2, A_4, and B_2. There is no boundary that can be drawn involving these data points without excluding the other data points from the region, and so it is not useful to create constraints based on them.

In two dimensions, finding these corner data points is straightforward. First, the data points in class C_1 are sorted in descending order on a single dimension, and then they can be tested in groups of three. In each group of three, if the "middle" data point (along the sorted dimension) falls below a line formed by the other two data points, it is not a corner data point, and can be discarded. This ensures that data points behind the frontiers of our region will not be used for constraints. Afterwards, each data point is added, one at a time, and the test is the repeated on the previous set of three data points in this manner until all data points in the class have been considered. To find the minimal data points in class C_2, we can follow a similar process, but sort the class C_2 data points in ascending order, and look for data points above each line to discard, rather than below it.

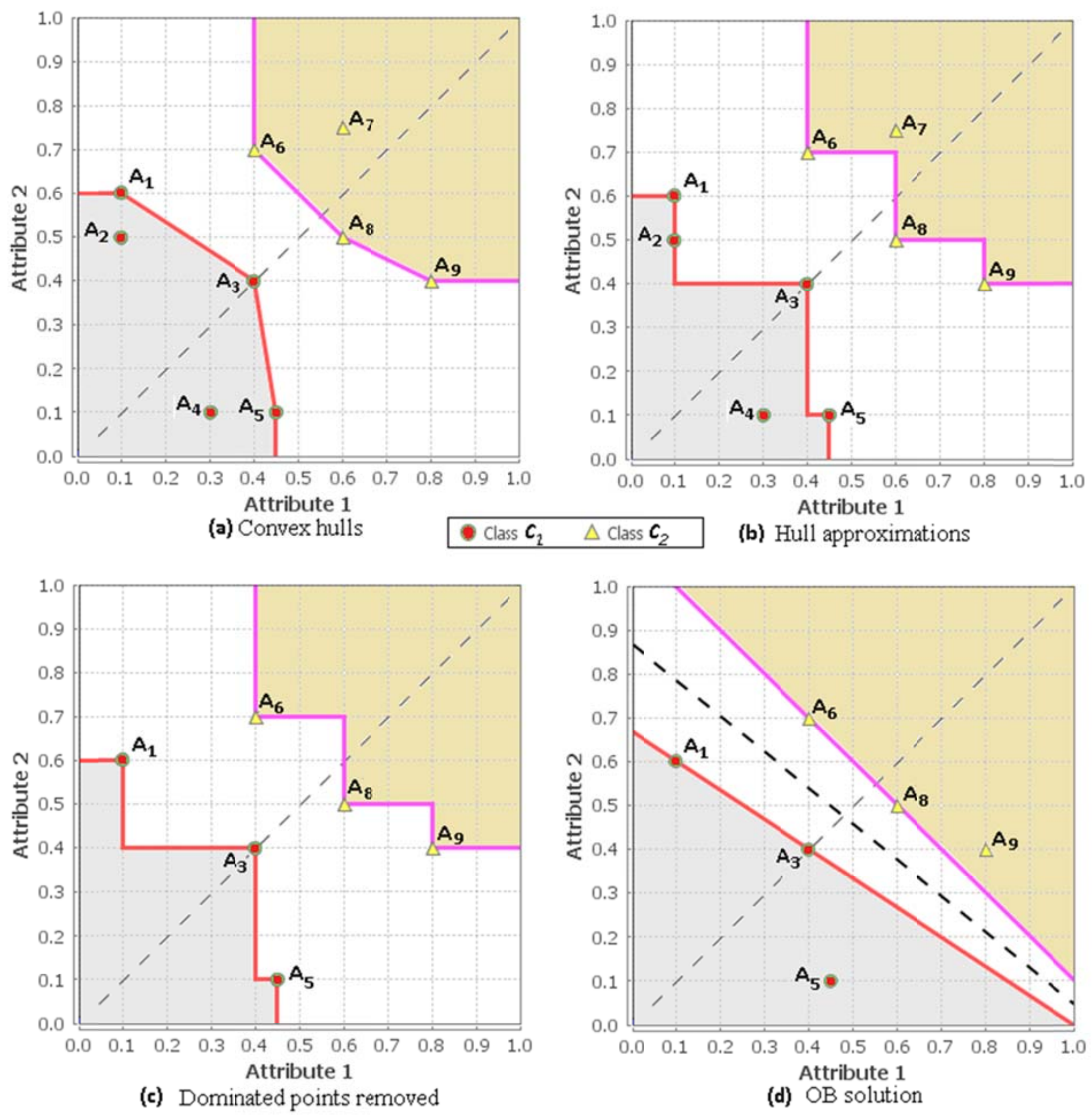


Figure 4-12. Removing unnecessary constraints.

In higher dimensions, a heuristic search would be necessary to determine which data points to select for a similar testing strategy. Instead, one can employ a simpler approach to approximate the convex hull, using the Pareto Frontier. The *Pareto Frontier* is the set of alternatives which are *Pareto Optimal* (Lotov & Miettinen, 2008). In this context, they are data points which are not dominated by any other data point in the class. A data point *dominates* another if it has a superior value in one attribute and is superior or equal in value in all other attributes. "Superior" is relative to the desired objective. When one is approximating the convex hull for class C_1, the "maximal" points which are Pareto Optimal have a greater value in one attribute and a greater or equal in value in all other attributes in comparison with the data points in class C_1 which they dominate. Conversely, for the lower convex hull of class C_2, a "minimal" Pareto Optimal data point has a lower value in one attribute and is lower or equal in value in all other attributes in comparison with the data points that it dominates.

Figure 4-12(b) shows the space which is dominated by the Pareto Optimal corner points. Figure 4-12(c) shows the classes with the dominated data points removed, and Figure 4-12(d) shows the formulation (4-3) solution for this example. This approximation approach can also be applied without modification to the convex region in the 2-plane separable problem.

4.3.2 Multi-class problems

For problems with more than two classes, a similar approach is used. For the 1st and the *n*th class, we follow the procedure detailed for 2-class problems, for class C_1 and class C_2, respectively. For the 2nd to *n*-1th classes in an *n*-class problem, one must consider both the minimal and maximal Pareto Optimal data points to determine which inner data points are unnecessary. This leaves us with only the data points necessary to approximate the convex hulls.

Figure 4-13(a) shows the convex hulls for class C_1, class C_2, and class C_3. Figure 4-13(b)&(c) show the approximated hulls before and after removing unnecessary data points. Figure 4-13(d) shows the solution for this data set after removing data points.

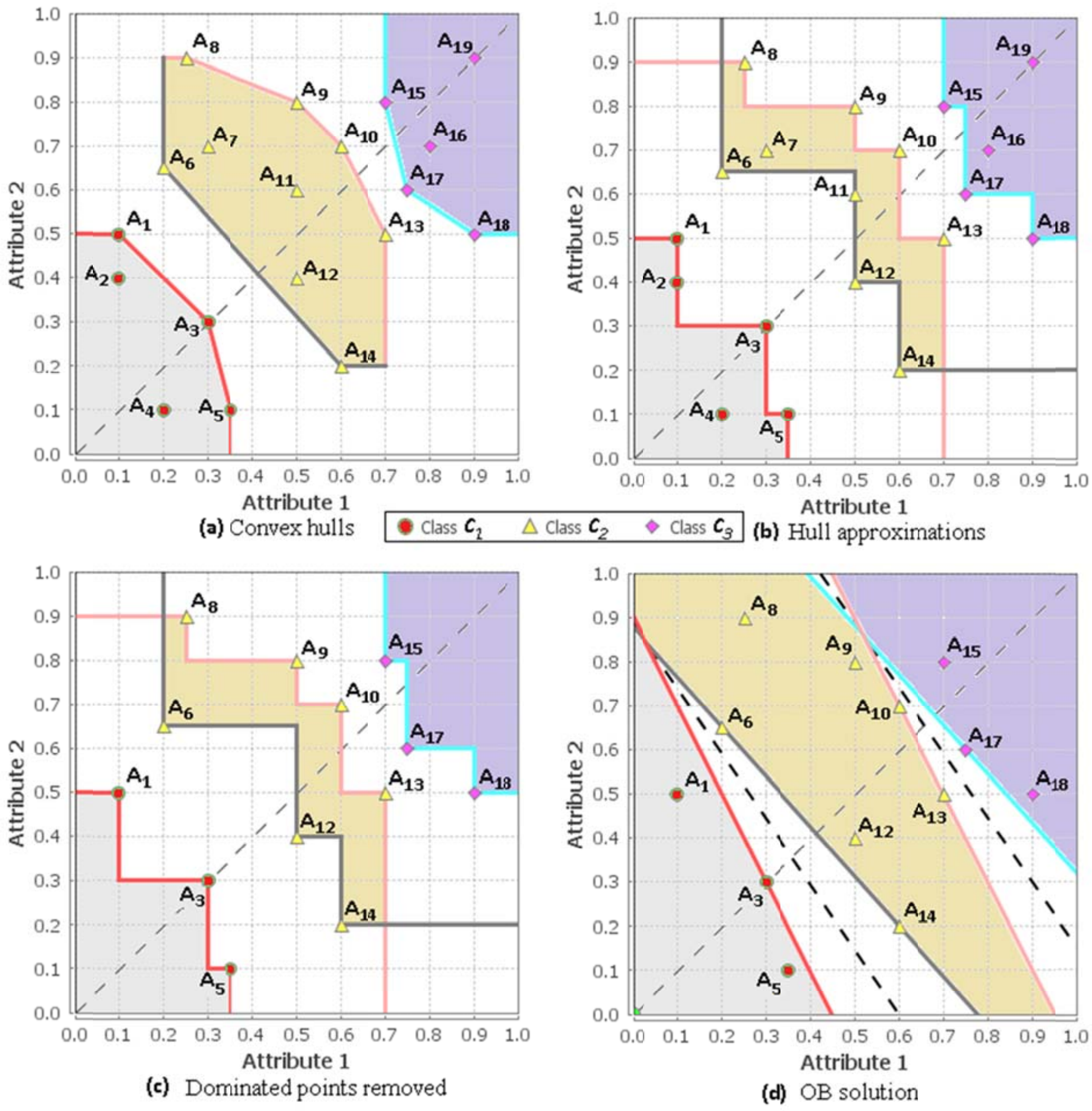


Figure 4-13. Removing unnecessary data points in a multi-class problem.

4.4 Non-separable problems

If one attempts to find a solution using a formulation like (4-3) or (4-5) for a particular data set, and it turns out to be infeasible, then either that data set is not linearly separable, or in the multi-class case, it is not linearly separable when using the same weight vector for all class boundaries. In these cases, at least one of our assumptions is not satisfied. For example, there may be some amount of error or inconsistency in the way the weights were applied.

It is also possible that some undocumented influence modified the class value after it was initially calculated. Although the decision maker could still be using a linear model such as the WSM, the class value may be different than in the perfectly consistent (separable) case due to this inconsistency. As a result, even if one knows the true values of the weights, some data points may still be classified incorrectly. However, it is possible that the proposed Ordinal Boundary approach is still appropriate and with some modifications, it will offer an improvement over the traditional approaches.

In this situation, one can no longer make an estimation based on the extreme cases, θ^{min} and θ^{max}, as there is no linear boundary that totally separates the data. Instead, the goal is to find a boundary θ with the least amount of variation in the weight vector. This variation can be accounted for by relaxing the constraints. To this end, some additional weight variables, *w1* and *w2,* are added for each training alternative. The *w1* variable represents a positive change to the weight vector while *w2* represents a negative change.

The goal of the new objective function is to reduce the total amount of variation from the boundary as much as possible. Thus we minimize the sum of the variations represented by *w1* and *w2*. Another possible objective would be to minimize the largest variation. However, there may be a case where a single alternative requires a large value in the *w1* or *w2* variables for a feasible solution, and such an objective function could result in a solution that has larger than necessary variations on other data points. Brief experiments were conducted to compare this alternative objective function in Section 5.3.5.

The formulation for a 2-class problem is as follows:

Objective function: Minimize $\sum_{i=1}^{m}\sum_{j=1}^{n}(w1_{ij} + w2_{ij})$ (4-8)

$\forall A_i \in \text{class } C_1 \mid \sum_{j=1}^{n} x_{ij}\left(w_j + w1_{ij} - w2_{ij}\right) < P(\theta)$

$\forall A_k \in \text{class } C_2 \mid \sum_{j=1}^{n} x_{kj}\,(w_j + w1_{kj} - w2_{kj}) \geq P(\theta)$

$\sum_{j=1}^{n} w_j = 1$

$P(\theta) \geq 0$

and $w_j, w1_{ij}, w2_{ij}, w1_{kj}, w2_{kj} \geq 0$, for all *i, j, k*.

Because the *w1* or *w2* variables do not have an upper bound, formulation (4-8) will always be feasible. To classify a new alternative (when a formulation for the separable problem is infeasible), the LP model detailed in formulation (4-8) is run, and the weight vector *w* from the solution is used to find the preference value of the new data point using the WSM as before.

The classification function is: If $P(A_i) < P(\theta)$ $\Rightarrow$ Class C_1,

otherwise $\Rightarrow$ Class C_2.

An extension to n classes is done by using $(n-1)$ boundaries, one between each ordered pair of classes, in the same manner as described for separable functions. For 3-class problems, with classes C_1, C_2, and C_3, the formulation is as follows:

Objective function: Minimize $\sum_{i=1}^{m}\sum_{j=1}^{n}(w1_{ij} + w2_{ij})$ (4-9)

$\forall A_i \in \text{class } C_1 \mid \sum_{j=1}^{n} x_{ij}\left(w_j + w1_{ij} - w2_{ij}\right) < P(\theta_1)$

$\forall A_k \in \text{class } C_2 \mid \sum_{j=1}^{n} x_{kj}\,(w_j + w1_{kj} - w2_{kj}) \geq P(\theta_1)$

$\forall A_k \in \text{class } C_2 \mid \sum_{j=1}^{n} x_{kj}\,(w_j + w1_{kj} - w2_{kj}) < P(\theta_2)$

$\forall A_p \in \text{class } C_3 \mid \sum_{j=1}^{n} x_{pj}\,(w_j + w1_{pj} - w2_{pj}) \geq P(\theta_2)$

$\sum_{j=1}^{n} w_j = 1$

$P(\theta^1), \qquad P(\theta^2) \geq 0$

and $w_j, w1_{ij}, w2_{ij}, w1_{kj}, w2_{kj}, w1_{pj}, w2_{pj} \geq 0$, for all *i, j, k, p*.

As in formulation (4-8), the *w1* or *w2* variables do not have an upper bound, thus formulation (4-9) will also always have a feasible solution. As a result, if the data set being considered was not created using consistently applied weights, the solution may have a value for $P(\theta_1)$ that is larger than $P(\theta_2)$. While it is possible to add further constraints to eliminate this possibility, the resulting classifier would likely be a poor choice for the data set. It is therefore advisable to examine the solution values for this type of anomaly.

The classification function is:

If $P(A_i) < P(\theta_1)$ $\Rightarrow$ Class C_1,

If $P(A_i) \geq P(\theta_2)$ $\Rightarrow$ Class C_3,

otherwise $\Rightarrow$ Class C_2.

Cases with more classes can be developed in a similar manner. In Chapter 5 we show some experimental results on a variety of data sets.

5. EXPERIMENTAL RESULTS

To provide empirical support for the proposed methods, a variety of experiments were performed. These experiments were done using data that was generated, collected, and gathered from published sources. The proposed methods were compared against similar alternative methods and traditional textbook approaches. In this chapter, the details of the data and methods used in the experiments are discussed and the results of those experiments are analyzed.

5.1 Data sets

In order to test our method against traditional classification methods, we have generated a number of data sets, which vary in both the number of criteria and the number of classes. We have also tested against several published data sets and collected a new data set.

5.1.1 Generated separable data

The first type of test data is generated in a consistent manner, i.e., the classes represented in these data sets are separable by linear boundaries that share a single weight vector. The attribute values are created by first randomly generating a set of alternatives, each with a vector of attribute values, using a uniform distribution. Then a single weight vector is randomly generated, and the preference values for each alternative are calculated based on the WSM. The alternatives are sorted by preference value, and threshold values for dividing the alternatives into classes are selected by the "Unsupervised Discretize" filter, a tool within the WEKA Data Mining suite (Witten & Frank, 2000). This tool attempts to divide the alternatives into groups with an equal number of examples. After generating the preference values and labeling the classes, the weights, the preference values, and the class threshold values are discarded. This leaves a list of the attribute values and their class labels for each alternative, similar to the example data shown in Table 2-4. The problem dimensions range from simple (2 attributes) to more complex than published problem data sets (100 attributes). A set of 2-class problems and a set of 10-class problems were tested. Each set contains 100 examples per class.

5.1.2 Generated non-separable data

The next set of generated test data is data that violates our assumptions by adding variation to the attribute weights in order to simulate some amount of error. First the alternatives are randomly generated as before, using a uniform distribution for the attribute values. Then a vector of weights is randomly generated using a uniform distribution. Before computing the class values, a modifier is randomly generated for each alternative, from a normal distribution with a specified variance. This modifier is then multiplied by the class value. This allows for the simulation of some inconsistency in the resulting class value for each alternative in a controllable manner.

5.1.3 Published data

Real world data sets have been difficult to procure. Much of the relevant work in the MADM field has been done on small example sets or proprietary data. In the Data Mining field, these ordered methods are sometimes tested on data sets originally processed for testing regression classifiers (Frank & Hall, 2001), and a few sets of "truly ordinal" real world data (Huhn & Hullermeier, 2009). Although the results for all relevant data sets are presented here, unfortunately, very few of these data sets are appropriate. The majority of them are not the result of a decision-making process, or a similar process that contributes to their ordinal properties in a consistent way, which is what the proposed Ordinal Boundary method attempts to capitalize on.

The three data sets from the “regression” group (Torgo, 2001) are labeled “Bank8FM”, ”Delta Elevators”, and “Fried Artificial”. “Bank8FM” is made of customer decisions between different banks, depending on queue length, distance, and levels of patience, which are represented in 8 numeric attributes. The class value is a rejection rate, which has been discretized into 10 categories. ”Delta Elevators” is data based on predicting the change in an F16 jet airplane’s elevator position using 6 numeric attributes, and the class value has been discretized into 10 categories. “Fried Artificial” is a synthetic data set designed for testing regression trees, with 10 numeric attributes, and the class value has been discretized into 10 categories. Because “Bank8FM”, ”Delta Elevators”, and “Fried Artificial” were large and complex enough to require significant computation time (on both the proposed methods and the traditional methods), they were randomly subsampled for testing.

The four decision related data sets are labeled “ERA”, “ESL”, “LEV” and “SWD”. “ERA” and “ESL” are both data on selecting a candidate employee’s fitness for work based on 4 numeric attributes, with a class value ranging from 1 to 9. “ERA” was collected during an MBA course experiment, while “ESL” was collected from expert recruiters. “LEV” is a collection of lecturer evaluations taken at the end of MBA courses, with 4 numeric attributes and a class value ranging from 0 to 4. “SWD” is a set of assessments from social workers on the risk facing children if not removed from their families at home. These assessments are represented by 10 numeric attributes and a class value ranging from 1 to 5. All of these data sets contain decision-making data based on ordinal attributes, with ordinal class values. However, the alternatives come from many different decision makers, so a highly consistent set of weights is not expected.

To address the lack of published decision-making data, we have collected a data set from the interviewing department of Lofton Staffing Services, a temporary-hire staffing agency in Baton Rouge, Louisiana. This data set has 7 ordinal attributes to describe an interviewee, and an ordinal class value that represents the opinion of the interviewer on the overall fitness for hiring. The staffing data was collected from one interviewer over a time period of two and a half months and covers 116 applicants. While testing, 10-fold cross validation (Tan, et al., 2006) was used on all data sets in order to divide them into training and testing data.

5.2 Traditional methods

To benchmark the proposed Ordinal Boundary method, some of the more widely used Data Mining classification techniques were chosen. All of these methods have the capability to handle both binary and multi-class classification problems. The implementations of the methods used are those included in WEKA 3.7 (Witten & Frank, 2000). A list of these methods follows.

Logistic Regression (LogReg): This is a Logistic Regression classifier which uses LogitBoost for fitting linear logistic models (Landwehr, et al., 2005).

Support Vector Machine (SVM): This is a Support Vector Machine classifier which uses an implementation of John Platt's Sequential Minimal Optimization algorithm for training (Platt, 1998; Keerthi, et al., 2001). Multi-class problems are solved using pair-wise classification. This SVM is used with a linear kernel and a complexity parameter of 1.0. The Pairwise Coupling extension (Wu, et al., 2004) to this method was also tested, but initial results on a large sampling of our generated data sets showed no significant improvement, and in several cases had slightly lower accuracy.

Naïve Bayes (NaïveB): This is a Naïve Bayes classifier which uses estimator classes. This classifier analyzes the training data in order to choose numeric estimator precision values (John & Langley, 1995).

JRIP: This classifier is an implementation of Repeated Incremental Pruning to Produce Error Reduction (RIPPER). This is a propositional rule learner proposed by William W. Cohen as an optimized version of IREP (Cohen, 1995). JRIP was run with 3 folds and 2 optimization runs.

J48: This classifier uses a basic pruned C4.5 decision tree (Quinlan, 1993). J48 was run with 3 folds and a confidence factor of 0.25.

Some of these methods were examined in earlier papers in regard to leveraging order (Huhn & Hullermeier, 2009) (Frank, 2001). The C4.5 decision tree was a base method that improved under different meta-classification schemes which effectively restricted its flexibility when applied to ordered data, and so it showed an improvement under these schemes. The linear SVM and LogReg methods showed little or no improvement, because they are methods that are already well suited to model ordered, linearly separable data (Huhn & Hullermeier, 2009). It was for these reasons they were chosen as benchmark methods.

The meta-classification schemes for ordered class data introduced in (Frank, 2001) and (Frank, 2004) were also tested on a large sampling of these data sets. The results are detailed in Section 5.3.7.

5.3 Results

In the following sections we detail the results of tests on separable and non-separable problems, from generated and published sources.

5.3.1 Results for separable generated data

The following result sets are plotted as dimensionality vs. accuracy for each problem size. The proposed Ordinal Boundary method is labeled "OB." In the 2-class case, the results show there is not a large advantage in using the proposed Ordinal Boundary method, which is expected, as there is no additional information to leverage against other linear models. However, it is interesting to note that as the dimensionality gets larger on the 2-class data sets, the Bayesian, rule-based, and decision tree classifiers start to suffer greatly in accuracy, while the linear models only decline slightly.

Table 5-1. Accuracy results for 2-class problems.

No. of Attributes	LogReg	SVM	NaïveB	JRIP	J48	OB
2	99.0%	97.5%	96.5%	96.5%	98.0%	99.0%
3	98.5%	97.0%	94.0%	90.0%	90.0%	98.5%
5	97.0%	94.0%	90.0%	78.0%	83.5%	98.5%
10	97.0%	96.0%	91.0%	76.5%	77.0%	97.0%
50	84.5%	79.5%	79.5%	55.5%	61.5%	85.5%
100	77.0%	82.0%	72.0%	52.5%	60.0%	82.5%

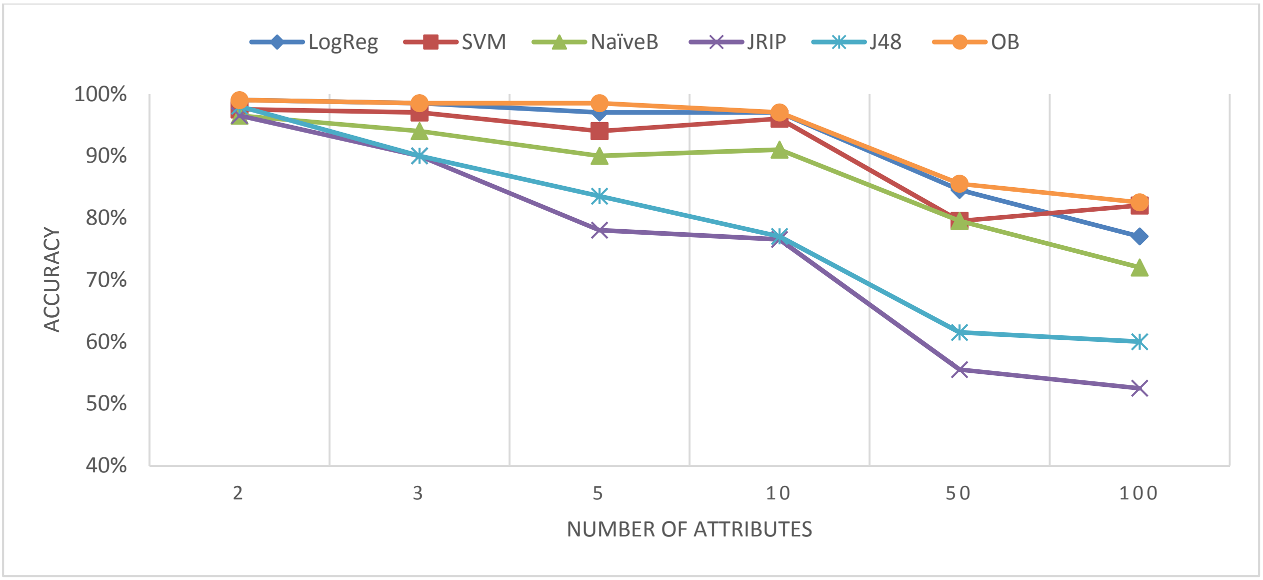


Figure 5-1. Accuracy results for 2-class problems.

To more closely examine the role that using only one weight vector to define the boundaries between classes plays in the proposed Ordinal Boundary method, a "non-consistent" baseline variation (labeled as "Base") was also tested. This method uses formulation (4-1) to create a binary classifier for each pair of classes. For classification, the baseline method uses the One-vs-One multi-class classification scheme detailed in Chapter 3. This baseline version of the method estimates each boundary separately, and has a separate weight vector for each boundary. With independent weight vectors, there is no restriction to ensure that the boundaries will be parallel. On 2-class problems (which have only a single class boundary), the baseline method is identical to the proposed Ordinal Boundary method.

Table 5-2. Accuracy results for 3-class problems.

No. of Attributes	LogReg	SVM	NaïveB	JRIP	J48	OB	Base
2	97.7%	92.3%	86.0%	91.3%	91.7%	99.7%	99.7%
3	97.0%	95.0%	84.7%	79.7%	82.7%	98.3%	97.7%
5	96.3%	93.7%	82.3%	72.3%	72.3%	99.0%	98.7%
10	94.0%	91.7%	78.0%	55.3%	59.0%	97.3%	97.0%
50	68.0%	81.0%	64.0%	45.7%	42.0%	89.3%	77.0%
100	57.0%	67.0%	50.3%	34.7%	32.3%	81.7%	68.3%

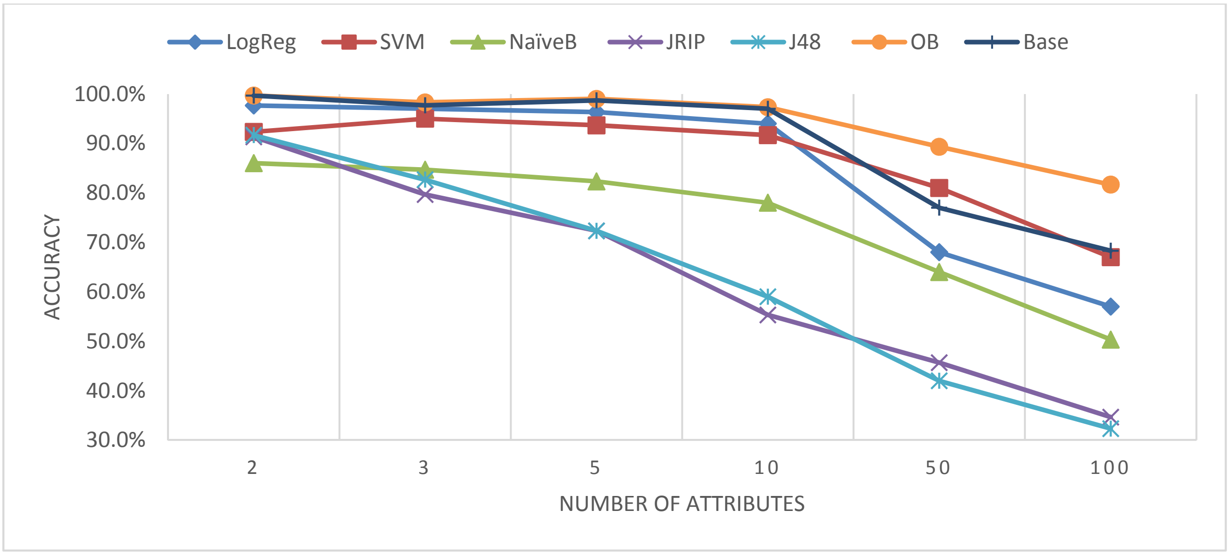


Figure 5-2. Accuracy results for 3-class problems.

The advantage offered by the proposed Ordinal Boundary method becomes clearer as the number of classes in the data sets becomes larger. Although the 3-class data set has only one additional boundary, on the 100-attribute data set, the proposed Ordinal Boundary method has a 13.4% increase in performance over the baseline version (81.7% for the Ordinal Boundary method versus 68.3% for the baseline version) and a 14.7% increase in performance over the highest performing textbook method, the SVM (67% accuracy.)

Table 5-3. Accuracy results for 5-class problems.

No. of Attributes	LogReg	SVM	NaïveB	JRIP	J48	OB	Base
2	96.6%	90.4%	64.0%	82.4%	87.0%	99.2%	99.2%
3	95.2%	91.0%	65.0%	53.2%	74.4%	98.4%	97.6%
5	92.4%	84.8%	59.6%	39.8%	56.0%	98.4%	97.2%
10	91.2%	86.2%	56.6%	34.4%	42.0%	98.6%	93.2%
50	61.8%	70.4%	45.6%	23.8%	22.4%	94.0%	76.4%
100	43.2%	52.8%	41.4%	21.6%	23.2%	85.8%	52.4%

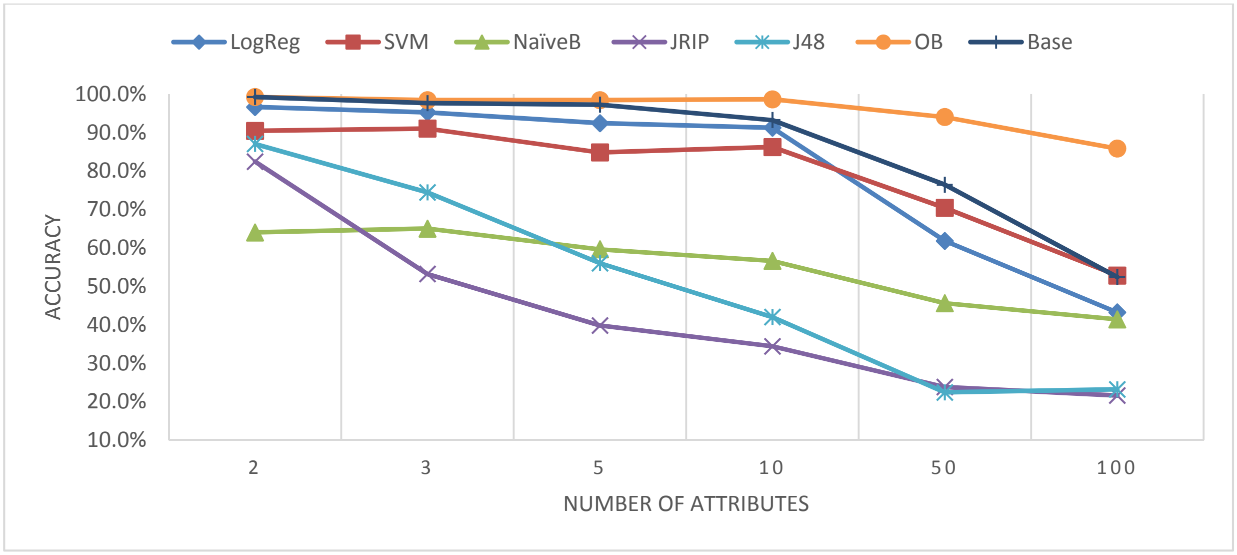


Figure 5-3. Accuracy results for 5-class problems.

Of the 5-class data, the 50-attribute data set is sufficiently complex to see a large improvement in accuracy for the proposed Ordinal Boundary method, which has an accuracy of 94%. This is an improvement of 17.6% over the baseline version (76.4%) and 23.6% over the SVM, which was the highest performing textbook method on this data set at 70.4%. The 5-class, 100-attribute data set shows an even larger improvement for the proposed Ordinal Boundary method. With an accuracy of 85.8%, it has an advantage of

33.4% over the baseline version (52.4%) and 33% over the Support Vector Machine classifier, which had an accuracy of 52.8%.

Table 5-4. Accuracy results for separable 10-class problems.

No. of Attributes	LogReg	SVM	NaïveB	JRIP	J48	OB	Base
2	95.2%	76.1%	45.6%	73.5%	83.2%	98.9%	98.3%
3	95.9%	72.7%	49.7%	54.2%	70.3%	99.2%	98.2%
5	91.2%	67.2%	42.5%	23.9%	41.9%	98.8%	97.2%
10	80.0%	60.9%	34.4%	18.2%	21.2%	98.6%	93.3%
50	42.7%	48.4%	24.8%	12.4%	12.6%	96.5%	64.6%
100	23.3%	36.9%	20.1%	10.0%	9.6%	92.3%	34.4%

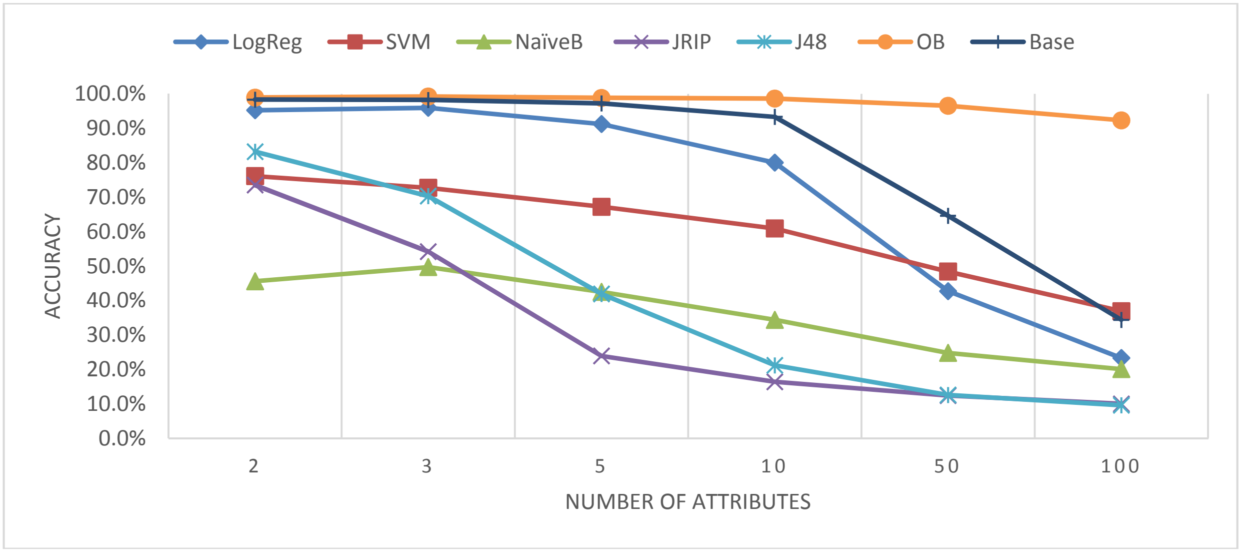


Figure 5-4. Accuracy results for separable 10-class problems.

On the 10-class data sets, the proposed Ordinal Boundary method shows a large performance advantage on even the less complex problems (e.g. the 5-attribute data set), over both the baseline version and the highest scoring textbook method. Performance for every other classifier is significantly lower on the 10-class problem set in comparison with the 2-class set, but for the proposed Ordinal Boundary method, these additional classes represent additional information about the true location of the separating boundaries between classes. Even on the 2-attribute variant, it has a significant lead over the highest competitor, the logistic regression classifier. On the 100-attribute variant, the proposed Ordinal Boundary method shows an improvement of 55.4% above the highest performing competition on the 10-class data set (92.3% for the Ordinal Boundary classifier versus 36.9% for the Support Vector Machine classifier). The baseline version has a slightly lower accuracy 34.4% than the Support Vector Machine classifier.

Table 5-5. Accuracy results for 20-class problems.

No. of Attributes	LogReg	SVM	NaïveB	JRIP	J48	OB	Base
2	93.2%	49.8%	34.8%	72.1%	81.2%	98.8%	98.6%
3	88.7%	45.3%	31.9%	30.4%	58.8%	99.0%	97.4%
5	68.5%	42.0%	24.3%	10.5%	22.5%	98.8%	96.1%
10	56.6%	40.5%	21.1%	10.3%	16.1%	98.8%	94.1%
50	26.5%	31.8%	13.3%	6.0%	6.9%	97.8%	58.3%
100	16.8%	24.0%	12.5%	5.4%	5.9%	95.4%	22.5%

While the baseline version of the classifier still competes very favorably with the textbook classification methods, it demonstrates the significant advantage gained from using a consistent set of weights. This becomes more obvious as the dimensionality is increased, but it is most obvious on data sets with a large number of class boundaries. On the 20-class, 100-attribute data set, the baseline version performs comparably to the Support Vector Machine classifier (24.0% accuracy) at 22.5% accuracy, while the proposed Ordinal Boundary classifier accuracy declines only slightly from the less complex 20-class data sets to 95.4% on the 100-attribute data set.

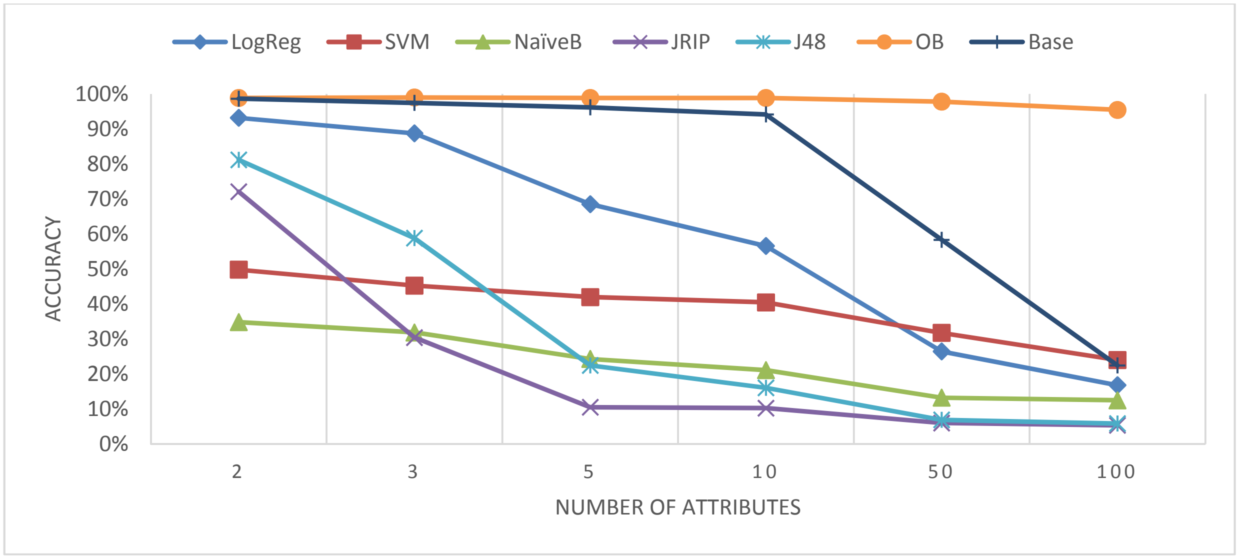


Figure 5-5. Accuracy results for 20-class problems.

It is also telling that the accuracy for the proposed Ordinal Boundary method increases with additional class boundaries. This is demonstrated in a comparison of the most complex (100-attribute) data sets from each of the groups of the generated separable data. In contrast, the accuracy of the baseline version and textbook methods steadily declines as the number of classes increases.

Table 5-6. Accuracy results for 100-attribute problems.

No. of Classes	LogReg	SVM	NaïveB	JRIP	J48	OB	Base
2	93.2%	49.8%	34.8%	72.1%	81.2%	98.8%	98.6%
3	88.7%	45.3%	31.9%	30.4%	58.8%	99.0%	97.4%
5	68.5%	42.0%	24.3%	10.5%	22.5%	98.8%	96.1%
10	56.6%	40.5%	21.1%	10.3%	16.1%	98.8%	94.1%
20	26.5%	31.8%	13.3%	6.0%	6.9%	97.8%	58.3%

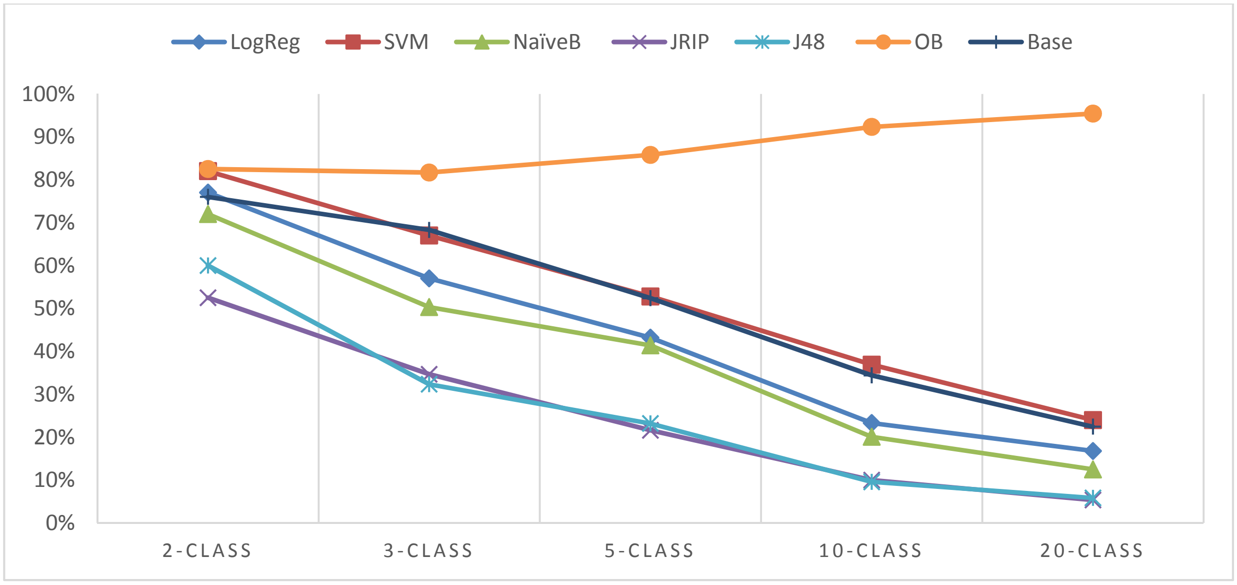


Figure 5-6. Accuracy results for 100-attribute problems.

5.3.2 Results for alternative objective functions

Empirical testing was done to investigate the alternative objective functions for the separable problem, as discussed in Section 4.1.6. The approach detailed in formulation (4-6) is labeled "MaxMin" and the approach detailed in formulation (4-7) is labeled "Parallel," while the approach used in formulation (4-5) is labeled "OB." Tests were done on many of the data sets, but there was no clear advantage to using a different objective function. This is shown in the result set in Table 5-7, created from tests on 50-attribute data sets.

Table 5-7. Accuracy results for 50-attribute problems.

No. of Classes	MaxMin	Parallel	OB
2	86.5%	86.0%	85.5%
3	89.0%	90.0%	89.3%
5	93.2%	92.0%	94.0%
10	95.4%	95.3%	96.5%
20	97.9%	97.7%	97.8%

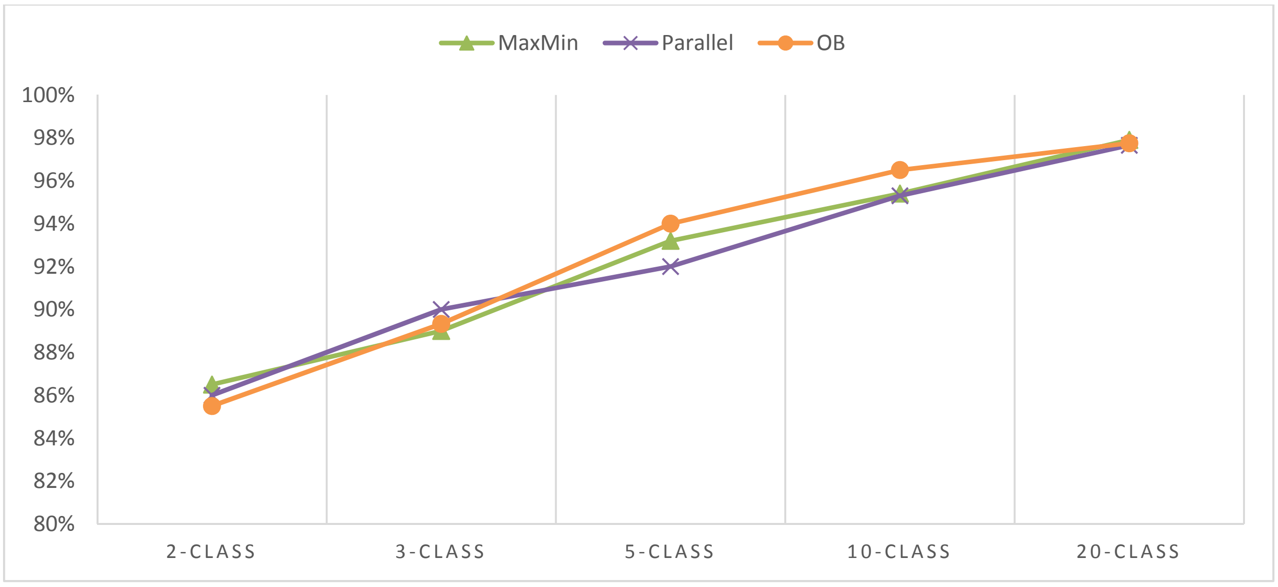


Figure 5-7. Accuracy results for 50-attribute problems.

5.3.3 Results for non-separable generated data

Three non-separable data sets were generated to examine how accuracy was affected by differences in the amount of noise added, the number of attributes, and the number of classes. The first set of non-separable data was chosen to investigate what happens to the accuracy rate as we increase the amount of error that is added into the weights. The data sets were generated using 10 classes, 10 attributes, and noise levels from 0.1% to 35%.

Table 5-8. Accuracy results for 10-class 10-attribute problems.

Noise Level	LogReg	SVM	NaïveB	JRIP	J48	OB	Base
0.1%	78.5%	64.0%	38.4%	20.6%	23.6%	97.3%	93.9%
0.2%	80.7%	60.9%	34.1%	18.4%	22.4%	96.9%	94.1%
0.4%	78.7%	63.2%	35.7%	19.0%	28.8%	94.5%	92.3%
0.6%	78.6%	62.2%	36.8%	19.2%	24.5%	93.1%	89.3%
0.8%	79.5%	63.3%	35.2%	19.1%	27.1%	91.1%	88.1%
1%	73.5%	59.2%	33.5%	19.1%	24.7%	87.1%	85.5%

Table 5-8 continued. Accuracy results for 10-class 10-attribute problems.

Noise Level	LogReg	SVM	NaïveB	JRIP	J48	OB	Base
3%	57.7%	52.1%	32.6%	16.7%	22.5%	65.5%	63.6%
5%	46.1%	44.6%	34.0%	19.1%	21.6%	50.0%	49.1%
10%	34.5%	34.7%	28.0%	17.5%	21.6%	38.2%	35.5%
15%	26.2%	25.8%	25.9%	14.3%	16.5%	27.8%	25.6%
20%	23.8%	24.2%	20.6%	11.8%	15.4%	23.6%	20.8%
25%	22.8%	23.1%	21.1%	12.5%	17.1%	22.0%	21.9%
30%	16.8%	17.9%	16.3%	10.4%	13.7%	20.3%	19.7%
35%	15.3%	15.8%	15.3%	10.0%	12.2%	17.8%	16.6%

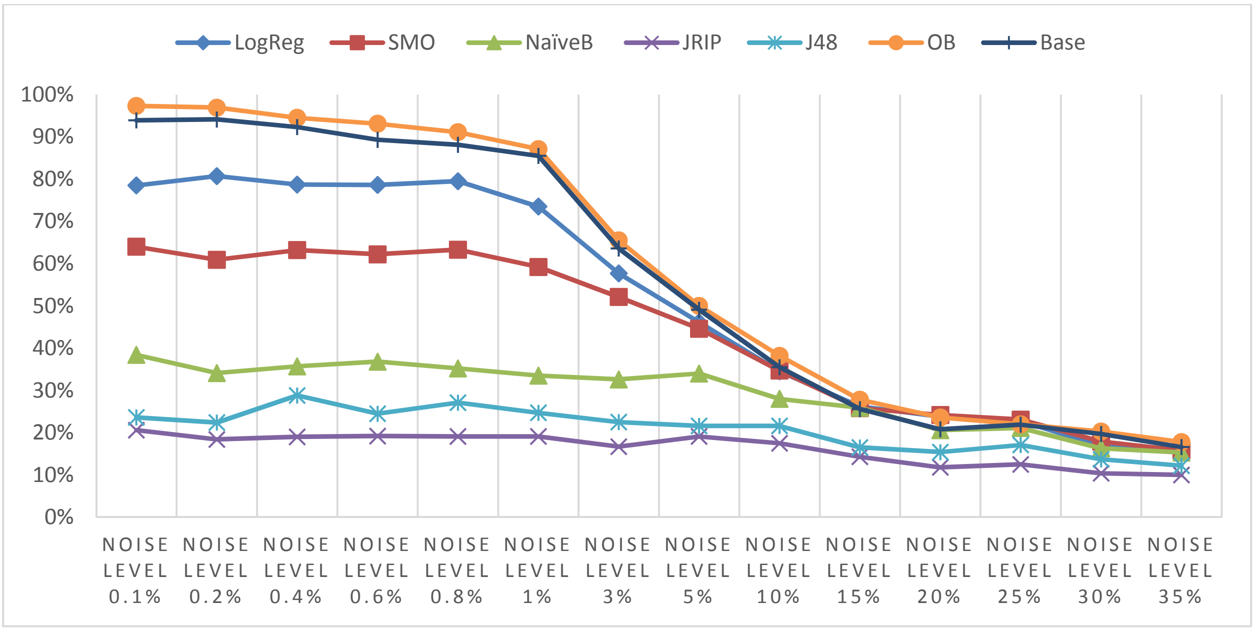


Figure 5-8. Accuracy results for 10-class 10-attribute problems.

The results show a steady decrease in accuracy for our method as the amount of random noise is increased. The Support Vector Machine and Logistic Regression methods show a similar decrease in performance, while the Baysian, rule-based and tree-based classifiers show poor performance throughout, with a large drop only from 10% to 15% noise levels.

The next set of non-separable data was generated to examine the effect on accuracy that increasing the number of attributes on 10-class problems with a 3% noise level. These parameters were chosen in light of the previous test set's results (shown in Table 5-8), which showed that 10-class problems with 3% noise and 10 attributes will have a significant decrease in performance in comparison with 10-class problems with 10 attributes and no noise. However, at 3% the proposed Ordinal Boundary method still maintains a sizable lead over the highest performing competing method.

Table 5-9. Accuracy results for 10-class problems with 3% noise.

No. of Attributes	LogReg	SVM	NaïveB	JRIP	J48	OB	Base
2	85.4%	70.3%	46.4%	71.1%	79.5%	86.0%	85.1%
3	78.6%	64.6%	42.6%	33.4%	52.7%	78.1%	77.1%
5	71.1%	61.2%	39.3%	25.1%	35.5%	75.4%	73.4%
10	60.9%	55.5%	28.1%	19.7%	23.2%	69.2%	67.0%
20	45.9%	44.3%	28.5%	13.8%	16.3%	57.5%	53.2%

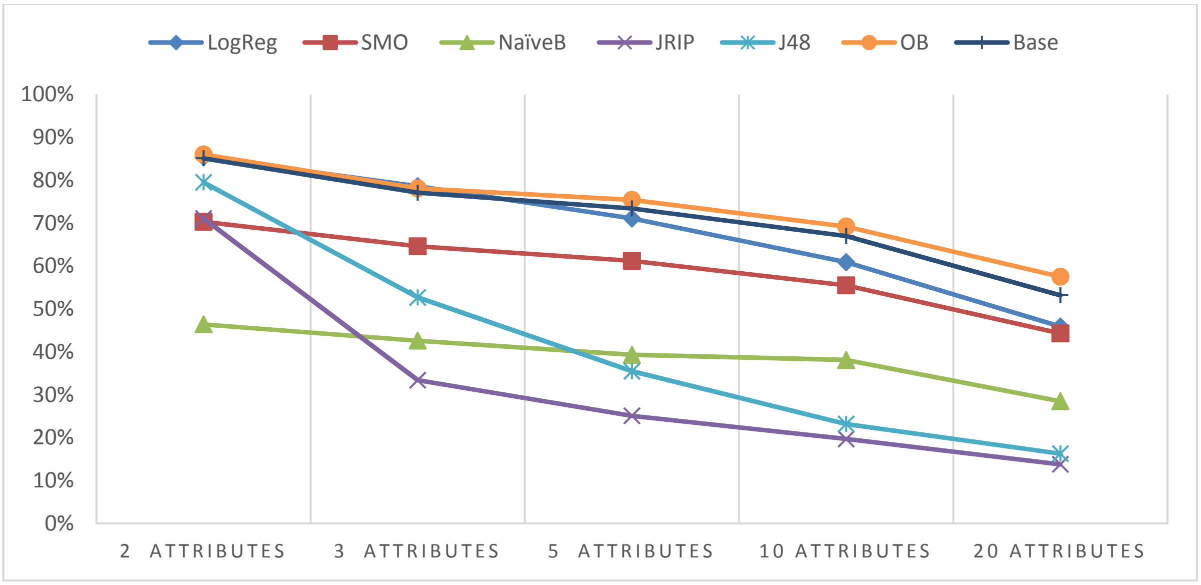


Figure 5-9. Accuracy results for 10-class problems with varying no. of attributes.

The results indicate that the number of attributes does have a significant effect on the performance of all tested methods on these data sets. As the number of attributes gets larger, accuracy drops. However, the proposed Ordinal Boundary method has lower decrease in performance out of all tested methods with the exception of the Bayesian classifier. However, the Bayesian classifier has a significantly worse performance throughout. The proposed Ordinal Boundary method also has the highest overall performance on each of these data sets. Of the two textbook approaches, the Logistic Regression classifier has the highest performance on the data sets with less attributes, but the performances of the Logistic Regression classifier and the Support Vector Machine classifier appear to converge as the number of attributes gets higher.

The next set of non-separable data was generated to examine the effect on accuracy that increasing the number of classes on a 10-attribute data set with a 3% noise level. These parameters were chosen in light of the previous test set's results (shown in Table 5-9). Those results showed a significant increase in performance for the proposed Ordinal Boundary method at 3% noise and 10 attributes over the Logistic Regression classifier, which was the highest performing of the competing methods on that data set.

Table 5-10. Accuracy results for multi-class 10-attribute problems with 3% noise.

No. of Classes	LogReg	SVM	NaïveB	JRIP	J48	OB	Base
2-class	91.5%	89.0%	84.5%	74.0%	69.5%	93.5%	93.5%
3-class	88.0%	87.3%	72.7%	54.0%	54.3%	89.7%	88.3%
5-class	79.2%	78.4%	55.2%	35.8%	44.0%	84.2%	83.4%
7-class	68.7%	62.7%	41.0%	24.4%	30.6%	74.0%	72.1%
10-class	61.3%	54.9%	32.6%	19.7%	23.3%	69.9%	67.0%
15-class	45.5%	40.3%	24.8%	13.3%	16.7%	53.3%	51.5%

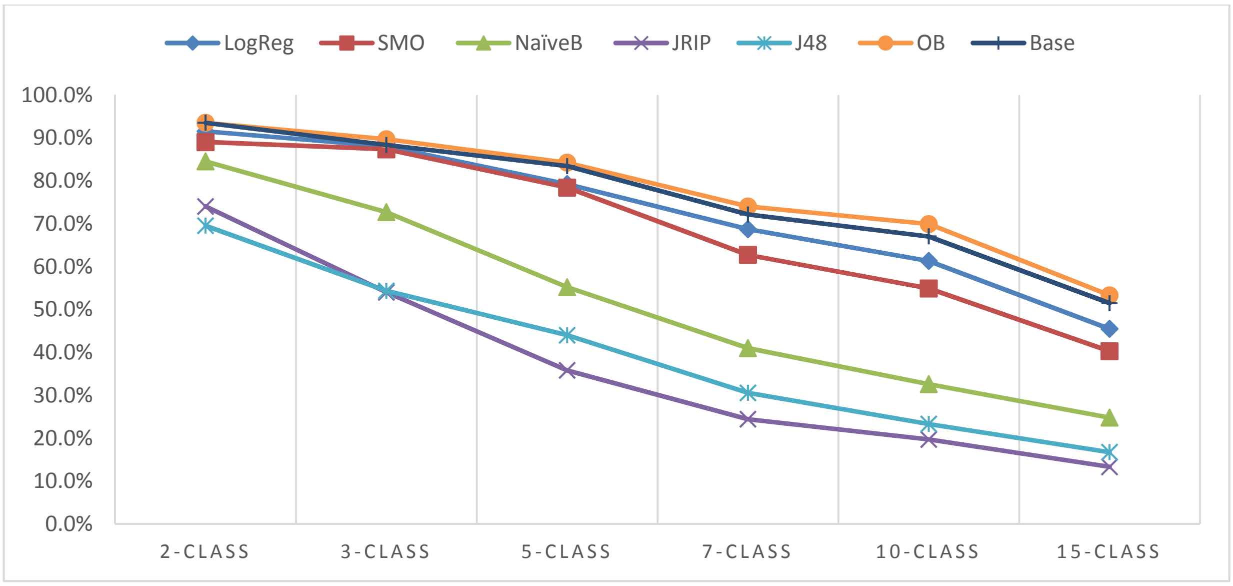


Figure 5-10. Accuracy results for multi-class problems with 10 attributes and 3% noise.

The results show a steady decrease in accuracy for all methods as the number of classes increases. However, the proposed Ordinal Boundary method has the highest accuracy in every case. While the performance of the Support Vector Machine and Logistic Regression classifiers is very close on the data sets with a lower number of classes, the proposed Ordinal Boundary method's advantage over them increases as the number of classes goes up.

5.3.4 Results for published data

Of the published data sets examined in (Huhn & Hullermeier, 2009) and (Frank, 2001), several appear to be relevant. Of the "regression" data sets, Bank-8FM, Delta-Elevators, and Fried-Artificial have an inherent consistency that gives the proposed Ordinal Boundary method a significant advantage over the baseline version. It should be noted that Bank-8FM and Fried-Artificial are synthetic data sets, and are presented here because they have been published and studied with these other real world data sets. Of the "truly ordinal" data sets, the ERA, ESL, LEV, and SWD data sets are all decision-making data. The decisions were made by several decision makers, so some inconsistency was expected.

Table 5-11. Details of published data sets.

Data set	No. of Attributes	No. of Classes	No. of Records
Bank-8FM	8	10	442
Delta-Elevators	6	10	500
Fried-Artificial	10	10	500
ERA	4	9	1,000
ESL	4	9	488
LEV	4	5	1,000
SWD	10	5	1,000
Staffing	7	5	116

Table 5-12. Accuracy results for published data sets.

Data set	LogReg	SVM	NaïveB	JRIP	J48	OB	Base
Bank-8FM	61.8%	29.0%	39.8%	37.3%	44.8%	64.3%	58.8%
Delta-Elevators	26.8%	27.2%	28.8%	17.0%	18.8%	28.8%	25.4%
Fried-Artificial	29.4%	28.6%	29.8%	19.8%	26.4%	34.0%	31.4%
ERA	29.1%	25.7%	26.0%	19.7%	27.5%	25.1%	26.2%
ESL	71.1%	61.9%	64.1%	65.0%	71.3%	70.1%	70.5%
LEV	59.4%	59.3%	56.8%	60.1%	61.1%	60.6%	58.9%
SWD	58.3%	58.7%	56.8%	57.0%	57.1%	56.4%	56.0%
Staffing	87.2%	88.0%	86.2%	76.1%	78.0%	92.2%	89.7%

Comparing the proposed Ordinal Boundary classifier accuracy to the baseline version shows whether there is enough internal consistency in the published data sets to leverage for an improvement in accuracy. In the "regression" data sets presented, the proposed Ordinal Boundary method shows a significant improvement over the baseline method, and also has the best performance of all methods. The classifiers made with the baseline method are the same as with the proposed Ordinal Boundary method on pairs of classes. The difference is that in multi-class cases, the baseline method uses the 1-vs-1 meta-classification scheme, while the proposed Ordinal Boundary method is restricted to considering only class boundaries that could be parallel. Thus, the increase in accuracy is compelling evidence that this restriction is responsible for the advantage the proposed Ordinal Boundary method has on these data sets.

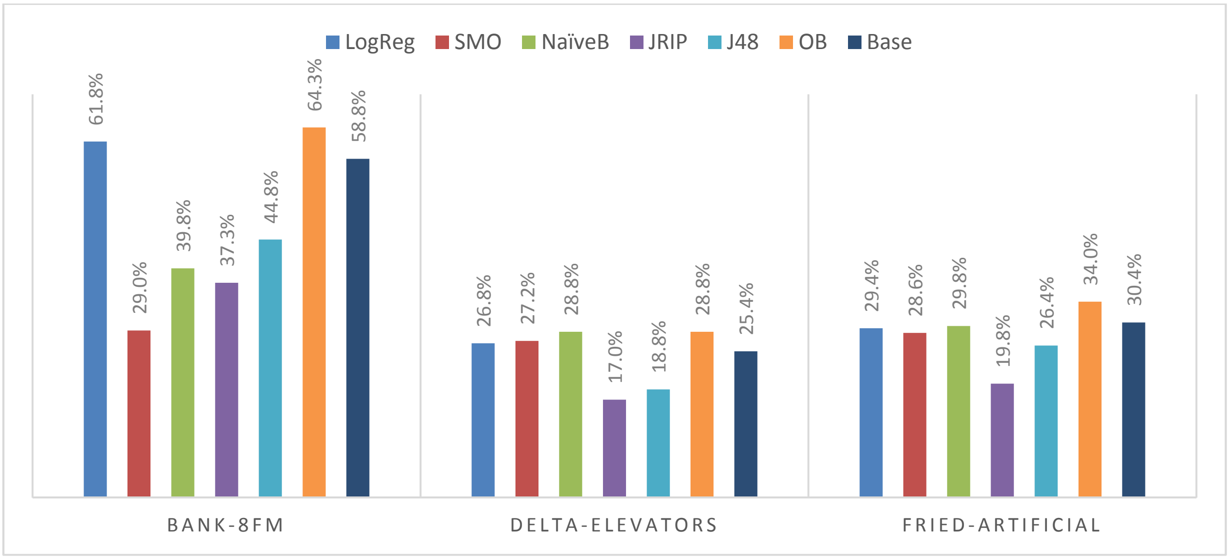


Figure 5-11. Accuracy results for “regression” data sets.

In the decision-making data, only results from the staffing set show a significant improvement when using the proposed Ordinal Boundary method over the baseline method. In the ERA, ESL, LEV, and SWD data sets, restricting the boundaries to a single weight vector does not hurt or help the classifier accuracy. This is likely due to the data being sourced from a number of decision makers, which leads to a less consistent decision-making process. However, the proposed Ordinal Boundary method remains competitive with the textbook algorithms.

On the staffing data, the proposed Ordinal Boundary, Logistic Regression, and Support Vector Machine classifiers all perform comparably, while the Bayesian, rule-based and tree-based methods were at a disadvantage. As performance is so high for these classifiers, this data set may not be complex enough to show as large of an advantage demonstrated in the generated data, however, the proposed Ordinal Boundary classifier significantly outperforms each of the textbook methods.

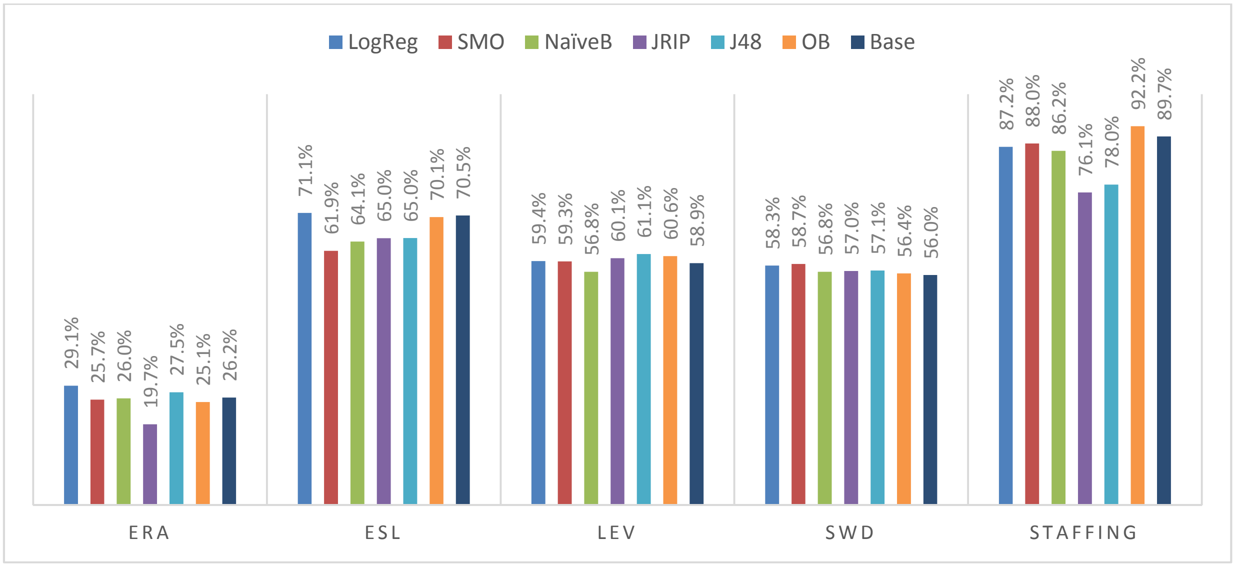


Figure 5-12. Accuracy results for published decision-making data sets.

5.3.5 Results for the non-separable alternative objective function

Tests were also conducted to investigate the alternative objective function for the non-separable problem, as discussed in Section 4.4. In this alternative objective function, we minimize the maximum variation (represented by *w1* or *w2*) instead of the sum of the variations. In most cases, including all generated test data and the published data sets "Bank8FM", "Delta Elevators", and "Fried Artificial", the alternative objective function resulted in identical classification accuracy as the proposed objective function. However, on some of the decision-making data sets, there was a difference in accuracy, as shown below. The alternative objective function (labeled as "MinMax") does not perform as well as the proposed objective function in these cases.

Table 5-12. Accuracy results for published data sets.

Data set	OB	MinMax
ERA	25.1%	24.3%
ESL	70.1%	69.1%
LEV	60.6%	57.7%
SWD	56.4%	54.4%

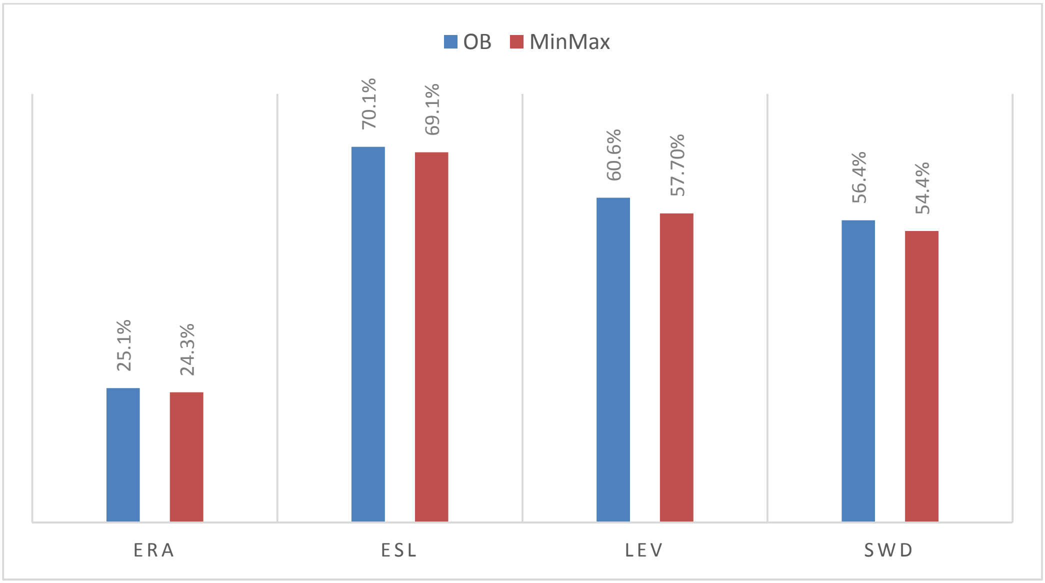


Figure 5-13. Accuracy results for published decision-making data sets.

5.3.6 Results for multiple decision maker data

A 2-class, 2-attribute data set, with 100 examples per class, was generated to test the multiple decision maker approach detailed in Section 4.2 for Case I, where the decision to classify a data point as class C_2 instead of class C_1 is the result of an agreement between both decision makers that the record should be classified so. The data set was created following the same procedure as detailed for the separable data, but with two weight vectors, one for each decision maker. The overall preference value for each alternative is then calculated according to each decision maker, and the lower of these two preference values is assigned to the alternative. This ensures that the alternative will not be assigned to class C_2 unless both of the preference values for that alternative are greater than the threshold value chosen for the class boundary.

Table 5-13. Accuracy results for published data sets.

Data set	LogReg	SVM	NaïveB	JRIP	J48	2DM
2-class 2-attribute	85.0%	84.0%	95.%	97.0%	97.0%	99.5%

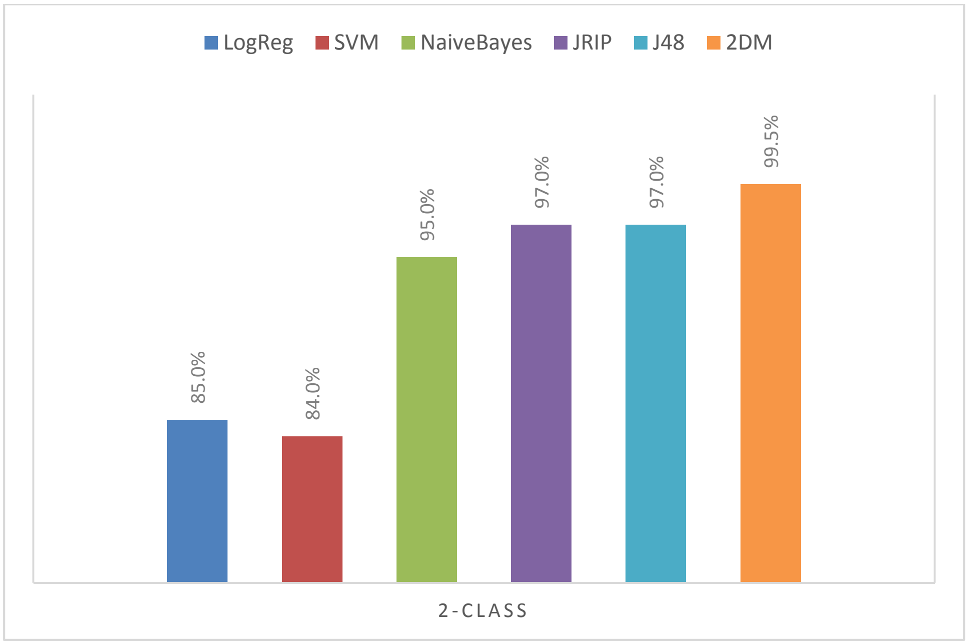


Figure 5-14. Accuracy results for multiple decision maker data.

The results show that this approach (labeled as “2DM”) offers a significant advantage in accuracy over the Logistic Regression and Support Vector Machine classifiers on this data set. This is to be expected, as these classifiers are attempting to model a single linear boundary instead of two. The more flexible classifiers, such as the rule-based and tree-based methods, had a very high performance. However, the approach from Section 4.2 was still able to offer a 2.5% increase in accuracy over even those classifiers (99.5% for 2DM vs 97.0% for JRIP and J48).

5.3.7 Results for ordinal meta-classification schemes

As discussed in Chapter 3, some “ordinal” meta-classification schemes have been proposed in an attempt to increase classifier accuracy on ordinal data sets. Of the methods described in Chapter 3, the approaches described in (Frank & Hall, 2001) and (Frank & Kramer, 2004) were examined, because they were presented with some promising results. These ordinal meta-classification schemes were tested using the Support Vector Machine, Logistic Regression, and baseline Ordinal Boundary classifiers as the underlying binary classifiers. The “Simple Ordinal” approach described in (Frank & Hall, 2001) is labeled “Frank” and the Nested Dichotomies approach described in (Frank & Kramer, 2004) is labeled “ND”. The One-vs-One approach is labeled “1vs1.” The implementations used were those included in the WEKA data mining tool suite (Witten & Frank, 2000).

A selection of the generated separable, generated non-separable, and published data sets were used to provide a comprehensive comparison between these ordinal approaches, the proposed Ordinal Boundary method and the traditional One-vs-One meta-classification approach. The One-vs-One approach is used as a benchmark because while it is not designed to be an ordinal approach, all pairs of classes can be considered trivially ordered.

Table 5-14. Accuracy results for meta-classification schemes with the baseline OB method as the underlying classifier.

Data set	Frank	ND	1vs1	OB
Separable 5-class 10-attribute	93.2%	53.2%	93.2%	98.6%
Separable 5-class 50-attribute	78.2%	45.0%	76.4%	94.0%
Separable 10-class 10-attribute	93.3%	30.1%	93.3%	98.6%
Separable 10-class 50-attribute	71.8%	31.3%	64.6%	96.5%
1% noise 10-class 10-attribute	85.6%	28.6%	85.5%	87.1%
3% noise 10-class 10-attribute	63.6%	27.6%	63.6%	65.5%
5% noise 10-class 10-attribute	49.4%	26.8%	49.1%	50.0%
10% noise 10-class 10-attribute	37.8%	21.8%	35.5%	38.2%
Bank8FM	60.2%	29.2%	58.8%	64.3%
DeltaElevators	30.6%	20.0%	25.4%	28.8%
Fried	32.2%	22.2%	34.0%	34.0%
ERA	28.4%	21.1%	26.2%	25.1%
ESL	70.3%	30.7%	70.5%	70.1%
LEV	60.2%	54.9%	58.9%	60.6%
SWD	56.6%	47.1%	56.0%	56.4%
Staffing	88.8%	44.0%	89.7%	92.2%

The first group of tests were performed using the “baseline” version (described in Section 5.3.1) of the proposed Ordinal Boundary method as the underlying binary classifier. On the generated data, the worst scoring meta-classification method by a large margin was the Nested Dichotomies meta-classifier. Frank’s Ordinal meta-classifier did offer some improvement over the One-vs-One meta-classifier on the more difficult data sets, such as the separable 5-class, 50-attribute and 10-class, 50-attribute data sets, and the non-separable 10% noise data set, where it had accuracy improvements of 1.8%, 7.2%, and 2.3%, respectively. However, the proposed Ordinal Boundary method outperformed all of the other methods on every generated data set.

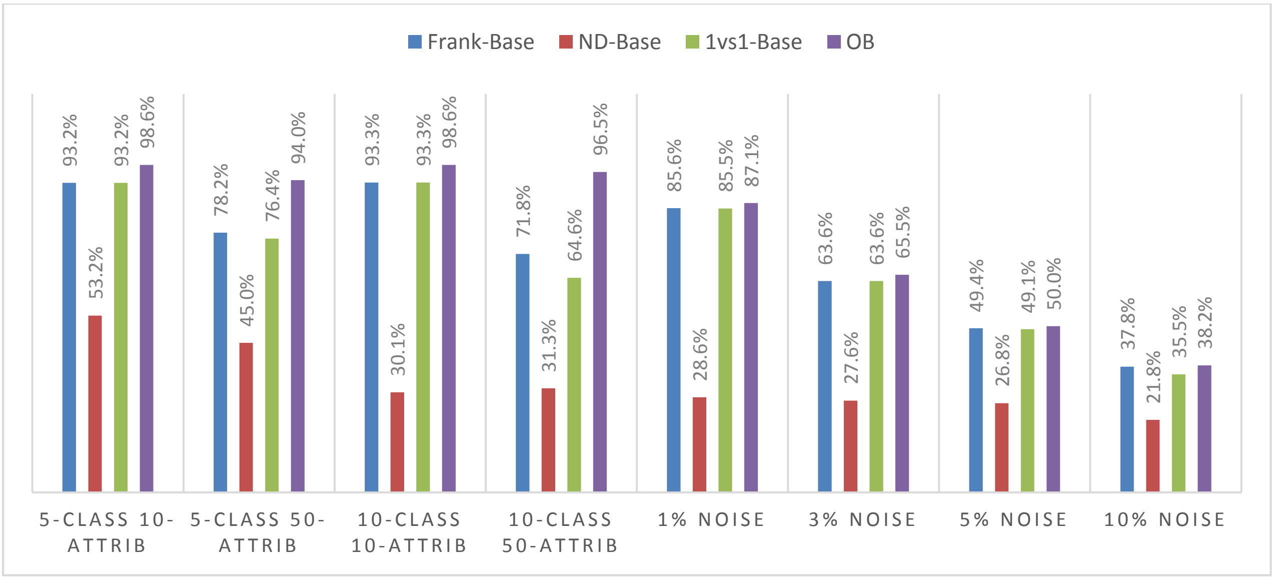


Figure 5-15. Accuracy results for meta-classification schemes with the baseline approach as the underlying classifier on generated data.

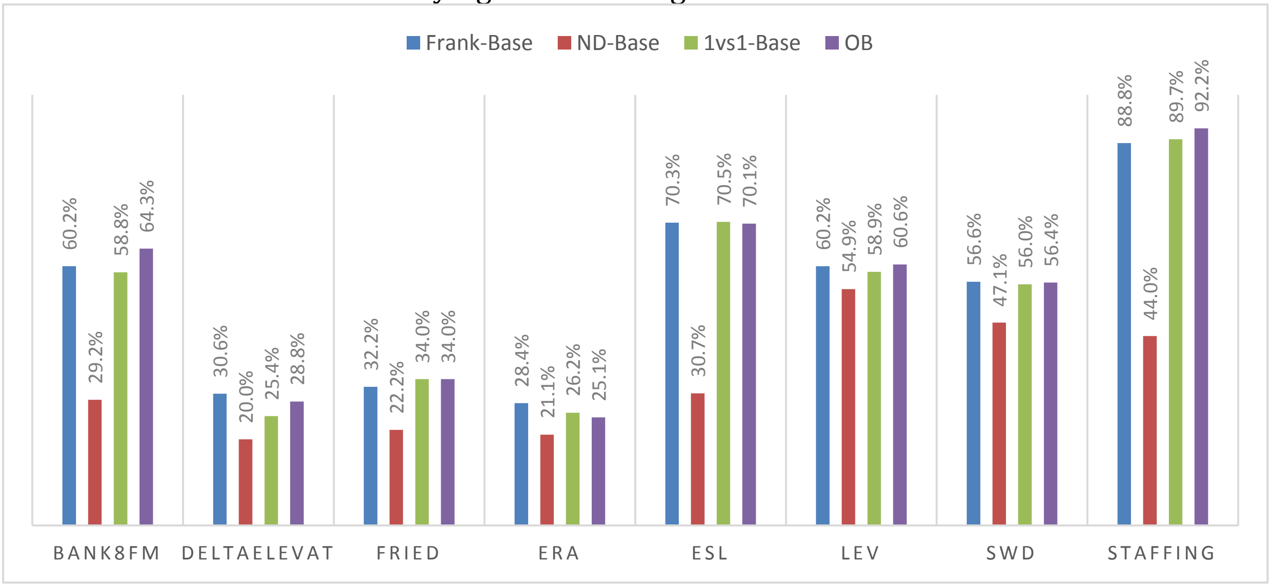


Figure 5-16. Accuracy results for meta-classification schemes with the baseline approach as the underlying classifier on published data.

On the published data, Frank's Ordinal meta-classifier (using the baseline approach as the underlying classifier) was more competitive. It had a 1.8% and 3.3% accuracy improvement on the "Delta Elevators" and "ERA" data sets, respectively, over the proposed Ordinal Boundary method. On the "ESL" data set, it had a marginal lead of 0.2% over the proposed Ordinal Boundary method, but lost to the One-vs-One baseline version by 0.2% as well. In the majority of cases where the proposed Ordinal Boundary method appeared to be appropriate (indicated by the fact that it outperformed the One-vs-One scheme in conjunction with the baseline binary classifier), it outperformed both Frank's Ordinal meta-classifier and the Nested Dichotomies meta-classifier.

Next, tests were performed using the less flexible underlying classifiers, specifically, the Logistic Regression classifier and the Support Vector Machine classifier. The native applications of these classifiers (with no meta-classification scheme) had the highest performance of the traditional methods on many of the earlier tests, in particular, the tests on separable generated data described in section 5.1.1. As with the tests which used the baseline version of the proposed Ordinal Boundary method as the underlying classifier, the Nested Dichotomies meta-classifier showed very poor performance throughout. It came in last place by a large margin on all generated data sets with both the Logistic Regression and Support Vector Machine underlying binary classifiers. This was also true for the majority of the published data (when the Nested Dichotomies meta-classifier was used with the Logistic Regression classifier), with the exception of the "LEV" and "SWD" data sets, where it was more competitive. However, it was not the top performer in either case. On the published data, when the Nested Dichotomies meta-classifier was used with the Support Vector Machine classifier, it was more evenly matched, but still had the worst performance of the four approaches.

In contrast, Frank's Ordinal meta-classifier used with the Logistic Regression classifier was competitive with the One-vs-One approach on the generated data sets. On the separable generated data, it had the largest advantage over the One-vs-One approach on the more complex (50-attribute) data sets, with a 5.4% and a 10% higher accuracy in the 5-class and 10-class data sets, respectively. On the non-separable generated data sets, it lost to the One-vs-One approach, but only by a small margin. However, in all of these cases, the proposed Ordinal Boundary method still had the highest accuracy, often by a large margin.

Table 5-15. Accuracy results for meta-classification schemes with the Logistic Regression classifier as the underlying classifier.

Data set	Frank	ND	1vs1	OB
Separable 5-class 10-attribute	92.8%	38.4%	91.0%	98.6%
Separable 5-class 50-attribute	67.6%	31.2%	62.2%	94.0%
Separable 10-class 10-attribute	82.3%	18.2%	86.1%	98.6%
Separable 10-class 50-attribute	53.1%	19.8%	43.1%	96.5%
1% noise 10-class 10-attribute	78.0%	16.4%	80.3%	87.1%
3% noise 10-class 10-attribute	59.1%	15.3%	60.5%	65.5%
5% noise 10-class 10-attribute	47.8%	12.4%	48.0%	50.0%
10% noise 10-class 10-attribute	35.7%	14.6%	35.2%	38.2%
Bank8FM	58.6%	20.1%	61.3%	64.3%
DeltaElevators	28.4%	16.8%	28.8%	28.8%
Fried	30.6%	11.8%	29.6%	34.0%
ERA	26.3%	15.0%	27.3%	25.1%
ESL	67.6%	28.9%	70.9%	70.1%
LEV	43.9%	50.6%	56.0%	60.6%
SWD	57.7%	57.7%	57.9%	56.4%
Staffing	84.7%	59.4%	88.9%	92.2%

On the published data, Frank's Ordinal meta-classifier used with the Logistic Regression classifier was competitive with the One-vs-One approach, but had a lower accuracy on all data sets except the "Fried Artificial" data set, where it had a 1% higher accuracy (30.6% accuracy for Frank's Ordinal meta-classifier vs. 29.6% accuracy for the One-vs-One meta-classifier.) Even in this case, however, the proposed Ordinal Boundary method had the best performance, with an accuracy of 34%. Also, in every case where the proposed Ordinal Boundary method had a higher accuracy than the One-vs-One meta-classifier, it also had the highest accuracy of all tested methods.

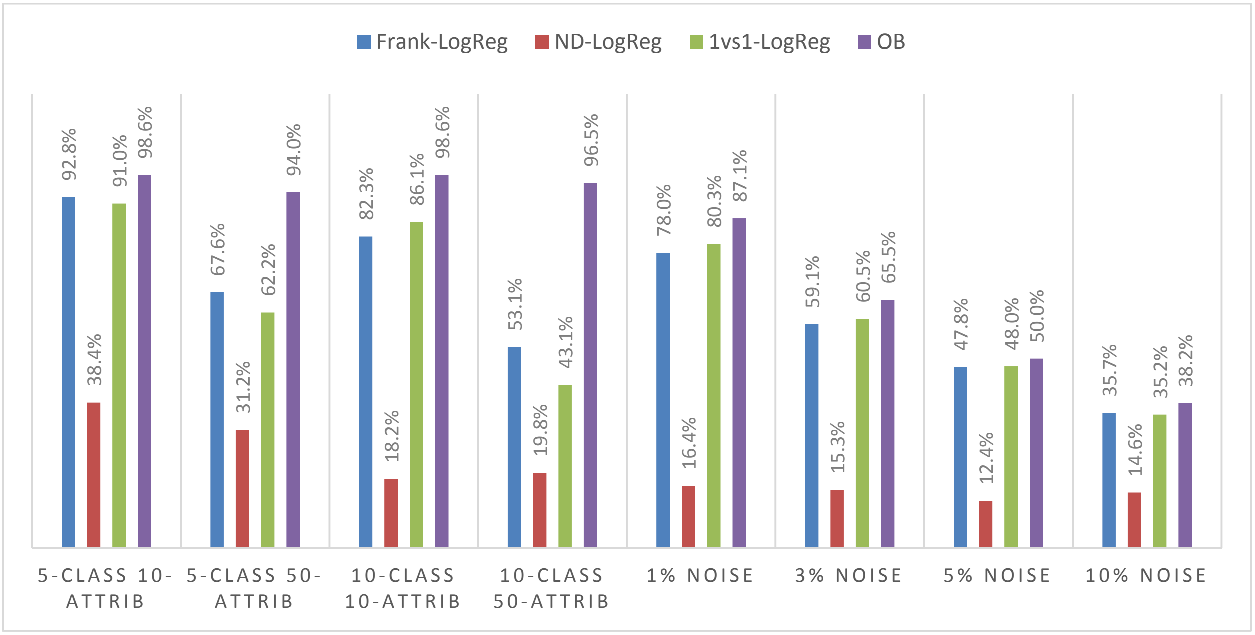


Figure 5-17. Accuracy results for meta-classification schemes with the Logistic Regression classifier as the underlying classifier on generated data.

When Frank's Ordinal meta-classifier was tested with the Support Vector Machine classifier, the advantage over the One-vs-One meta-classifier was significantly larger than with the Logistic Regression classifier. On the most complex (10-class, 50-attribute) separable generated data set, it had a 17.4% increase in accuracy over the One-vs-One meta-classifier (66.1% accuracy for Frank's Ordinal meta-classifier vs. 48.7% accuracy for the One-vs-One meta-classifier). Although the advantage decreased as the level of noise increased on the non-separable generated data, it still had a 2% increase in accuracy over the One-vs-One meta-classifier on the 10% noise data set (36.5% accuracy for Frank's Ordinal meta-classifier vs. 34.5% accuracy for the One-vs-One meta-classifier).

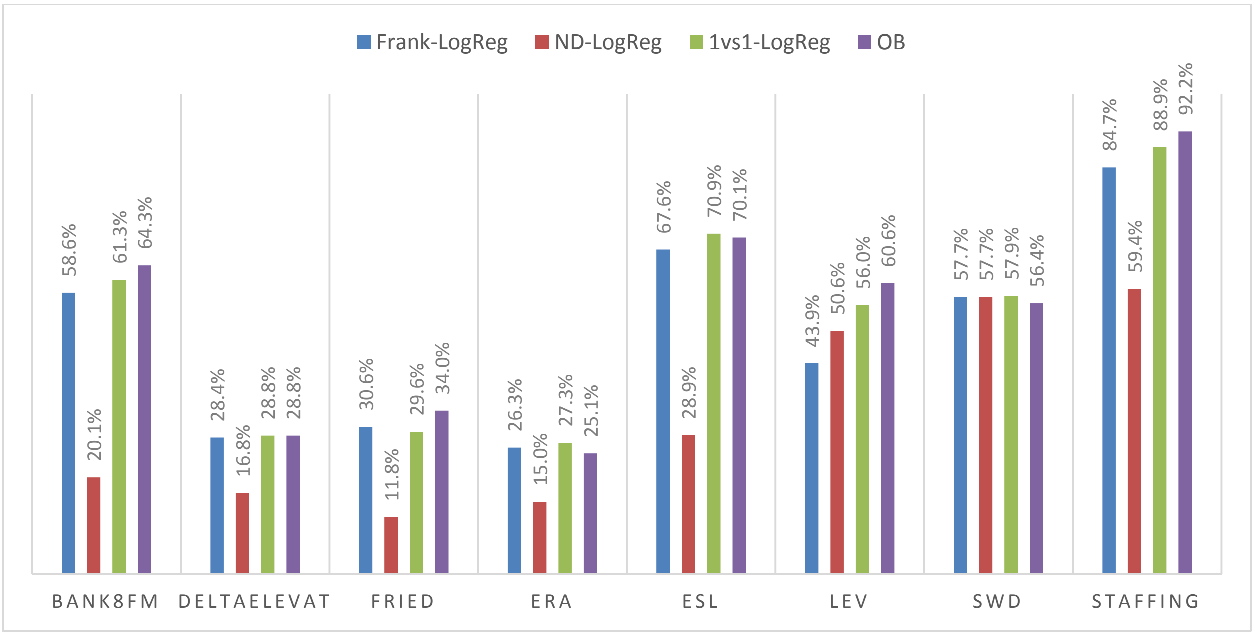


Figure 5-18. Accuracy results for meta-classification schemes with the Logistic Regression classifier as the underlying classifier on published data.

On the published data, Frank's Ordinal meta-classifier used with the Support Vector Machine classifier was competitive in all cases and significantly outperformed the One-vs-One meta-classifier on the "Bank8FM" and "ESL" data sets. However, the proposed Ordinal Boundary approach had a significantly better accuracy in both cases.

Overall, while the Nested Dichotomies meta-classification scheme did poorly in most cases, Frank's "Simple Ordinal" classifier showed a significant improvement in many of the tested ordinal data sets, both generated and published, over the One-vs-One meta-classification benchmark. In the overwhelming majority of these cases, the proposed Ordinal Boundary approach had the best performance, but it was clear that both methods gained an advantage from the ordinal nature of the data. Frank's meta-classification scheme has the ability to work with any binary classifier which gives probability outputs, and may therefore be useful for a larger range of ordinal data than the proposed Ordinal Boundary method.

Table 5-16. Accuracy results for meta-classification schemes with the Support Vector Machine classifier as the underlying classifier.

Data set	Frank	ND	1vs1	OB
Separable 5-class 10-attribute	86.6%	63.2%	86.2%	98.6%
Separable 5-class 50-attribute	73.4%	51.0%	70.4%	94.0%
Separable 10-class 10-attribute	80.6%	35.3%	61.5%	98.6%
Separable 10-class 50-attribute	66.1%	33.5%	48.7%	96.5%
1% noise 10-class 10-attribute	78.5%	39.4%	59.2%	87.1%
3% noise 10-class 10-attribute	59.5%	34.8%	52.1%	65.5%
5% noise 10-class 10-attribute	48.6%	33.5%	44.6%	50.0%
10% noise 10-class 10-attribute	36.5%	26.6%	34.5%	38.2%
Bank8FM	42.1%	23.7%	29.0%	64.3%
DeltaElevators	25.6%	21.8%	28.0%	28.8%
Fried	27.0%	25.6%	28.6%	34.0%
ERA	24.1%	22.4%	26.2%	25.1%
ESL	67.0%	42.6%	60.9%	70.1%
LEV	58.4%	55.1%	59.3%	60.6%
SWD	57.6%	56.3%	58.5%	56.4%
Staffing	89.0%	63.9%	89.9%	92.2%

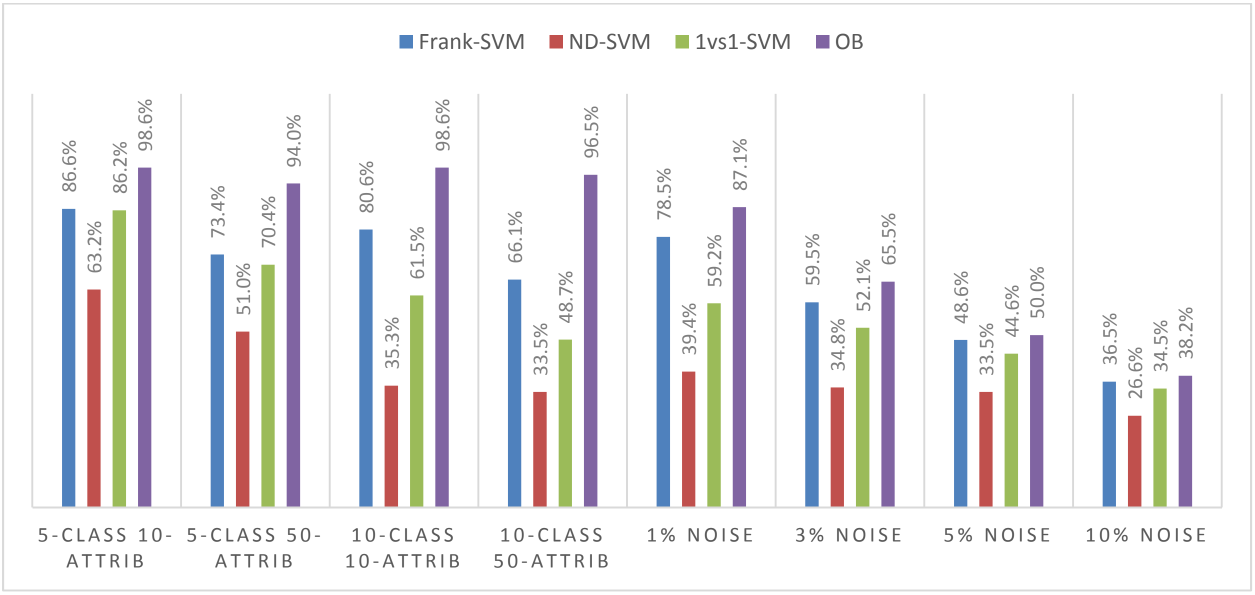


Figure 5-19. Accuracy results for meta-classification schemes with the Support Vector Machine classifier as the underlying classifier on generated data.

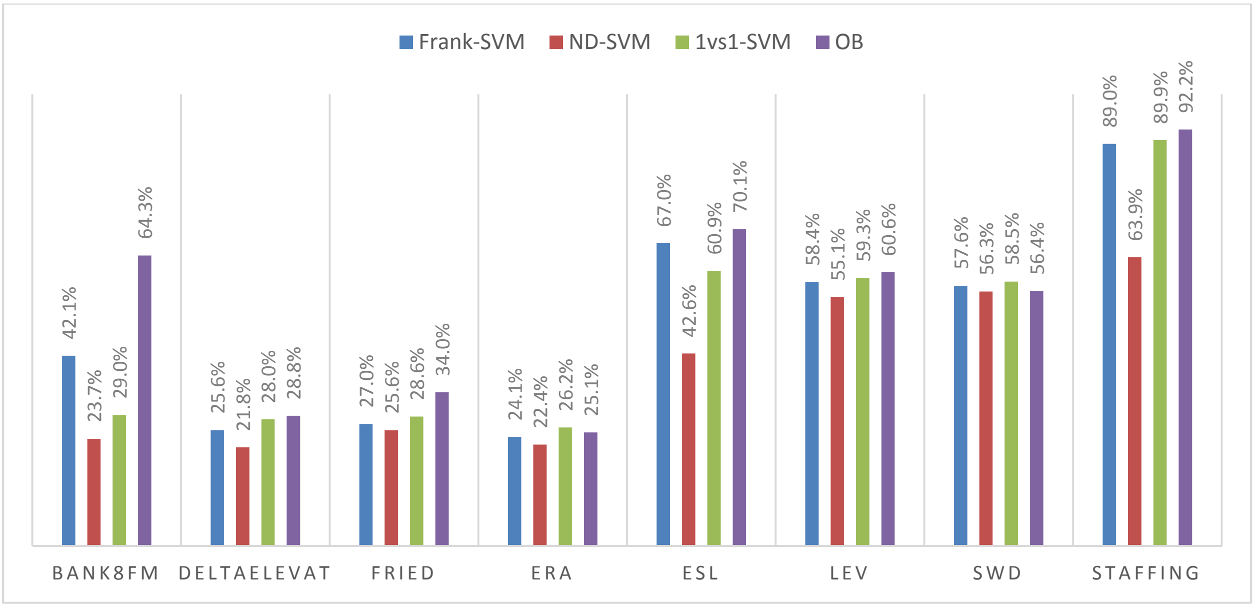


Figure 5-20. Accuracy results for meta-classification schemes with the Support Vector Machine classifier as the underlying classifier on published data.

6. SUMMARY AND CONCLUSIONS

We have developed a group of new and highly promising methods for use in several variations of classification based Multi-Attribute Decision-Making problems. These are classification problems where previous decisions based on a list of known alternatives are available for use in training models. Each alternative consists of a list of weighted attributes, which a decision maker combines to determine a preference value for that alternative. Classifications are made according to threshold values between ordered categories, which are pre-determined by the decision maker.

We have made several contributions in this area. We have reviewed the relevant work in both the Multi-Attribute Decision-Making and the Data Mining/Knowledge Discovery fields. Much of the MADM research on the classification problem is focused on defining more complex models for decision makers to employ in making the original classification and only limited discussion of using past decisions to infer a previous decision maker's preferences. The field of Data Mining/Knowledge Discovery has a large established base of general purpose approaches that fit this purpose. However, traditional Data Mining methods do not assume a data set of ordered classes, and make no attempt to capitalize on this additional information.

There have been several investigations to find empirical support for methods that use meta-classification schemes with various "ordered grouping" approaches. However, they do not thoroughly compare their methods against all traditional methods (Frank & Hall, 2001) or they do not show significant improvement over their base learners (Kim & Ahn, 2012).

Our methods are based on the assumption that the apparent order in a data set comes from a consistent classification of the training data, and that this consistency can be leveraged to provide a more accurate classification method than traditional approaches offer. Our proposed Ordinal Boundary method estimates boundaries between classes which can be characterized with a single weight vector. We introduced a method to find the full range of boundaries between class data using the largest range of the possible preference values which would be assigned to these data points in the typical approaches used in Multi-Attribute Decision-Making. We constructed a linear program to determine the lowest and highest possible boundaries in terms of these preference values, and used bisectors to equally divide the area of the uncertainty region between these boundaries in order to make a classification.

We developed an approach for investigating the case where two decision makers were operating under this assumption, but their decisions were used in tandem to determine the classification of the data. We also discussed the methods for reducing the amount of redundant constraints in our linear programs.

In order to leverage the knowledge about the ordinal properties of the classes that the training data is grouped in, one must consider what it was about these decisions that was consistent and therefore made them ordinal. In the case of a model such as the WSM, the same weight vector is applied to each data point to find its preference value, which is then used to determine its class value. The proposed Ordinal Boundary method takes advantage of this information by restricting the range of possible boundaries between classes to those which share the same weight vector. The minimum boundaries between all classes are restricted to the lowest preference value possible using a single weight vector (for those boundaries) and the maximum boundaries between all classes are similarly restricted to the highest preference value possible using a single weight vector.

This restriction eliminates an infeasible area of the uncertainty regions between multiple classes from consideration, while other Data Mining/Knowledge Discovery methods include this area as part of the possible region boundaries may lie in. Our extensive empirical results show that using this restriction can lead to significantly increased accuracy. The results also show that as the data sets grow more complicated (in terms of the number of attributes), our method suffers a smaller decrease in accuracy in comparison with traditional approaches. Furthermore, they show that a large increase in the number of classes in a problem set does not significantly reduce our accuracy rates, whereas with traditional approaches, the decrease in accuracy is large.

To show the impact that this consistency assumption makes, we compared an alternative baseline approach that does not use this restriction, and instead allows a separate weight vector for every boundary between classes. This method performed very suitably against traditional approaches, but suffered significantly as the number of classes in the test data sets was increased. On the data set with the largest number of attributes it performed nearly identically to the Support Vector Machine implementation that was tested, while the proposed Ordinal Boundary classifier maintained a large advantage in accuracy. We also compared some alternative objective functions. One of these used parallel minimum and maximum boundaries, similar to the parallel support vectors in a Support Vector Machine. Another alternative used a Max-Min approach on the difference in boundary preference values, instead of the maximum sum of these differences. However, experimental results showed no advantage in using these alternative approaches.

There are some problem sets which are not separable (using linear boundaries that share a single weight vector), but can still benefit from a classifier that assumes some consistency in the original decision-making. We modified our method to handle these cases by fitting a single weight vector overall, with the smallest amount of variation in that vector at each data point. Our empirical results have shown that this approach can be of benefit when there is modest noise that prevents the data from being linearly separable.

In addition to the proposed Ordinal Boundary method, we tested two other ordinal meta-classification schemes against the One-vs-One meta-classification scheme on a selection of the previous test data sets. Frank's "Simple Ordinal" classifier showed significant improvements over the One-vs-One approach on many of the tested data sets, while the Nested Dichotomies meta-classification scheme had poor performance throughout. However, in the majority of cases where the proposed Ordinal Boundary method outperformed the "baseline" version, and was therefore appropriate for a particular data set, it outperformed the other ordinal meta-classification schemes as well.

There are several areas of potential future research. More data collection for testing these various assumptions on real world data is desirable. Testing is currently difficult as most models proposed for Multi-Attribute Decision-Making show only theoretical examples and many of those with test data use proprietary information. In-depth investigation of these methods being used on a mix of cost and benefit attributes may lead to further improvements. Perhaps most interesting would be the modification of other Machine

Learning/Data Mining methods, so that they too try to take advantage of the assumption of consistent classification instead of using a meta-classification approach. The performance of a Support Vector Machine which followed this restriction in particular would be very telling. In the area of multiple decision makers, it would also be useful to see what heuristics can be developed to attack higher dimension data sets. Finally, perhaps some non-linear models used for classification outside the MADM field, which result in class data that appears ordinal, can be investigated to for other ways to leverage consistency between the estimated class borders.

BIBILIOGRAPHY

Araz, C., & Ozkarahan, I. (2007). Supplier Evaluation and Management System for Strategic Sourcing Based on a New Multicriteria Sorting Procedure. *International Journal of Production Economics,* 106(2), 585-606.

Ben-David, A., & Sterling, L. (2005). Generating Rules From Examples of Human Multiattribute Decision Making Should Be Simple. *Expert Systems with Applications,* 31(2), 390-396.

Bugera, V., Uryasev, S., & Zrazhevsky, G. (2008). Classification Using Optimization: Application to Credit Ratings of Bonds. In *Computational Methods in Financial Engineering: Essays in Honour of Manfred Gilli* (pp. 211-237). New York: Springer.

Cohen, W. (1995). Fast Effective Rule Induction. In *Twelfth International Conference on Machine Learning* (pp. 115-123). Tahoe City.

Dimitras, A. I., Slowinski, R., Susmaga, R., & Zopounidis, C. (1999). Business Failure Prediction Using Rough Sets. *European Journal of Operational Research,* 114(2), 263-280.

Duan, K., & Keerthi, S. (2005). Which is the Best Multiclass SVM Method? An Empirical Study. In *Multiple Classifier Systems* (pp. 278-285). Seaside.

Dutka, A. F. (1995). *AMA Handbook for Customer Satisfaction*. Lincolnwood: NTC Business Books.

Figueira, J., Mousseau, V., & Roy, B. (2005). ELECTRE methods. In *Multiple Criteria Decision Analysis: State of the Art Surveys* (pp. 133-153). New York: Springer.

Fishburn, P. C. (1967). Additive Utilities with Incomplete Product Set: Applications to Priorities and Assignments. *Operations Research*, 15(3), 537-542.

Fisher, R. A. (1936). The Use of Multiple Measurements in Taxonomic Problems. *Annals of Human Genetics,* 7(2), 179-188.

Frank, E., & Hall, M. (2001). A Simple Approach to Ordinal Classification. In *12th European Conference on Machine Learning* (pp. 145-156). Freiburg.

Frank, E., & Kramer, S. (2004). Ensembles of Nested Dichotomies for Multi-class Problems. In *Proceedings of the Twenty-First International Conference on Machine Learning*, (pp. 39-47). New York.

Fürnkranz. J. (2002). Round Robin Classification. *The Journal of Machine Learning Research*, 2(Mar), 721-747.

Hastie, T., & Tibshirani, R. (1998). Classification by Pairwise Coupling. *The Annals of Statistics*, 26(2), 451-471.

Huhn, J. C., & Hullermeier, E. (2009). Is an Ordinal Class Structure Useful in Classier Learning? *International Journal of Data Mining, Modelling and Management*, 1(1), 45-67.

Jacquet-Lagreze, E., & Siskos, J. (1982). Assessing a Set of Additive Utility Functions for Multicriteria Decision-Making, the UTA Method. *European Journal of Operational Research*, 10(2), 151-164.

Jacquet-Lagreze, E., & Siskos, J. (2001). Preference Disaggregation: 20 years of MCDA Experience. *European Journal of Operational Research,* 130, 233-245.

John, G., & Langley, P. (1995). Estimating Continuous Distributions in Bayesian Classifiers. In *Eleventh Conference on Uncertainty in Artificial Intelligence*, (pp. 338-345). San Mateo.

Keerthi, S. S., Shevade, S. K., Bhattacharyya, C., & Murthy, K. R. K. (2001). Improvements to Platt's SMO Algorithm for SVM Classifier Design. *Neural Computation*, 13(3), 637-649.

Kim, K., & Ahn, H. (2012). A Corporate Credit Rating Model Using Multi-Class Support Vector Machines with an Ordinal Pairwise Partitioning Approach. *Computers & Operations Research,* 39(8), 1800-1811.

Landwehr, N., Hall, M. & Frank, E. (2005). Logistic Model Trees. *Machine Learning* 59(1), 161-205.

Malakooti, B., & Zhou, Y. Q. (1994). Feedforward Artificial Neural Networks for Solving Discrete Multiple Criteria Decision Making Problems. *Management Science*, 40(11), 1542-1561.

McFadden, D. (1973). Conditional Logit Analysis of Qualitative Choice Behavior. In *Frontiers in Econometrics* (pp. 105-142). New York: Academic Press.

Mousseau, V., & Slowinski, R. (1998). Inferring an ELECTRE TRI Model from Assignment Examples. *Journal of Global Optimization,* 12(2), 157-174.

Lotov, A., & Miettinen, K. (2008). Visualizing the Pareto Frontier. In *Multiobjective Optimization: Lecture Notes In Computer Science Volume 5252* (pp.213-243). Berlin-Heidelberg: Springer.

Pardalos, P. M., Michalopoulos, M., & Zopounidis, C. (1997). On the Use of Multi-Criteria Methods for the Evaluation of Insurance Companies in Greece, inc. Zopounidis (ed.), In *New Operational Approaches for Financial Modeling* (pp. 271–283). Berlin-Heidelberg: Physica.

Pokorny, J. (2011). *Information Systems Development: Business Systems and Services: Modeling and Development.* New York: Springer.

Quinlan, J. R. (1993). *C4. 5: Programs for Machine Learning*. San Mateo: Morgan Kaufmann.

Rifkin, R., & Klautau, A. (2004). In Defense of One-Vs-All Classification. *Journal of Machine Learning Research,* 5(Jan), 101-141.

Roy, B. (1996). *Multicriteria Methodology for Decision Aiding*. New York: Springer.

Tan, P., Steinbach, M., & Kumar, V. (2006). *Introduction to Data Mining.* Boston: Addison-Wesley.

Torgo, L. (2001). *Regression Data Sets.* University of Porto, Faculty of Economics, Porto, Portugal. [http://www.dcc.fc.up.pt/~ltorgo/Regression/DataSets.html]

Triantaphyllou, E. (2000). *Multi-Criteria Decision Making Methods: A Comparative Study*, Boston: Kluwer Academic Publishers.

Wang, X., & Triantaphyllou, E. (2008). Ranking Irregularities When Evaluating Alternatives by Using Some ELECTRE methods. *Omega,* 36(1), 45-63.

Witten, I., & Frank, E. (2000). *Weka: Practical Machine Learning Tools and Techniques with Java Implementations.* San Francisco: Morgan Kaufmann.

Wu, T., Lin, C., & Weng, R. (2004). Probability Estimates for Multi-Class Classification by Pairwise Coupling. *The Journal of Machine Learning Research*, 5(Aug), 975-1005.

Zopounidis, C., & Doumpos. M. (2002). Multicriteria Classification and Sorting Methods: A Literature Review. *European Journal of Operational Research,* 138(2), 229-246.

VITA

Forrest Justin Osterman was born in 1979 in Mountain View, California, and raised in the city of Picayune, Mississippi. He was admitted to Louisiana State University at age 15 through the early admission program. He was a National Merit Finalist, and was awarded the Chevron Top 100 scholarship. During the summers he worked as a software developer for Nation Computer Services at John C. Stennis Space Center in support of the Naval Oceanographic Office. He graduated with a bachelor's degree in Computer Science in May, 2000.

Forrest worked as a software engineer at Vedalabs, a media software company, until 2001, and Innovative Emergency Management, from 2002 until 2006. He entered the Ph.D. program in Computer Science in the fall of 2006 on a research fellowship under the direction of Professor Evangelos Triantaphyllou. He will graduate in May, 2015, and plans to provide consulting services in relation to data mining and optimization related tasks.