A PROBABILISTIC APPROACH TO MULTI CRITERIA SORTING PROBLEM

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

ASLI GÜL BUĞDACI

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
INDUSTRIAL ENGINEERING

AUGUST 2009

Approval of the thesis:

A PROBABILISTIC APPROACH TO MULTI CRITERIA SORTING PROBLEM

submitted by **ASLI GÜL BUĞDACI** in partial fulfillment of the requirement for the degree of **Master of Science in Industrial Engineering Department, Middle East Technical University** by,

Prof. Dr. Canan Özgen
Dean, Graduate School of **Natural and Applied Sciences** _____

Prof. Dr. Nur Evin Özdemirel
Head of Department, **Industrial Engineering** _____

Prof. Dr. Murat Köksalan
Supervisor, **Industrial Engineering Dept., METU** _____

Assist. Prof. Dr. Selin Bilgin Özpeynirci
Co-supervisor, **Industrial Systems Engineering Dept., IUE** _____

Examining Committee Members:

Assoc. Prof. Dr. Yasemin Serin
Industrial Engineering Dept., METU _____

Prof. Dr. Murat Köksalan
Industrial Engineering Dept., METU _____

Assist. Prof. Dr. Selin Bilgin Özpeynirci
Industrial Systems Engineering Dept., IUE _____

Prof. Dr. Meral Azizoğlu
Industrial Engineering Dept., METU _____

Assist. Prof. Dr. Özgür Özpeynirci
Logistics Management Dept., IUE _____

Date: **05.08.2009**

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last name : Aslı Gül BUĞDACI

Signature :

ABSTRACT

A PROBABILISTIC APPROACH TO MULTI CRITERIA SORTING PROBLEM

Buğdacı, Aslı Gül
M.S., Department of Industrial Engineering
Supervisor : Prof. Dr. Murat Köksalan
Co-supervisor : Assist. Prof. Dr. Selin Bilgin Özpeynirci

August 2009, 109 pages

We aim to classify alternatives evaluated in multiple criteria among preference ordered classes assuming an underlying additive utility function. We develop a probabilistic classification method by calculating the probability of an alternative being in each class. We assign alternatives to classes based on threshold probabilities. We require the decision maker to place an alternative to a class when no alternatives satisfy the required thresholds. We find new probabilities for unassigned alternatives in the light of new information and repeat the procedure until all alternatives are classified.

We implemented our algorithm to classify MBA programs among preference ordered groups. We evaluate our algorithm based on the number of misclassified alternatives and the number of alternatives placed by the decision maker.

Keywords: multi criteria sorting, additive utility function, interactive approach.

ÖZ

ÇOK AMAÇLI GRUPLANDIRMA PROBLEMİ İÇİN BİR OLASILIKSAL YAKLAŞIM

Buğdacı, Aslı Gül
Yüksek Lisans, Endüstri Mühendisliği Bölümü
Tez Yöneticisi : Prof. Dr. Murat Köksalan
Ortak Tez Yöneticisi : Yrd. Doç. Dr. Selin Bilgin Özpeynirci

Ağustos 2009, 109 Sayfa

Çok amaç altında değerlendirilen seçenekleri tercih sıraları olan sınıflara atamayı hedefliyoruz. Bunu yapmak için karar vericinin parçalı doğrusal bir fayda fonksiyonuna sahip olduğunu varsayıyoruz. Her seçeneğin her sınıfta olma olasılığını hesaplayan olasılıksal bir gruplandırma metodu geliştiriyoruz. Hesaplanan olasılıkları, olasılık eşiğiyle kıyaslayarak seçenekleri gruplara atıyoruz. Yerleştirme yapılamadığı durumlarda karar vericinin bir seçeneği sınıfına yerleştirmesini istiyoruz. Yeni bilgi ışığında, yerleştirilmemiş seçenekler için yeni olasılıklar hesaplıyoruz. Tüm seçenekler sınıflandırılana kadar bu yöntemi tekrar ediyoruz.

Yöntemimizi MBA programlarını aralarında tercih sırası bulunan sınıflara yerleştirmek için uyguladık. Yöntemimizin değerlendirilmesini, yanlış sınıflandırılan seçenek sayısını ve karar verici tarafından yerleştirilen seçenek sayısını temel alarak yapıyoruz.

Anahtar Kelimeler: çok amaçlı gruplandırma, toplamsal fayda fonksiyonu, etkileşimli yaklaşım.

To my parents and
my dear brother

ACKNOWLEDGEMENTS

I am grateful to my supervisor Prof. Dr. Murat Köksalan for his guidance and support. I am indebted to him for his valuable advises and encouragements throughout this study. I am thankful to Assist. Prof. Dr. Selin Bilgin Özpeynirci for her support in this study. I am also thankful to Assoc. Prof. Dr. Yasemin Serin for her contribution to this study.

I would like to thank my mother Nida Buğdacı for her gentle love, endless support and guidance throughout my life. I thank my father Ali Buğdacı for his love and his faith in me. I also thank my brother Okan Can for filling my life with laughter.

I am thankful to my dear friend Gülçin Haktanır for being with me whenever I need. I thank Gülşah Karakaya, Burcu Özsayın and Diclehan Tezcaner for their kind friendships. They helped me and encouraged me both in this study and in my life. I thank Melih Çelik for helping me with kindness whenever I need. I also thank Zehra Atlı for her warm friendship.

I would like to thank TÜBİTAK (The Scientific and Technological Research Council of Turkey) for supporting this study through a graduate study scholarship.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

CHAPTER 1

INTRODUCTION AND LITERATURE REVIEW

1.1. INTRODUCTION

Multi-criteria problems dealing with discrete set of alternatives can be considered in different groups such as selecting the best alternative, ranking the alternatives, classifying the alternatives and sorting the alternatives into preference ordered classes. Multi-criteria sorting problems assign a discrete set of alternatives to predefined classes which are preference ordered. Each alternative has an evaluation on each criterion and an alternative can be defined as a vector composed of the scores on criteria. The decision maker (DM) has preferences regarding these criteria such as more is better or less is better. Formally there are m discrete alternatives defined by n criteria. The scores of the alternatives on each of the n criteria are known. The problem is how to assign these alternatives to q preference ordered classes.

There are two main approaches to multi-criteria sorting problems. One uses outranking relations to classify alternatives and the other assumes an underlying utility function for the DM. Outranking is a binary relation between two alternatives. Alternative i outranks alternative j if there are enough arguments to confirm that alternative i is at least as good as alternative j and there is no strong opposition to this argument. The assignment of the alternatives is done based on the outranking relations between the alternatives and reference profiles that separate each of the consecutive classes. They are used to derive preference and incomparability relationships with the given criteria weights and threshold levels. In the utility function method, a utility function is estimated which assigns

a utility value to each alternative. The utility threshold levels between consecutive classes are determined. The alternatives are assigned to the classes by comparing the estimated utility of alternatives with utility threshold levels.

In both approaches, the estimation of the parameters is a major issue. The reference profiles, weights and threshold levels should be estimated while using outranking relations. The utility thresholds and weights should be estimated while using the utility function method. These parameters can be estimated by directly asking to the DM. If the DM finds it hard to give exact values of the parameters, then they can be inferred from an exemplary set given by the DM which is capable of reflecting the preferences of the DM. Jacquet-Lagzere and Siskos (2001) state that this exemplary set can consist of past data, a subset of the available data or a representative fictitious data. Decision support systems (DSS) have been developed to find the parameters which can regenerate a set of example assignments made by the DM. Then the estimated model can be used to classify unassigned alternatives. Some of the DSSs developed let the interaction of the DM. As the DM sees the results, he/she can add or remove constraints or modify the assignment.

The DM can give information at the beginning of the process and the assignment of alternatives is done based on this preference information. However, there are also algorithms which assign alternatives interactively with the DM. They take information from the DM during the assignment when needed.

There are many real life multi-criteria sorting applications. Especially in financial area, multi-criteria sorting methods are applied to many different cases such as credit risk assessment for firms, stock evaluation, and country risk assessment. There are also other applications in areas such as energy management.

1.2. LITERATURE REVIEW

Zopounidis and Doumpos (2002) have written a literature review on multi-criteria sorting and classification methods. They first define what sorting is and mention the applications from different areas. Then they mention 4 criteria aggregation models: Outranking relations, estimating a utility function, rough set approach and neural networks. They mention model development techniques used and explain which ways are used to estimate the parameters. They discuss mathematical models which have been used while testing the models. After this, they mention decision support systems developed and their applications to real world problems.

One main approach is using outranking relations to solve multi-criteria sorting problems. Mousseau et al. (2000) describe the ELECTRE TRI method by referring to the studies of Yu and Roy & Bouyssou. ELECTRE TRI uses outranking relations to assign alternatives to classes. In ELECTRE TRI method, first the comparison of each alternative with each reference profile is done on each criterion. Partial concordance indices on each criterion are calculated by using indifference and preference thresholds. Then they are multiplied with the weights of each criterion to find comprehensive concordance index. Discordance indices are calculated by using preference and veto thresholds. Then by using concordance index and discordance indices, a credibility index is calculated for estimation of the strength of the argument alternative j is at least as good as reference profile b_k. The credibility index for the reverse argument is also calculated. Two credibility indices are compared with a cutting level to derive preference and incomparability relationships between alternatives and reference profiles. After deriving the relationships, the assignment of the alternatives to the classes can be done in two procedures. In pessimistic procedure, an alternative is assigned to the highest class which it outranks its upper profile. In optimistic procedure, an alternative is assigned to the lowest class whose upper profile is preferred to the alternative.

They propose software called ELECTRE TRI Assistant and they explain how to use it. The software estimates the parameters from some assignment examples given by the DM. ELECTRE TRI Assistant estimates the parameters by nonlinear programming which is formed for pessimistic procedure without veto. The objective of the program is to find parameters which can result in credibility indices as far as from the cutting level. It is to maximize the minimum distance. While using software, the user can give preference information about the weights and cutting level by giving bounds or comparisons. Lastly they work on an example data by taking reference profiles and thresholds as given to show how the software helps the user to deal with inconsistencies.

Mousseau et al. (2001) argue that the nonlinear program in ELECTRE TRI can be written as a linear program if reference profiles and thresholds are considered as given. The objective of LP is same as the NLP. Based on a real world data, they discuss some sensitivity issues. They say that as the objective function value of LP increases, the model becomes more stable. If a larger set of assignment examples is used, then the model incorporates more preference information but it becomes less stable. To see if the model is able to detect inconsistencies, they make a check by assigning an example alternative into a wrong class. Regarding this check, they conclude that the model shows good ability. Lastly they try different objectives but conclude that this does not bring any significant improvement.

Dias and Mousseau (2003) present a software called IRIS (Interactive Robustness Analyses and Parameters' Inference for Multi-criteria Sorting Problems) which infers the combination of the weights and cutting level from an example set. They assume that the reference profiles and threshold levels are given as input. Combinations are able to regenerate the given assignment. Again with the objective of maximizing the minimum slack, IRIS chooses one combination of parameters. The proportion of the combinations of the parameters is considered as an indicator of the precision of the input. Geometric average of the number of

classes to which each action can be assigned is considered as an indicator of precision of the output. IRIS gives user also the ability to insert constraints on the parameter values. In an interactive manner, the user can insert new examples or new constraints. If the added constraints cause an inconsistency, IRIS presents ways to get rid of this inconsistency.

Damart et al. (2007) make an application using IRIS to sort loan applications to a hypothetical bank when there are 5 criteria and 4 categories. There are 4 experts to assign 15 past loan applications to classes. To get rid of inconsistencies, there can be discussions between the experts. To reach a consensus, they propose some measures such as proportion of the experts that agree with the current assignment and how many shifts it will cost to a particular expert to agree. If they cannot agree in anyway, the process stops. The aim is to incorporate multiple expert views. Here IRIS is used if there is a possible solution which can take into account the opinions of all the experts.

Köksalan et al. (2009a) use outranking relations to assign alternatives into classes. They take an initial assignment from the DM. They assume preference and indifference thresholds are known but criteria weights are unknown. Alternatives to be placed are compared with the reference alternatives and assigned to categories.

Doumpos and Zopounidis (2004a) propose a sorting method based on the pairwise comparisons of alternatives. They suggest an algorithm for two class case but state that the algorithm can be generalized to multiple class case. They compare the criterion values of unassigned alternatives and already assigned alternatives. For each unassigned alternative, they find a preference index over each alternative in class 2 and take average of these indices. They name it as positive flow. They also find a preference index for each alternative in class 1 over each unassigned alternative. For each unassigned alternative, they take average of these indices over the alternatives in class 1 and name it as negative

flow. They find the net flow for each unassigned alternative. They make classification by comparing the net flow with a cut off level. The cut off level can be given by the DM or can be estimated by solving an LP that minimizes the weighted sum of violations of classification rule for assigned alternatives.

The other main approach is to assume the DM has an underlying utility function. If the criteria are preferentially independent, then an additive utility function can be used. Criteria weights indicate the relative importance of an improvement in one criterion from its worst level to its best level compared with the changes in other criteria. Pöyhönen and Hamalainen (2001) explain some of these methods. One is direct point allocation which requires the DM to distribute 100 points among the criteria and weights of each criterion are calculated as ratios of number of points distributed to 100 points. Another method is SMART that is first described by Edwards (1977). It requires the DM to rank the criteria starting from least important one. 10 points are given to the least important criterion. Then the DM gives points to each criterion by comparing them with the least important criterion. The numbers of points given by the DM are normalized to find the weights. SMARTER has been developed by Edwards and Barron (1994). SMARTER requires the DM to rank criteria starting from the least important one and calculates the weights from this ranking. SWING described by von Winterfeld and Edwards (1986) assumes that the DM has lowest scores in all criteria and asks the DM in which criterion he/she wants the highest score first. 100 points are assigned to that criterion. Then the DM chooses the criteria in the order he/she wants to improve to the highest score and assigns number of points. The weights are found by normalizing the number of points given by the DM.

Larichev and Moshkovich (1994) suggest a sorting method using dominance relations. They look at the dominance relations between the unassigned alternatives and already assigned ones. The values on each criterion scale are in verbal form and the DM orders each criterion scale according to his/her preferences.

UTADIS (Utilites Additives Discriminantes) method is developed by Jacquet-Lagreze (see for example Doumpos and Zopounidis, 2004b). UTADIS assumes an underlying piecewise additive utility function. Each criterion range is divided into subintervals. It is needed to estimate the utility values of each subinterval on each criterion. UTADIS estimates the model parameters from a reference set in a way that gives the minimum amount of misclassification for the alternatives in the reference set. Then it uses the estimated parameters to assign alternatives into classes.

Diakoulaki et al. (1999) use UTADIS to sort 14 countries into 3 classes according to energy intensities. Energy intensity is the ratio of energy consumption per unit output. 13 factors are determined which have effect on reaching a desired level of energy efficiency. The aim of the study is to see the relative importance of the criteria and how it is changed through the years. They want to find the most important factors affecting energy efficiency.

Zopounidis and Doumpos (2000) propose a DSS called PREFDIS (Preference Discrimination) which uses UTADIS and its 3 variants to sort the alternatives into classes. They propose a method to cope with non-monotone utility functions. If the utility function is non-monotone, the article suggests dividing criteria into subintervals in which the DM has monotone preferences. Each subinterval is divided to subintervals inside and the model is modified considering this issue. There are variants of UTADIS used to increase the accuracy of classification. After UTADIS is solved, the first variant UTADIS 1 maximizes the distances of the correctly classified alternatives from the threshold levels. UTADIS 2 minimizes the number of misclassified alternatives. UTADIS 3 combines the first and second variants. They present two applications in which PREFDIS is used. The first one is to make a country risk assessment and assign the countries to classes according to the income level. In this study, 66 countries are evaluated on 12 criteria. The countries are assigned to 4 classes according to income level such as high, upper-middle, lower-middle and low. On the second application 60 Greek

firms are classified into two classes: acquired and non-acquired. The classification is done according to the 10 financial ratios.

Doumpos and Zopounidis (2004b) make an experimental design to see the effects of the parameters involved in UTADIS. They want to see the effects of the way the additive utility functions are formed on the performance of the developed model and the stability of the significance of the evaluation criteria of the model. The performance of the model is measured with classification accuracy which is the consistency of the recommendations of the model with the actual decisions. The stability of the model is measured as the variance of the significance of the evaluation criteria over all the solutions obtained during the post optimality stage.

Zopounidis and Doumpos (2001) propose a hierarchical multi-criteria sorting method called M.H.DIS (Multi-Group Hierarchical Discrimination). This method works iteratively and each time assigns alternatives to two classes. M.H.DIS first assigns the alternatives to the best class. Then it takes the unassigned alternatives in the other class and applies the same procedure. If there are q classes, this procedure should be applied for $q-1$ times. At each step, the model estimates two utility functions from alternatives whose classes are given by the DM. One is an increasing function with respect to criterion values and gives the utility of placing alternatives to class k. The other one is a decreasing function and it gives the utility of assigning alternatives to a lower class than k. At each step, the utility values for alternatives are compared. An alternative is assigned to class k if first utility function has a larger value and to a lower class if second utility function has a larger value. At each step, after the assignment 3 mathematical models are solved to improve the classification accuracy. First model is a linear program and minimizes the overall misclassification error. If there are misclassified alternatives according to the result of the first model, the second model which is a mixed integer model wants to minimize the number of misclassification. Correctly classified alternatives according to model 1 are taken as given. After this, a third model which is a linear program is solved to maximize

the minimum distance (the difference between two utility values) of correctly classified alternatives while misclassified alternatives remain as misclassified. They apply this model to country risk assessment case.

Pasiouras et al. (2007) makes a multi-criteria application to UK firms. After an audit, the firms are classified to two groups. First group consists of the firms that are able to meet the requirements and the second group consists of the firms that are not able to meet the requirements. They apply both UTADIS and M. H. DIS to the company sorting problem for 4 different years each time using all the past data set as the reference set. They also use two statistical techniques which are debit analyses and logit analyses. Here M.H.DIS is applied in one step. Both UTADIS and M.H.DIS gives better results than statistical techniques but they get a classification accuracy of around 70%. They conclude that UTADIS and M.H.DIS have a similar performance for this problem.

Zopounidis (1999) discusses how multi-criteria decision aids should apply to the financial problems and list the benefits of them. He mentions the suitability of multi-criteria approaches to the financial problems by giving application examples briefly.

Doumpos and Zopounidis (2001) develop a system called FINCLASS that uses UTADIS and its 3 variants in the classification problems at financial area. They take the data of past years as reference set in UTADIS and estimate the parameters of the underlying model. They use these parameters to classify the alternatives in current year. They say that FINCLASS can be used for bankruptcy risk evaluation, credit granting and assessment of corporate performance.

Weber (1985) proposes a method for ranking a set of alternatives. In the method, he assumes an underlying piecewise additive utility function. He requires the DM to give information on the utility values of hypothetical alternatives. These hypothetical alternatives come from an orthogonal design. Each subinterval limit

of the marginal utility function of each criterion is thought as a level in the orthogonal design concept. Criterion values of orthogonal alternatives are on the limits of subintervals. Weber (1985) mentions two methods. First one requires the DM to give utility values of orthogonal alternatives which results in getting the utility function of the DM completely by solving mathematical equations. The other one requires the DM to give utility ranges for orthogonal alternatives. Linear programs are solved to get information from the given ranges. He uses this information to rank alternatives. If a complete ranking cannot be obtained, he suggests to calculate the probability of each alternative being better than the other alternatives. He uses these probabilities to rank the alternatives. Although he proposes an approach to rank the alternatives, the ideas can be used in the sorting problem as well.

Greco et. al. (2008) propose a multi-criteria ranking method. The preference information is taken from the DM as pairwise comparison of alternatives. They find necessary ranking on two alternatives if all utility functions compatible with the preferences of the DM results a larger utility value for one of them. They find a possible ranking if there exists at least one utility function compatible with the preferences of the DM resulting a larger utility value for one of them. They suggest an interactive procedure for ranking by using necessary and possible ranking relations.

Köksalan and Ulu (2003) propose an interactive approach by assuming an underlying additive utility function. There are already classified alternatives and some preference information coming from the DM. The algorithm is proposed for a linear utility function. For an alternative, dominance relations are checked first. The class range the alternative can be placed is found. Then the convex dominance relations between the alternative and already assigned alternatives are checked to tighten the class range the alternative can be placed. After this, weights space reduction method is applied and it is checked if there are feasible weights which make the alternative better or worse than the alternatives in a class.

If no feasible weights are found, a tighter class range is obtained. The found range is presented to the DM and asked to select the class to place the alternative. It is said that if there are not too many criteria, it is an effective procedure when there is a large number of alternatives.

Ulu and Köksalan (2001) propose another interactive approach to place the alternatives to acceptable and unacceptable sets. The DM is asked to place some alternatives to acceptable, barely acceptable and unacceptable classes. Two other classes are defined for the alternatives not in acceptable class and not in unacceptable class. For linear utility function case, dominance relations, convex domination and weights space reduction techniques are applied. For quasiconcave function cases, they first suggest to use dominance relations. Then they use the idea that convex combination of acceptable alternatives are acceptable and suggest to check if alternatives can be written as convex combinations of acceptable alternatives. Then they use cone generating techniques. Alternatives that are inefficient with respect to a cone generated by acceptable alternatives and an unacceptable alternative are unacceptable. They check if alternatives are inefficient with respect to cones generated. For the general monotone case, they suggest to use dominance relations.

Köksalan and Özpeynirci (2009) propose an interactive method that assumes an underlying piecewise additive utility function. They find the best and the worst class an alternative can be placed by checking if it is feasible to place the alternative to particular classes. They narrow down the number of classes the alternative can be placed. They place the alternatives whose classes are narrowed down to one. If it is not narrowed down to one, they ask the DM to place the alternative to one of the classes from the narrowed set.

In this study, we propose an interactive approach. We suggest a probabilistic classification method. The algorithm we suggest is a generalization of the algorithm Köksalan and Özpeynirci (2009) propose. We assume that the DM has

an underlying additive piecewise utility function. We calculate the probabilities of each alternative being in each class with the available class information given by the DM. We make classification by comparing the probabilities with a given threshold value. If we are not able to make a classification, we will ask the DM class of an alternative and continue with the new information. We evaluate our model by number of alternatives misclassified and number of alternatives placed by the DM. We implement our algorithm to assign MBA programs to 3 preference ordered classes.

We suggest a probabilistic algorithm and the probabilistic approaches are not very common. The DM can choose the probability threshold by considering the amount of information s/he wants to give and the amount of misclassification s/he accepts. In our algorithm, we use linear programs to calculate probabilities. Using linear programs instead of integer programs brings computational benefits.

In Chapter 1, we make a brief introduction and present the literature review. In Chapter 2, we mention some methods. We use some ideas from those methods in our study. In Chapter 3, we present the algorithm we developed. We give the results of the implementation of our algorithm on example problems in Chapter 4. Lastly, we give some conclusions and present a further study area in Chapter 5.

CHAPTER 2

BACKGROUND

We will briefly explain 3 different methods used for multi-criteria sorting and ranking problems. We will use some of the ideas mentioned in these methods in developing our algorithm. First how sorting problems can be handled by using UTADIS method will be explained and the notation we use in our algorithm will be presented. Then an interactive sorting approach developed by using the ideas of UTADIS will be mentioned. Lastly, a method for solving the ranking problem will be discussed. We will utilize some of the ideas developed for the ranking problem in our algorithm for the sorting problem.

2.1. UTADIS

UTADIS method is developed by Jacquet-Lagreze (see for example Doumpos and Zopounidis, 2004b). There is a set A of m alternatives $a_1, a_2, ..., a_m$ evaluated using n criteria. UTADIS assigns these m alternatives to one of the q categories $C_1, C_2, ..., C_q$ where C_1 is the most and C_q is the least preferred one. Each criterion range is divided into b_i subintervals and the marginal utility on each criterion is thought to be piecewise linear on the criterion range. The utility of each alternative is the sum of the marginal utility of alternatives on each criterion. The assignment of alternatives to the classes is done according to the comparison of utility thresholds with the utility values of alternatives.

Let j be the index for alternatives where $j = 1,2,...,m$, i be the index for criteria where $i = 1,2,...,n$, k be the index for classes where $k = 1,2,...,q$ and p be the index for subintervals where $p = 1,2,...,b_i$.

Let $g_i(a_j)$ be the score of alternative j on criterion i. Then $U[g(a_j)]$ is the global utility of alternative j where $g(a_j) = [g_1(a_j), g_2(a_j),..., g_n(a_j)]$. Let $u_i[g_i(a_j)]$ be the marginal utility of alternative j on criterion i. Since individual utility function of each criterion is piecewise linear, each criterion range is divided into b_i subintervals $[g_i{}^p, g_i{}^{p+1}]$, $p = 1,2,...,b_i$.

Let subinterval p have an unknown utility value of w_{ip} on criterion i where $w_{ip} = u_i[g_i{}^p] - u_i[g_i{}^{p+1}]$. If $g_i{}^{r_{ji}} \le g_i(a_j) \le g_i{}^{r_{ji}+1}$ for some $1 \le r_{ji} \le b_i$, then $u_i[g_i(a_j)]$ is found by linear interpolation between $u_i[g_i{}^p]$ and $u_i[g_i{}^{p+1}]$. Since this is an additive utility function, $U[g(a_j)] = \sum_{i=1}^{n} u_i[g_i(a_j)]$. We can also write

$$U[g(a_j)] = \sum_{i=1}^{n}\left[\sum_{p=1}^{r_{ji}-1} w_{ip} + \frac{g_i(a_j) - g_i{}^{r_{ji}}}{g_i{}^{r_{ji}+1} - g_i{}^{r_{ji}}} w_{ir_{ji}}\right].$$

Classification is done by comparing the overall utility of each alternative with the utility thresholds. Let u_k be the utility threshold that separates the classes.

$U[g(a_j)] \ge u_1$, then a_j is in class 1.

$u_k \le U[g(a_j)] < u_{k-1}$, then a_j is in class k for $k = 2,...,q-1$

$U[g(a_j)] < u_{q-1}$, then a_j is in class q.

There is a set of reference alternatives whose classes are known and UTADIS aims to estimate the parameters of the underlying model in a way to minimize the misclassification errors. Misclassification errors are defined as follows:

$s_j^+ = \max\{0, u_k - U[g(a_j)]\}$, for every $a_j \in C_k$

$s_j^- = \max\{0, U[g(a_j)] - u_{k-1}\}$, for every $a_j \in C_k$

Let $m_k = |C_k|$ and $m_r = \sum_{k=1}^{k=q} m_k$.

The linear program used in UTADIS is as follows:

$$Min \quad \sum_{k=1}^{q} \left[\frac{\sum_{a_j \in C_k} (s_j^+ + s_j^-)}{m_k} \right]$$

s.to

$$U[g(a_j)] = \sum_{i=1}^{n} \left[\sum_{p=1}^{r_{ji}-1} w_{ip} + \frac{g_i(a_j) - g_i^{r_{ji}}}{g_i^{r_{ji}+1} - g_i^{r_{ji}}} w_{ir_{ji}} \right], \quad j = 1,2,...,m_r \qquad (2.1)$$

$$U[g(a_j)] - u_k + s_j^+ \geq 0, \quad \forall a_j \in C_k \quad (k = 1,2,...,q-1) \qquad (2.2)$$

$$U[g(a_j)] - u_{k-1} - s_j^- \leq -d, \quad \forall a_j \in C_k \quad (k = 2,3,...,q) \qquad (2.3)$$

$$\sum_{i=1}^{n} \sum_{p=1}^{b_i-1} w_{ip} = 1, \quad \forall i = 1,2,...,n, \quad p = 1,2,...,b_i \qquad (2.4)$$

$$u_k - u_{k+1} \geq s, \quad \forall k = 1,2,...,q-2 \qquad (2.5)$$

$$w_{ip} \geq 0, \quad \forall i = 1,2,...,n, \quad p = 1,2,...,b_i \qquad (2.6)$$

$$s_j^+ \geq 0, \quad s_j^- \geq 0, \quad j = 1,2,...,m_r \qquad (2.7)$$

where d and s are user-defined constants.

Constraint (2.1) expresses each $U[g(a_j)]$ in terms of w_{ip} values. If an alternative is assigned to C_k, then the utility of it should be greater than or equal to the utility threshold u_k that is the lower limit of class k. Constraint (2.2) indicates this

condition but it allows violation by an amount of s_j^+. Utility of an alternative assigned to C_k should be smaller than the utility threshold u_{k-1} that is the upper limit of class k. Constraint (2.3) satisfies this condition by allowing violation by an amount of s_j^-. Constraint (2.4) expresses that sum of utility values of all subintervals should be equal to 1. This means utility of an alternative having the maximum scores in all criteria is 1. Constraint (2.5) provides that the upper limit of a class is greater than its lower limit. Constraint (2.6) indicates that the utility values of subintervals should be nonnegative. Constraint (2.7) ensures that the amount of misclassification for each alternative is nonnegative.

UTADIS estimates the parameters of the model with the class information of the alternatives in the reference set. The objective is to minimize the sum of misclassification errors of the alternatives. It aims to find the model parameters that can give the original classification of the DM for the reference set with the minimum amount of misclassification. After estimating the parameters that minimize the sum of misclassification errors, the other alternatives are classified by calculating their utility values and comparing the utility values with the estimated utility thresholds.

2.2. AN INTERACTIVE SORTING METHOD FOR ADDITIVE UTILITY FUNCTIONS

Köksalan and Özpeynirci (2009) demonstrate that UTADIS can misclassify many alternatives even when there is considerable preference information on hand. UTADIS assumes that there is an underlying piecewise additive utility function. UTADIS estimates the parameters of the underlying model and assigns alternatives to classes using these parameters. Köksalan and Özpeynirci (2009) argue that there are many parameters to be estimated which result in many alternative optimal solutions for the linear program used in UTADIS. This means there are many different combinations of parameters that can give the original

classification of the DM for the reference set. The classification of alternatives is done by the estimated parameters. Therefore, classification of alternatives depends on which alternative optimal solution is selected.

They make experiments with UTADIS and show that various numbers of alternative solutions can be obtained with UTADIS. They solve an example problem with 3 classes and get a solution with no misclassification. Then by setting misclassification to 0 and solving the model with different objectives, they end up with many different solutions and different classifications depending on these solutions. With different solutions, each of the nonreference alternatives can be in all 3 classes. They look at the deviations of the estimated parameters obtained from the different solutions and underlying parameters. They say that the deviations are very large. With these results, they make an observation that UTADIS and its variations are not successful for the particular problem they solved.

They propose an interactive method for classifying the alternatives and the approach guarantees to classify all the alternatives correctly. They use similar ideas used by Köksalan and Ulu (2003) for linear utility function cases. They select an alternative and solve integer programs to find the possible classes the alternative can be placed in. If the worst and best class of the alternative is same, they place the alternative to that class. If the number of classes the alternative can be placed is greater than 1, then they present the DM the possible classes of this alternative and ask the DM to place. After placing this alternative, they select another alternative. Each time the alternative is placed either by finding the class of it or asking the DM. This procedure repeats until all alternatives are classified.

Let A be the set of all alternatives and S be the set of alternatives whose classes are not known. Let C_k be the set of alternatives known to be in class k, $k = 1,2,...,q$. Let $a_t \in S$. Let C_t^w denote the worst class a_t can be assigned to and C_t^b denote the best class a_t can be assigned to.

Here the notation presented in UTADIS is used. One difference is that v_j is used instead of $U[g(a_j)]$ to denote utility of alternative j. They also define another decision variable as follows.

$$x_{jk} = \begin{cases} 1 & \textit{if alternative } j \textit{ is assigned to class } k \\ 0 & \textit{otherwise} \end{cases}$$

The following IP named $IP1_{a_t,h}$ checks whether the worst class of a_t is class h.

Max e

s.t.

$$v_j = \sum_{i=1}^{n}\left(\sum_{p=1}^{r_{ji}-1} w_{ip} + \frac{g_i(a_j) - g_i^{r_{ji}}}{g_i^{r_{ji}+1} - g_i^{r_{ji}}} w_{ir_{ji}} \right), \quad \forall a_j \in A \tag{2.8}$$

$$u_k \le v_j + M(1 - x_{jk}) + e, \quad k = 1,2,...,q-1, \quad \forall a_j \in S - \{a_t\} \tag{2.9}$$

$$u_{k-1} \ge v_j - M(1 - x_{jk}) + e, \quad k = 2,3,...,q, \quad \forall a_j \in S - \{a_t\} \tag{2.10}$$

$$\sum_{i=1}^{n}\sum_{p=1}^{b_i-1} w_{ip} = 1 \tag{2.11}$$

$$u_k - u_{k+1} \ge s, \quad k = 1,2,...,q-2 \tag{2.12}$$

$$\sum_{k=1}^{q} x_{jk} = 1, \quad \forall a_j \in S - \{a_t\} \tag{2.13}$$

$$u_1 \le v_j, \quad \forall a_j \in C_1 \tag{2.14}$$

$$u_k \le v_j \le u_{k-1} - e, \quad k = 2,3,...,q-1, \quad \forall a_j \in C_k \tag{2.15}$$

$$v_j \le u_{q-1} - e, \quad \forall a_j \in C_q \tag{2.16}$$

$$v_t \le u_h - e \tag{2.17}$$

$$e \ge 0 \tag{2.18}$$

$$w_{ip} \ge 0, \quad i = 1,2,...,n, \quad p = 1,2,...,b_i \tag{2.19}$$

$$x_{jk} \in \{0,1\}, \quad \forall a_j \in S - \{a_t\}, \quad k = 1,2,...,q \tag{2.20}$$

They propose the theorem that if $IP1_{a_t,h}$ is infeasible, then C_t^w is h.

If constraint (2.17) is changed with $v_t \geq u_{h-1}$, this IP is named as $IP2_{a_t,h}$ and checks if the best class of a_t is class h. They propose the second theorem if $IP2_{a_t,h}$ is infeasible, then C_t^b is h.

They suggest that LP relaxations of both $IP1_{a_t,h}$ and $IP2_{a_t,h}$ should be solved before solving IPs. Constraint (2.9), (2.10) and (2.13) are for the unassigned alternatives to be assigned to one of the classes while finding the worst or the best class for a_t. In the LP relaxation, these constraints are not written. If LP relaxation is infeasible, then IP is infeasible. It is easier and less time consuming to solve LP. If LP is feasible for a case, then IP which is thought to be more restrictive should be solved.

They propose an algorithm that finds the worst and the best classes of each alternative to be placed. They select an alternative. They solve $IP1_{a_t,h}$ starting from class 1 until it is infeasible for a class and find the worst class of the alternative. If $IP1_{a_t,h}$ is feasible for every h, then they decide that the worst class of alternative is class q. Then they solve $IP2_{a_t,h}$ starting from the worst class of the alternative until it is infeasible and find the best class of the alternative. If $IP2_{a_t,h}$ is feasible for every h, they decide that the best class of alternative is class 1. If best class and worst class of the alternative are same, then the alternative is assigned to this class. If they are different, it is asked to the DM to place the alternative to one of the classes between the worst and best class of the alternative. The algorithm continues until all alternatives are placed.

They tried the algorithm on the example problem they solved using UTADIS and they give the results based on the number of classes the alternatives are narrowed down to. In that problem, for 38% of alternatives the precise class is determined. 40% of the alternatives are narrowed down to 2 classes and 12% of the alternatives are asked to the DM for 3 classes.

2.3. A METHOD OF MULTIATTRIBUTE DECISION MAKING

Weber (1985) proposes a method for ranking a set of alternatives. In the method, he assumes an underlying piecewise additive utility function. To rank alternatives, he suggests a way to find ranges for utility differences of alternatives. After finding ranges, he calculates probability of an alternative being better than another alternative. He uses these probabilities to rank the alternatives. The methods used in Weber's article, HOPE and HOPIE, will be mentioned next.

2.3.1. HOPE Method

HOPE method ranks a set of alternatives that are evaluated in multiple criteria. To do this, it is suggested to ask the DM the utility value of a set of hypothetical alternatives. The method assumes an underlying additive utility model and piecewise linear marginal utility functions. These hypothetical alternatives come from an orthogonal design in which the interaction between factors is not considered. Each subinterval limit of the marginal utility function of each criterion is thought as a level in the orthogonal design concept. Each criterion value of each orthogonal alternative is on one of the limits of subintervals. Under the assumption that there is no interaction between criteria, an orthogonal array is constructed such that rows correspond to levels of an alternative. Number of orthogonal alternatives asked depends on the number of criteria and number of subintervals each criterion is divided. Let r correspond to the number of levels on each factor. Addelman and Kempthorne (1962) state that if an orthogonal array is constructed with no interaction between factors, the maximum number of factors that can be handled with r^n treatment combinations is $\frac{r^n - 1}{r - 1}$. In our case, factors can be thought as criteria, number of levels as number of subinterval limits and treatment combinations as alternatives. If there are 3 subinterval limits, $\frac{3^2 - 1}{3 - 1} = 4$ criteria can be handled at maximum with 3^2 alternatives and 13 criteria can be handled at maximum with 3^3 alternatives. Hedayat et. al. (1992)

explain how asymmetric orthogonal arrays where each factor has different number of levels are constructed.

Mathematical equations can be written on the utility of levels. Since it is an additive utility function, utility of each orthogonal alternative can be written as the sum of the utility of levels. The utility of each level is an unknown. Let there be n criteria and r levels on each criterion. Let u_{ip} be the utility of level p on criterion i and a_j denote the alternatives. h_j is the utility of alternative j. z_{ip} is the value corresponding to level p on criterion i. y_{ij} is the value of alternative j on criterion i. Since alternatives are defined by an orthogonal design, each y_{ij} value of orthogonal alternatives is equal to one of the z_{ip} values for each i. Then, the value of u_{ip} can be found by solving the following equation:

$$r \cdot u_{ip} = \left(\sum h_j \quad s.t. \quad y_{ij} = z_{ip}\right) - \left(\sum h_j \quad s.t. \quad y_{ij} = z_{i1}\right) \quad \forall i \quad and \quad \forall p \neq 1$$

Since it is assumed that the DM is able to give h_j values for orthogonal alternatives, then the value of each u_{ip} can be found. This equation can be written because of the orthogonality concept. When $h_j \quad s.t. \quad y_{i'j} = z_{i'j}$ are summed, we have $r \cdot u_{i'p}$ and the sum of u_{ip} for $i \neq i$ and $\forall p$. The second summation is done for the $p = 1$ on criterion i'. When we take the difference, we end up with $r \cdot \left(u_{i'p} - u_{i'1}\right)$.

Since we work on utility values, it can be scaled between 0 and 1. Then it is assumed that $u_{i1} = 0$ for $\forall i$ because they are the minimum values on each criterion i. Similarly $\sum_{i=1}^{i=n} u_{ir} = 1$ because u_{ir} corresponds to the maximum value on each criterion i. Therefore we can get the u_{ip} values for each i and p.

By solving the linear equations, one can get the utility values of the levels. This means the underlying utility model is completely found. Then the utility values of the actual alternatives are found by making linear interpolation on the utility values of the subinterval limits. Then ranking is done according to the calculated utility values for each alternative.

2.3.2. HOPIE Method

HOPE method asks the DM the utility values of a set of alternatives. Weber (1985) discusses that it is not an easy task for the DM to give the exact utility values of the alternatives and he suggests HOPIE method to get rid of this. The orthogonal design concept is again used. The utility ranges of the hypothetical orthogonal alternatives are asked to the DM. A linear program is written using the range of orthogonal alternatives. The constraints of LP are orthogonal to each other. Let the interval coming from the DM be $\left[h_j^-, h_j^+\right]$. The set of constraints used in HOPIE method is as follows:

$$r \cdot u_{ip} = \left(\sum h_j \quad s.t. \quad y_{ij} = z_{ij}\right) - \left(\sum h_j \quad s.t. \quad y_{ij} = z_{i1}\right), \quad \forall i, \quad \forall p \neq 1 \qquad (2.21)$$

$$h_j \geq h_j^-, \quad \forall j \qquad (2.22)$$

$$h_j \leq h_j^+, \quad \forall j \qquad (2.23)$$

$$h_j - h_{j'} \geq 0 \quad if \quad a_j \ \mathbf{f} \ a_{j'} \qquad (2.24)$$

$$u_{ip} - u_{i(p-1)} \geq 0, \quad \forall i, \quad j = 2,3,...,r \qquad (2.25)$$

$$\sum_{i=1}^{i=n} u_{ir} = 1 \qquad (2.26)$$

$$u_{i1} = 0, \quad \forall i \qquad (2.27)$$

The utility of each level and utility of orthogonal alternatives are decision variables and constraint (2.21) provides mathematical equations on the decision variables with the same idea of HOPE. Constraint (2.22) and (2.23) set bounds on the utility of orthogonal alternatives. The utility range given by the DM for

each alternative is written as constraint. Constraint (2.24) is written if the DM has given any preference information on the alternatives.

Two LPs are solved for each pair of alternatives which are to be ranked subject to constraints $(2.21)-(2.27)$. The objective of the first LP is to minimize the difference of the utility values of two alternatives and the objective of the second LP is to maximize the difference of the utility values of two alternatives. If the number of alternatives to be ranked is m, then $m\cdot(m-1)$ LPs should be solved.

By solving these LPs, preference information can be obtained. For two alternatives a_t and a_l, if $\min(h_t - h_l) \geq 0$, then a_t is a better alternative than a_l. If $\max(h_t - h_l) < 0$, then a_t is a worse alternative than a_l.

If the calculated range for utility difference of two alternatives does not include 0, information about which alternative is better is obtained. If it includes 0, then the information of which alternative is better cannot be obtained. For the situations which an exact result cannot be obtained, Weber (1985) suggests calculating probability of an alternative being better than another alternative. It is assumed that the distribution of the difference of utility values can be approximated by symmetric triangular distribution within the range calculated.

Weber (1985) states that this approximation is based on the assumption that the individual utility values of alternatives follow a uniform distribution between the interval $\left[h_j^-, h_j^+\right]$. The distribution of the difference of utility of two alternatives can be approximated by the convolution of distributions on the intervals $\left[h_j^-, h_j^+\right]$.

According to Weber (1985), probability of a_t being better than a_l can be approximated by symmetric triangular distribution. Let $a = \min(h_t - h_l)$,

$b = \max(h_t - h_l)$ and $p = P[(h_t - h_l) \geq 0]$. It can be found from the following formula:

$$p = \begin{cases} 1 \quad if \quad a > 0 \\ 1 - 2\dfrac{a^2}{(b-a)^2} \quad if \quad a \leq 0, \quad \dfrac{(a+b)}{2} > 0 \\ 2\dfrac{b^2}{(b-a)^2} \quad if \quad \dfrac{(a+b)}{2} \leq 0, \quad b > 0 \\ 0 \quad if \quad b \leq 0 \end{cases}$$

The calculated probabilities are thought as a measure of strength of preferences. Weber (1985) suggests a ranking in accordance with the calculated probabilities and tries to obtain “a most similar” ranking.

CHAPTER 3

AN INTERACTIVE PROBABILISTIC MULTI-CRITERIA SORTING ALGORITHM

3.1. OVERVIEW OF THE ALGORITHM

We aim to classify a set A of m alternatives $a_1, a_2, ..., a_m$ evaluated using n criteria into one of the q classes $C_1, C_2, ..., C_q$ where C_1 is the most and C_q is the least preferred class. We assume an underlying additive utility function. Additive utility functions can represent a quite general behavior. The marginal utility of alternatives on each criterion is thought to be piecewise linear. A is the set of all alternatives. Let C_0 be the set of alternatives whose classes are not known and C_r be the set of alternatives whose classes are known. $A = C_r \cup C_0$. Using the class information of alternatives in C_r, we make inferences about the classes of alternatives in C_0. We classify alternatives in an iterative way. We start using class information given to us. For each alternative, we calculate probability of belonging each class using available class information. After calculating probabilities, we try to assign alternatives to classes either exactly or probabilistically. Probabilistic classification is done by comparing probabilities with a given threshold value. If we cannot classify any alternatives with the available information, we ask the DM to place an alternative. With this new information, we calculate new probabilities and try to assign alternatives in C_0 to classes. This procedure continues until all the alternatives in C_0 are assigned to one of the classes.

We propose two different approaches. One is to compare the unassigned alternatives with readily assigned alternatives. We find probabilities for each unassigned alternative being better than readily assigned ones. The other is to compare unassigned alternatives with utility thresholds and calculate probabilities for each unassigned alternative being better than each utility threshold.

3.2. THE ALGORITHM WITH PAIRWISE COMPARISON OF ALTERNATIVES

A probabilistic classification is suggested by using the probability of an alternative being better than an alternative whose class is known. This algorithm compares the utility of each alternative to be classified with the utility values of already assigned alternatives. We find the minimum and the maximum values of utility differences of each unassigned alternative and each assigned one. We find ranges for the differences and calculate the probability of each alternative being better than an assigned alternative. Alternatives are classified by comparing probabilities with a given probability threshold according to some conditions which will be defined later. Alternatives that satisfy the conditions will be assigned to classes. If no new alternative can be classified with the available information, the DM is asked to place a chosen alternative to its correct class. In the light of the new information, we will find ranges, calculate probabilities and try to assign the remaining alternatives to classes. Same procedure will be applied until all the alternatives are classified.

3.2.1. Linear Programs for Utility Ranges

In our algorithm, we assume an underlying additive utility function similar to UTADIS. UTADIS estimates the parameters of the underlying utility function. However, our algorithm does not require making such estimations. We work with different objectives instead of minimizing the sum of misclassification errors.

Any misclassification of the alternatives in C_r is not allowed in the models we used. We do not let violation of any class constraints.

For each alternative in C_0, we want to find the minimum and the maximum values of its utility difference with each alternative in C_r using available class information. We use the notation presented in UTADIS.

Let j be the index for alternatives where $j = 1,2,...,m$, i be the index for criteria where $i = 1,2,...,n$, k be the index for classes where $k = 1,2,...,q$ and p be the index for subintervals where $p = 1,2,...,b_i$.

Let $g_i(a_j)$ be the score of alternative j on criterion i. Then $U[g(a_j)]$ is the global utility of alternative j where $g(a_j) = [g_1(a_j), g_2(a_j),..., g_n(a_j)]$. Let $u_i[g_i(a_j)]$ be the marginal utility of alternative j on criterion i. Each criterion range is divided into b_i subintervals $[g_i{}^p, g_i{}^{p+1}]$, $p = 1,2,...,b_i$.

Let subinterval p have an unknown utility value of w_{ip} on criterion i where $w_{ip} = u_i[g_i{}^p] - u_i[g_i{}^{p+1}]$. If $g_i{}^{r_{ji}} \le g_i(a_j) \le g_i{}^{r_{ji}+1}$ for some $1 \le r_{ji} \le b_i$, then $U[g(a_j)] = \sum_{i=1}^{n}\left[\sum_{p=1}^{r_{ji}-1} w_{ip} + \frac{(g_i(a_j) - g_i{}^{r_{ji}})}{(g_i{}^{r_{ji}+1} - g_i{}^{r_{ji}})} w_{ir_{ji}}\right]$.

Let $a_l \in C_r$. Let the following constraints define the constraint set S:

$$U[g(a_j)] = \sum_{i=1}^{n}\left[\sum_{p=1}^{r_{ji}-1} w_{ip} + \frac{\left(g_i(a_j) - g_i^{\,r_{ji}}\right)}{\left(g_i^{\,r_{ji}+1} - g_i^{\,r_{ji}}\right)} w_{ir_{ji}}\right], \quad \forall a_j \in A \tag{3.1}$$

$$U[g(a_l)] - u_k \geq 0, \quad \forall a_l \in C_k \quad (k = 1,2,...,q-1) \tag{3.2}$$

$$U[g(a_l)] - u_{k-1} \leq -d, \quad \forall a_l \in C_k \quad (k = 2,3,...,q) \tag{3.3}$$

$$\sum_{i=1}^{n}\sum_{p=1}^{b_i-1} w_{ip} = 1, \quad \forall i = 1,2,...,n, \quad p = 1,2,...,b_i \tag{3.4}$$

$$u_k - u_{k+1} \geq s, \quad \forall k = 1,2,...,q-2 \tag{3.5}$$

$$w_{ip} \geq 0, \quad \forall i = 1,2,...,n, \quad p = 1,2,...,b_i \tag{3.6}$$

d and s are user defined constants. In our calculations, we take d as $1x10^{-6}$ and s as $1x10^{-3}$. Constraint (3.1) expresses utility of alternatives that are members of C_r in terms of w_{ip} values. Constraint (3.2) indicates that if an alternative is assigned to C_k, then the utility of it should be greater than or equal to the utility threshold u_k that is the lower limit of class k. Constraint (3.3) indicates that utility of an alternative assigned to C_k should be smaller than the utility threshold u_{k-1} that is the upper limit of class k. Constraint (3.4) expresses that sum of utility values of all subintervals should be equal to 1. Constraint (3.5) provides that the upper limit of a class is greater than the lower limit of that class. Constraint (3.6) indicates that utility values of subintervals should be nonnegative.

Let $a_t \in C_0$. To compare a_t with a_l, we solve the following models:

$LP_{3.1}^{\,t,l}$

$Min \quad U[g(a_t)] - U[g(a_l)]$

s.to

$(3.1) - (3.6)$

$LP_{3.2}{}^{t,l}$

$Max \quad U[g(a_t)] - U[g(a_l)]$

s.to

$(3.1) - (3.6)$

These models should be solved for each alternative $a_t \in C_0$ and for each $a_l \in C_r$. Let the optimal objective function value of $LP_{3.1}{}^{t,l}$ for a_t and a_l be $obj^*_{3.1}(t,l)$ and the optimal objective function value of $LP_{3.2}{}^{t,l}$ for a_t and a_l be $obj^*_{3.2}(t,l)$.

If $obj^*_{3.1}(t,l) \geq 0$ for $a_l \in C_{k'}$, then $obj^*_{3.1}(t,l) \geq 0$ for $a_l \in C_k$ where $k = k'+1, k'+2, ..., q$.

If $obj^*_{3.1}(t,l) \geq 0$, then $obj^*_{3.2}(t,l) \geq 0$.

Obtaining $obj^*_{3.1}(t,l) \geq 0$ for any $a_l \in C_r$, we know that a_t is a better alternative than a_l so the worst class of a_t is the class of a_l. Then there is no need to solve $LP_{3.1}{}^{t,l}$ for the remaining alternatives in C_r which are in the same class with a_l and which are in a worse class than the class of a_l. There is also no need to solve $LP_{3.2}{}^{t,l}$ for the alternatives which are in the same class with a_l and which are in a worse class than the class of a_l.

If $obj^*_{3.2}(t,l) < 0$ for $a_l \in C_{k'}$, then $obj^*_{3.2}(t,l) < 0$ for $a_l \in C_k$ where $k = 1, 2, ..., k'-1$.

If $obj^*_{3.2}(t,l) < 0$, then $obj^*_{3.1}(t,l) < 0$.

Obtaining $obj^*_{3.2}(t,l) < 0$ for any $a_l \in C_r$ means that a_t is a worse alternative than a_l so the best class of a_t is the class of a_l. Therefore, there is no need solve

$LP_{3.2}^{t,l}$ for remaining alternatives in C_r which are in the same class with a_l and which are in a better class than the class of a_l. There is also no need to solve $LP_{3.1}^{t,l}$ for the alternatives which are in the same class with a_l and which are in a better class than the class of a_l.

Let $|C_0| = m_o$ and $|C_r| = m_r$. Then the maximum number of linear programs to be solved is $2 \cdot m_o \cdot m_r$.

3.2.2. Probability Calculation

In our study, we assume that the DM has an underlying additive utility function and utility thresholds that separate the classes. The DM classifies the alternatives by using this utility function and utility thresholds. We do not know the actual parameters of the DM. If we make an exact classification by using our algorithm, we are able to present the DM the original classification. When we make probabilistic classification, we want to make a classification close to original classification of the DM. We do not know the actual difference found by using the parameters of the underlying model of the DM but we have the information of the minimum and maximum value the difference can take. We try to fit appropriate probability distributions to the difference. We first assume that the difference of the utility values of alternatives follows uniform distribution within the range $\left[obj_{3.1}^{*}(t,l), obj_{3.2}^{*}(t,l)\right]$. Assuming uniform distribution implies that actual difference can be each of the possible values in the found range with equal probability. We wonder if actual values are more likely to be some of the values in the range. We look if we can work with different probability distributions. We try to fit a symmetric triangular distribution to the difference. After this, we work on the distribution of utility values of subintervals to find a distribution to the difference. We assume that the utility values of subintervals follow uniform

distribution within their possible range. Under this assumption, we fit a normal distribution to the difference.

We use symmetric distributions because the only information we know about the difference is the range. We do not have any information about the difference being more likely on the right or left side of the range.

Uniform Distribution

We assume that the difference of utility of alternatives has a uniform distribution between the calculated the minimum and maximum value. Utility difference follows $U\left[obj_{3.1}^{*}(t,l), obj_{3.2}^{*}(t,l)\right]$ for $a_t \in C_0$ and $a_l \in C_r$.

What we need in our model is $P[U[g(a_t)] - U[g(a_l)] \geq 0]$. Let $a = obj_{3.1}^{*}(t,l)$ and $b = obj_{3.2}^{*}(t,l)$.

Let $p(t,l) = P[U[g(a_t)] - U[g(a_l)] \geq 0]$ and then $p(t,l)$ is found as follows:

$$p(t,l) = \begin{cases} 1 \quad if \quad a > 0 \\ \dfrac{(b-0)}{(b-a)} \quad if \quad a \leq 0, \quad b > 0 \\ 0 \quad if \quad b \leq 0 \end{cases}$$

Symmetric Triangular Distribution

We can think that the utility value of each alternative follows a uniform distribution. We can find the minimum and the maximum values of them subject to constraint set S. Then we can work on the distribution of the difference of two uniform random variables. We know that the difference of two independent uniform random variables either follow a symmetrical triangular distribution or a trapezoid distribution. Let random variable X_1 follow $U[\min_1, \max_1]$ and X_2 follow $U[\min_2, \max_2]$. If the widths of the ranges of uniform random variables

are same, then the difference $X_1 - X_2$ follows a symmetric triangular distribution within the range $[\min_1 - \max_2, \max_1 - \min_2]$. If they are not same, the difference follows a symmetrical trapezoid distribution within the range $[\min_1 - \max_2, \max_1 - \min_2]$.

We can use these distributions for the differences if the uniform random variables are independent. In our case, the utility of alternatives are not independent from each other. Weber (1985) approximates the utility difference of alternatives with symmetrical triangular distribution within the range. Since they are not independent of each other, assuming the difference follows a symmetrical triangular distribution will be only an approximation. This implies that actual value of difference is more likely to be close to the middle of the interval.

If we assume that the difference of the utility values of alternatives follow a symmetric triangular distribution, probabilities will be calculated in the following way:

Let $a = obj^*_{3.1}(t,l)$, $b = obj^*_{3.2}(t,l)$ and $p(t,l) = P[U[g(a_t)] - U[g(a_l)] \geq 0]$. Then $p(t,l)$ is found as follows:

$$p(t,l) = \begin{cases} 1 \quad if \quad a > 0 \\ 1 - 2\dfrac{a^2}{(b-a)^2} \quad if \quad a \leq 0, \quad \dfrac{(a+b)}{2} > 0 \\ 2\dfrac{b^2}{(b-a)^2} \quad if \quad \dfrac{(a+b)}{2} \leq 0, \quad b > 0 \\ 0 \quad if \quad b \leq 0 \end{cases}$$

Normal Distribution

We try to fit a normal distribution to the differences of utility values of alternatives. To justify this, we worked on the distribution of the utility values of subintervals and used them to find the distribution of utility values of alternatives.

Recall that $U[g(a_j)] = \sum_{i=1}^{n} \left[\sum_{p=1}^{r_{ji}-1} w_{ip} + \frac{\left(g_i(a_j) - g_i^{r_{ii}}\right)}{\left(g_i^{r_{ii}+1} - g_i^{r_{ii}}\right)} w_{ir_{ji}} \right]$.

From the above equation, it is understood that the marginal utility of each alternative can be written as the sum of w_{ip} values for the first $(r_{ji} - 1)$ subintervals and $w_{ir_{ji}}$ multiplied by the coefficient found by the linear interpolation of $g_i(a_j)$ between $g_i^{r_{ii}}$ and $g_i^{r_{ii}+1}$. Let $z_{ip}(a_j)$ denote the coefficient of w_{ip} for alternative j while expressing $U[g(a_j)]$. Then we can write the following equation:

$$U[g(a_j)] = \sum_{i=1}^{n} \sum_{p=1}^{b_i} z_{ip}(a_j) \cdot w_{ip}$$

where

$$z_{ip}(a_j) = \begin{cases} 1 & if \quad p < r_{ji} \\ \dfrac{\left(g_i(a_j) - g_i^{r_{ii}}\right)}{\left(g_i^{r_{ii}+1} - g_i^{r_{ii}}\right)} & if \quad p = r_{ji} \\ 0 & if \quad p > r_{ji} \end{cases}$$

Distribution of Utility Values of Subintervals

We work with the probability distributions of the utility values of subintervals. Using the distribution of each w_{ip}, we will find a probability distribution for each $U[g(a_j)]$. We can find the minimum and maximum value each w_{ip} can take subject to constraint set S. For this aim, we need to solve the following models for $i = 1,2,...,n$ and $p = 1,2,...,b_i$.

$LP_{3.3}{}^{i,p}$

$Min \quad w_{ip}$

s.to

$(3.1)-(3.6)$

$LP_{3.4}{}^{i,p}$

$Max \quad w_{ip}$

s.to

$(3.1)-(3.6)$

Let $obj^*_{3.3}(i,p)$ be the optimal objective value of $LP_{3.3}{}^{i,p}$ and $obj^*_{3.4}(i,p)$ be the optimal objective value of $LP_{3.4}{}^{i,p}$. We need to solve $2\cdot\sum_{i=1}^{n} b_i$ LPs to find the range of all w_{ip} values. Each w_{ip} can take any value between $obj^*_{3.3}(i,p)$ and $obj^*_{3.4}(i,p)$. The only information we have for each w_{ip} is its range and we will assume that each w_{ip} follows uniform distribution within $obj^*_{3.3}(i,p)$ and $obj^*_{3.4}(i,p)$. Let $m(w_{ip})$ be the expected value of w_{ip} and $s(w_{ip})$ be the standard deviation of w_{ip}. Then;

$$E[w_{ip}] = m(w_{ip}) = \frac{\left(obj^*_{3.3}(i,p) + obj^*_{3.4}(i,p)\right)}{2}$$

$$V[w_{ip}] = s^2(w_{ip}) = \frac{\left(obj^*_{3.4}(i,p) - obj^*_{3.3}(i,p)\right)^2}{12}.$$

Distribution of Utility Values of Alternatives

We need the distribution of utility values of alternatives. We assume each w_{ip} is a random variable. Then each $U[g(a_j)]$ will be a random variable since it is a linear combination of random variables. We assume that w_{ip} values are independent of

each other. This means that the utility of each subinterval is independent from the others.

Each $U[g(a_j)]$ will be the sum of at most $\sum_{i=1}^{n} b_i$ uniform random variables. We use Central Limit Theorem and approximate the distribution of sum of m independent uniform variables by a normal distribution even for small values of m. By using this information, we assume that $U[g(a_j)]$ will follow a normal distribution. Let $m(a_j)$ be the expected value of $U[g(a_j)]$ and $s(a_j)$ be the standard deviation of $U[g(a_j)]$. $m(a_j)$ and $s^2(a_j)$ can be calculated as follows:

$$E\left[U[g(a_j)]\right] = m(a_j) = E\left[\sum_{i=1}^{n}\sum_{p=1}^{b_i} z_{ip}(a_j)\cdot w_{ip}\right] = \sum_{i=1}^{n}\sum_{p=1}^{b_i} z_{ip}(a_j)\cdot E[w_{ip}]$$
$$= \sum_{i=1}^{n}\sum_{p=1}^{b_i} z_{ip}(a_j)\cdot m(w_{ip})$$

Since we assume w_{ip} values are independent of each other,

$$V\left[U[g(a_j)]\right] = s^2(a_j) = V\left[\sum_{i=1}^{n}\sum_{p=1}^{b_i} z_{ip}(a_j)\cdot w_{ip}\right] = \sum_{i=1}^{n}\sum_{p=1}^{b_i} (z_{ip}(a_j))^2\cdot V[w_{ip}]$$
$$= \sum_{i=1}^{n}\sum_{p=1}^{b_i} (z_{ip}(a_j))^2\cdot s^2(w_{ip})$$

Distribution of the Difference of the Utility of Alternatives

We need the distribution of the difference of each $U[g(a_t)]$, $a_t \in C_0$ from each $U[g(a_l)]$, $a_l \in C_r$. We found that both $U[g(a_t)]$ and $U[g(a_l)]$ follow a normal distribution. Let $d_{tl} = U[g(a_t)] - U[g(a_l)]$.

$$d_{tl} = \sum_{i=1}^{n}\sum_{p=1}^{b_i} z_{ip}(a_t)\cdot w_{ip} - \sum_{i=1}^{n}\sum_{p=1}^{b_i} z_{ip}(a_l)\cdot w_{ip} = \sum_{i=1}^{n}\sum_{p=1}^{b_i}\left[z_{ip}(a_t) - z_{ip}(a_l)\right]\cdot w_{ip}$$

Let $m(d_{tl})$ be the expected value of d_{tl} and $s(d_{tl})$ be the standard deviation of d_{tl}. $m(d_{tl})$ and $s^2(d_{tl})$ can be calculated as follows::

$$E[d_{tl}] = m(d_{tl}) = E\left[\sum_{i=1}^{n}\sum_{p=1}^{b_i}\left[z_{ip}(a_t) - z_{ip}(a_l)\right]\cdot w_{ip}\right] = \sum_{i=1}^{n}\sum_{p=1}^{b_i}\left[z_{ip}(a_t) - z_{ip}(a_l)\right]\cdot m(w_{ip})$$

$$V[d_{tl}] = s^2(d_{tl}) = V\left[\sum_{i=1}^{n}\sum_{p=1}^{b_i}\left[z_{ip}(a_t) - z_{ip}(a_l)\right]\cdot w_{ip}\right] = \sum_{i=1}^{n}\sum_{p=1}^{b_i}\left[z_{ip}(a_t) - z_{ip}(a_l)\right]^2\cdot s^2(w_{ip})$$

We found that d_{tl} follows $N\left[m(d_{tl}), s^2(d_{tl})\right]$. We have a constraint on the sum of w_{ip} values. For our problem, $\sum_{i=1}^{n}\sum_{p=1}^{b_i} w_{ip} = 1$. We need to find the distribution of the difference under this condition. Let $Y = \sum_{i=1}^{n}\sum_{p=1}^{b_i} w_{ip}$, then Y is the sum of $\sum_{i=1}^{n} b_i$ uniform random variables. We need to find the joint distribution of d_{tl} and Y to find the conditional distribution of the difference. Let $m(Y)$ be the expected value of Y and $s(Y)$ be standard deviation of Y. $m(Y)$ and $s^2(Y)$ are calculated in a similar way.

$$E[Y] = m(Y) = E\left[\sum_{i=1}^{n}\sum_{p=1}^{b_i} w_{ip}\right] = \sum_{i=1}^{n}\sum_{p=1}^{b_i} E[w_{ip}] = \sum_{i=1}^{n}\sum_{p=1}^{b_i} m(w_{ip})$$

$$V[Y] = s^2(Y) = V\left[\sum_{i=1}^{n}\sum_{p=1}^{b_i} w_{ip}\right] = \sum_{i=1}^{n}\sum_{p=1}^{b_i} V[w_{ip}] = \sum_{i=1}^{n}\sum_{p=1}^{b_i} s^2(w_{ip})$$

We found that d_{tl} follows $N\left[m(d_{tl}), s^2(d_{tl})\right]$ and Y follows $N\left[m(Y), s^2(Y)\right]$ approximately. d_{tl} and Y are not independent from each other since they are

different linear combinations of the same random variables. The joint distribution of the two dependent normally distributed random variables d_{tl} and Y can be defined as a bivariate normal distribution (Hines et. al. p. 160). We know the means and the variances of d_{tl} and Y but we need to find the correlation coefficient r to define the bivariate normal distribution. Let us denote d_{tl} as D, $m(d_{tl})$ as $m(D)$ and $s(d_{tl})$ as $s(D)$. Let $Cov(D,Y)$ denote the covariance between D and Y. r can be found from the following equation.

$$r = \frac{Cov(D,Y)}{s(D)s(Y)}$$

We know $s(D)$ and $s(Y)$. We need to find $Cov(D,Y)$.

$$Cov(D,Y) = E[(D - m(D))(Y - m(Y))]$$
$$Cov(D,Y) = E[DY] - m(D)E[Y] - E[D]m(Y) + m(D)m(Y)$$
$$Cov(D,Y) = E[DY] - m(D)m(Y)$$

We need to find $E[DY]$ to find $Cov(D,Y)$.

$$E[DY] = E\left[\left(\sum_{i=1}^{n}\sum_{p=1}^{b_i}\left[z_{ip}(a_t) - z_{ip}(a_l)\right]\cdot w_{ip}\right)\left(\sum_{i=1}^{n}\sum_{p=1}^{b_i} w_{ip}\right)\right]$$
$$E[DY] = E\left[\left(\sum_{i=1}^{n}\sum_{p=1}^{b_i}\sum_{i'=1}^{n}\sum_{p'=1}^{b_{i'}}\left[z_{ip}(a_t) - z_{ip}(a_l)\right]\cdot w_{ip}\cdot w_{i'p'}\right)\right]$$
$$E[DY] = \left(\sum_{i=1}^{n}\sum_{p=1}^{b_i}\sum_{i'=1}^{n}\sum_{p'=1}^{b_{i'}}\left[z_{ip}(a_t) - z_{ip}(a_l)\right]\cdot E\left[w_{ip}\cdot w_{i'p'}\right]\right)$$

Since we assume w_{ip} values are independent of others, $E[w_{ip}\cdot w_{i'p'}]$ can be written as $E[w_{ip}]\cdot E[w_{i'p'}]$.

$$E[DY] = \sum_{i=1}^{n}\sum_{p=1}^{b_i}\left[z_{ip}(a_t) - z_{ip}(a_l)\right]\cdot E\left[w_{ip}{}^2\right]$$
$$+ \sum_{i=1}^{n}\sum_{p=1}^{b_i}\sum_{i'=1}^{n}\sum_{p'=1}^{b_{i'}}\left[z_{ip}(a_t) - z_{ip}(a_l)\right]\cdot E\left[w_{ip}\right]\cdot E\left[w_{i'p'}\right] - \sum_{i=1}^{n}\sum_{p=1}^{b_i}\left[z_{ip}(a_t) - z_{ip}(a_l)\right]\cdot E\left[w_{ip}\right]^2$$

$$E[DY] = \sum_{i=1}^{n}\sum_{p=1}^{b_i}\left[z_{ip}(a_t) - z_{ip}(a_l)\right]\cdot \left(V\left[w_{ip}\right] + E\left[w_{ip}\right]^2\right)$$
$$+ \sum_{i=1}^{n}\sum_{p=1}^{b_i}\sum_{i'=1}^{n}\sum_{p'=1}^{b_{i'}}\left[z_{ip}(a_t) - z_{ip}(a_l)\right]\cdot E\left[w_{ip}\right]\cdot E\left[w_{i'p'}\right] - \sum_{i=1}^{n}\sum_{p=1}^{b_i}\left[z_{ip}(a_t) - z_{ip}(a_l)\right]\cdot E\left[w_{ip}\right]^2$$

$$E[DY] = \sum_{i=1}^{n}\sum_{p=1}^{b_i}\left[z_{ip}(a_t) - z_{ip}(a_l)\right]\cdot \left(s^2\left(w_{ip}\right) + m^2\left(w_{ip}\right)\right)$$
$$+ \sum_{i=1}^{n}\sum_{p=1}^{b_i}\sum_{i'=1}^{n}\sum_{p'=1}^{b_{i'}}\left[z_{ip}(a_t) - z_{ip}(a_l)\right]\cdot m\left(w_{ip}\right)\cdot m\left(w_{i'p'}\right) - \sum_{i=1}^{n}\sum_{p=1}^{b_i}\left[z_{ip}(a_t) - z_{ip}(a_l)\right]\cdot m\left[w_{ip}\right]^2$$

After finding $E[DY]$, we can write $Cov(D,Y)$ and r.

$$Cov(D,Y) = E[DY] - m(D)\cdot m(Y)$$

$$r = \frac{Cov(D,Y)}{s(D)s(Y)}$$

The joint distribution of D and Y follows a bivariate normal distribution with parameters $m(D)$, $s^2(D)$, $m(Y)$, $s^2(Y)$, r.

We scale the utility of alternatives between 0 and 1. Y can be thought as the utility of an alternative that has the maximum scores in all the criteria. In our case, Y is equal to 1. Therefore, we need the distribution of utility of each alternative given that Y is equal to 1.

Conditional distribution of bivariate normally distributed random variables follows normal distribution (Hines et. al. p. 162). For a particular value of Y, D will follow normal distribution with parameters $E[D|Y=y]$ and $V[D|Y=y]$.

$$E[D|Y=y] = m(D) - r\,\frac{s(D)}{s(Y)}(y - m(Y))$$

$$V[D|Y=y]=(1-r^2)s^2(Y)$$

As it is seen, $V[D|Y=y]$ does not depend on the value of y. For our case, $y=1$. Let d_{tl}' denote $D|Y=1$. Let $m(d_{tl}')$ be the expected value of $D|Y=1$ and $s(d_{tl}')$ be standard deviation of $D|Y=1$.

$$m(d_{tl}')=E[D|Y=1]=m(D)-r\frac{s(D)}{s(Y)}(1-m(Y))$$

$$s^2(d_{tl}')=V[D|Y=1]=(1-r^2)s^2(Y)$$

Probability Calculation

We need the probabilities of $a_t \in C_0$ being better than each $a_l \in C_r$. We found the parameters of the distribution and we can calculate the probabilities. However, we have more information about the difference. We have found the minimum and maximum value the difference can take. Outside the found range, the value of the difference is not feasible subject to constraint set S. We will work on the part of the assumed distribution which is feasible subject to constraint set S.

Let $p(t,l)=P\left[d_{tl}'\geq 0 \middle| obj_{3.1}^*(t,l)\leq d_{tl}'\leq obj_{3.2}^*(t,l)\right]$, $p(t,l)$ is found as follows:

$$p(t,l)=\frac{P\left[(d_{tl}'\geq 0)\cap\left(obj_{3.1}^*(t,l)\leq d_{tl}'\leq obj_{3.2}^*(t,l)\right)\right]}{P\left[obj_{3.1}^*(t,l)\leq d_{tl}'\leq obj_{3.2}^*(t,l)\right]}$$

If $obj_{3.1}^*(t,l)\geq 0$, the intersection is the universal set and all possible values of d_{tl}' are greater than 0 and $p(t,l)=1$.

If $obj_{3.1}^*(t,l)<0$ and $obj_{3.2}^*(t,l)\geq 0$,

$$p(t,l)=\frac{P\left[\left(d_{tl}'\geq 0\right)\cap\left(obj_{3.1}^{*}(t,l)\leq d_{tl}'\leq obj_{3.2}^{*}(t,l)\right)\right]}{P\left[obj_{3.1}^{*}(t,l)\leq d_{tl}'\leq obj_{3.2}^{*}(t,l)\right]}=\frac{P\left[0\leq d_{tl}'\leq obj_{3.2}^{*}(t,l)\right]}{P\left[obj_{3.1}^{*}(t,l)\leq d_{tl}'\leq obj_{3.2}^{*}(t,l)\right]}$$

$$p(t,l)=\frac{f\left(\frac{obj_{3.2}^{*}(t,l)-m(d_{tl}')}{s(d_{tl}')}\right)-f\left(\frac{0-m(d_{tl}')}{s(d_{tl}')}\right)}{f\left(\frac{obj_{3.2}^{*}(t,l)-m(d_{tl}')}{s(d_{tl}')}\right)-f\left(\frac{obj_{3.1}^{*}(t,l)-m(d_{tl}')}{s(d_{tl}')}\right)}$$

If $obj_{3.2}^{*}(t,l)<0$, the intersection is an empty set and there does not exist any possible value of d_{tl} that is greater than 0 and $p(t,l)=0$. In summary,

$$p(t,l)=\begin{cases}1 & if \quad obj_{3.1}^{*}(t,l)\geq 0\\ \dfrac{f\left(\frac{obj_{3.2}^{*}(t,l)-m(d_{tl}')}{s(d_{tl}')}\right)-f\left(\frac{0-m(d_{tl}')}{s(d_{tl}')}\right)}{f\left(\frac{obj_{3.2}^{*}(t,l)-m(d_{tl}')}{s(d_{tl}')}\right)-f\left(\frac{obj_{3.1}^{*}(t,l)-m(d_{tl}')}{s(d_{tl}')}\right)} & if \quad obj_{3.1}^{*}(t,l)<0,\quad obj_{3.2}^{*}(t,l)\geq 0\\ 0 & if \quad obj_{3.2}^{*}(t,l)<0\end{cases}$$

Grouping Probabilities

Probabilities are calculated for each alternative in C_0 against each alternative in C_r. The calculated probabilities are grouped according to the classes of alternatives in C_r. For each alternative in C_0, probabilities of being better than the alternatives in C_r which are in class k are grouped together and the minimum and maximum of them are selected.

Let $\max_k(t)$ denote the maximum probability of a_t being better than any alternative in class k and $\min_k(t)$ denote the minimum probability of a_t being better than any alternative in class k.

$$\max_k(t) = \max_{a_l \in C_k} \quad p(t,l)$$

$$\min_k(t) = \min_{a_l \in C_k} \quad p(t,l)$$

3.2.3. The Classification of Alternatives

We make the classification by comparing $\min_k(t)$ and $\max_k(t)$ with a given threshold on probabilities for each a_t. Let th denote the threshold on probabilities.

Exact Classification

If we set th as 0, then we will make an exact classification. Since all the alternatives are either correctly classified or asked to the DM, the algorithm works with exact information.

If $\max_k(t) = 1$, then a_t is better than an alternative in class k and there does not exist any set of parameters that makes a_t worse than that alternative in class k, Therefore, the worst class of a_t is k.

If $\min_k(t) = 0$, then a_t is worse than an alternative in class k and there does not exist any set of parameters that makes a_t better than that alternative in class k, Therefore the best class of a_t is k.

With these probabilities we can make an exact classification if the following conditions are satisfied:

If $\max_1(t) = 1$, then a_t is in C_1.
If $\max_k(t) = 1$ and $\min_k(t) = 0$, then a_t is in C_k.

If $\min_q(t) = 0$, then a_t is in C_q.

Probabilistic Classification

If $th > 0$, then we make a probabilistic classification. We assign alternatives to classes even though we do not have exact information about their classes. The assignment of alternatives to the classes is done in the following way:

If $\max_k(t) \geq (1-th)$, then a_t is better than an alternative in class k with at least $(1-th)$ probability.
If $\min_k(t) \leq th$, then a_t is worse than an alternative in class k with at least $(1-th)$ probability.

With these probabilities we can make a probabilistic classification if the following conditions are satisfied:

If $\max_1(t) \geq (1-th)$, then a_t is assigned to C_1.
If $\max_k(t) \geq (1-th)$ and $\min_k(t) \leq th$, then a_t is assigned to C_k.
If $\min_q(t) \leq th$, then a_t is assigned to C_q.

With the calculated probabilities, the alternatives in set C_0 which satisfy one of the above conditions are classified. If no assignments are done with the available information, then the DM is asked to place an alternative to a class. With the new class information taken from the DM, new ranges and probabilities are calculated and assignments are done accordingly. Each time, narrower ranges are calculated for each alternative because linear programs are solved for the same objectives with more constraints. This procedure will repeat until all the alternatives are assigned to a class. Each time after the probabilities are calculated, either one of the mentioned conditions is satisfied and at least one alternative is classified or the DM is asked to place an alternative.

3.2.4. Selection of the Alternative to Ask the DM

To select the alternative to be asked to the DM, the following procedure is applied. Let a_t^s denote the selected alternative to be asked to the DM. For each $a_t \in C_0$,

$d_1(t) = \min_k \left(\min_k(t) - \max_{k-1}(t)\right)$

$a_t^s = \arg\min_{a_t \in C_0} d_1(t)$ is selected to be asked to the DM.

This aims to ask the alternatives which are thought to be close to utility thresholds that separate the classes. It is known that $\min_k(t) > \max_{k-1}(t)$ since the worst alternative of C_{k-1} has a greater utility than the best alternative of C_k. We think that class information of such alternatives will give us more restrictive constraints than the other alternatives and give us narrower utility ranges.

When we select the alternative to present to the DM for placement, we have some information about the class of the alternative. Although the algorithm is not able to classify a_t^s exactly, it can narrow down the number of classes it can be placed. According to the available information, it will be infeasible for a_t^s to be placed in some classes. Therefore, while asking the DM, we can provide fewer classes to select from.

If $\max_{k'}(t^s) = 1$, then a_t^s cannot be placed to a worse class than k'. If $\max_{k'}(t^s) = 1$ and $\max_{k'-1}(t^s) < 1$, then k' will be the worst class a_t^s can be placed. If $\min_{k''}(t^s) = 0$, then a_t^s cannot be placed to a better class than k''. If $0 < \min_{k''+1}(t^s)$, then k'' will be the best class a_t^s can be placed. This means a_t^s can be placed to classes $k', k'+1, ..., k''$. The DM can decide between $k''-k'+1$ classes instead of q classes to place a_t^s.

3.2.5. The Algorithm

We suggest two different ways to implement the approach we proposed. One is using the class information of assigned alternatives by the model to classify alternatives. The class information of the assigned alternatives will be taken as given even though they may be incorrectly classified. If there are alternatives assigned by the model, then linear programs should be solved again and new probabilities should be calculated by considering the information coming from the recently assigned alternatives.

The second one is using only the class information given by the DM. Here, we will use the class constraints of alternatives whose classes are given to us by the DM. The class constraints of the assigned alternatives by the algorithm are not added to linear programs. If there are alternatives assigned by the model, then linear programs need not be solved again because we will not use the information coming from the assigned alternatives. In this way, after assignments are done with the available information, class of an alternative should be asked to the DM to get new information. Here, we use class information of alternatives given to us at the beginning or asked to the DM during the classification process.

Step 0: Separate the alternatives into two sets C_0 and C_r. Put the alternatives to be classified to C_0 and put the alternatives whose classes are known to C_r. Group the alternatives in C_r according to their classes and put them into sets $C_1, C_2, ..., C_q$. Let h be used to count the alternatives assigned in an iteration. Let k denote the class index.

Step 1: Set $k = 1$ and $h = 0$. Go to step 2.

Step 2:

For each $a_t \in C_0$;

Step 2.1:

Solve $LP_{3.1}^{t,l}$ for a_t and each $a_l \in C_k$.

If $obj_{3.1}^{*}(t,l) \geq 0$ for one of $a_l \in C_k$, go to Step 2.2.

If $k \neq q$ and $obj_{3.1}^{*}(t,l) < 0$ for all $a_l \in C_k$, set $k = k+1$ and go to Step 2.1.

If $k = q$ and $obj_{3.1}^{*}(t,l) < 0$ for $a_l \in C_q$, go to Step 2.2.

Step 2.2:

Solve $LP_{3.2}{}^{t,l}$ for a_t and each $a_l \in C_k$.

If $obj_{3.2}^{*}(t,l) < 0$ for one $a_l \in C_k$, go to Step 2.3.

If $k \neq 1$ and $obj_{3.2}^{*}(t,l) \geq 0$ for all $a_l \in C_k$, set $k = k-1$ and go to Step 2.2.

If $k = 1$ and $obj_{3.2}^{*}(t,l) \geq 0$ for all $a_l \in C_1$, go to Step 2.3.

Step 2.3:

Find $p(t,l)$ for each a_l for which both $obj_{3.1}^{*}(t,l)$ and $obj_{3.2}^{*}(t,l)$ is found.

Find $\max_k(t)$ and $\min_k(t)$. Go to Step 2.4.

Step 2.4 :

If $\max_1(t) \geq (1-th)$, take a_t from C_0 and assign to C_1. Set $h = h+1$.

If $\max_k(t) \geq (1-th)$ and $\min_k(t) \leq th$, take a_t from C_0 and assign to C_k. Set $h = h+1$.

If $\min_q(t) \leq th$, take a_t from C_0 and assign to C_q. Set $h = h+1$.

Step 3:

If $|C_0| = 0$, go to Step 4.

If $|C_0| > 0$ and $h > 0$, go to step 1.

If $|C_0| > 0$ and $h = 0$, find $a_t{}^{s} = \arg\min_{a_t \in C_0} d_1(t)$ and present the DM the possible classes of a_t. Ask the DM to place a_t to one of the presented classes. Take this alternative from C_0 and assign it to the class, C_{k^*}, chosen by the DM. If $|C_0| > 0$, go to step 1. If $|C_0| = 0$, go to Step 4.

Step 4: Present alternatives in C_k as in class k to the DM for $k = 1,2,...,q$.

This algorithm uses both correct and potentially incorrect information. If we want to work with only guaranteed correct information, the algorithm will be modified as follows:

Let C_k' denote the set of alternatives assigned by the model to class k.

Step 2.4:

If $\max_1(t) \geq (1-th)$, take a_t from C_0 and put a_t to C_1' and $h = h+1$.

If $\max_k(t) \geq (1-th)$ and $\min_k(t) \leq th$, take a_t from C_0 and put a_t to C_k' and $h = h+1$.

If $\min_q(t) \leq th$, take a_t from C_0 and put a_t to C_q' and $h = h+1$.

Step 3:

If $|C_0| = 0$, go to Step 4.

If $|C_0| > 0$, find $a_t^s = \arg\min_{a_t \in C_0} d_1(t)$ and present the DM the possible classes of a_t. Ask the DM to place a_t to one of the presented classes. Take this alternative from C_0 and assign it to the class, C_{k^*}, chosen by the DM. If $|C_0| > 0$, go to step 1. If $|C_0| = 0$, go to Step 4.

Step 4: Present alternatives in C_k and C_k' as in class k to the DM for $k = 1,2,...,q$.

This algorithm uses only guaranteed correct information which constitute of the classes of alternatives given to us by the DM at the beginning or during the solution process.

3.3. THE ALGORITHM COMPARING ALTERNATIVES WITH UTILITY THRESHOLDS

We suggest a probabilistic classification by calculating the probability of each alternative being in each class. We calculate probabilities in a different way. We find the probability of alternatives to be classified being better than utility thresholds that separate the classes. We look at the minimum and maximum values of the difference of utility of each unassigned alternative and each utility threshold. We find ranges for the difference and calculate probabilities under assumptions on the distribution of the differences. After calculating probabilities, alternatives are classified by comparing probabilities with a given threshold. If none of the alternatives are classified with the available information, the class of an alternative is asked to the DM. Same procedure will be applied until all the alternatives are classified.

3.3.1. Linear Programs for Utility Ranges

In this case, we work with the same constraint set but different objective functions compared to the previous case. We compare unassigned alternatives with the utility thresholds. We find the minimum and maximum values of the difference of utility of each alternative from each utility threshold. To find the ranges, two linear programs should be solved for each $a_t \in C_0$ and each u_k. The objective of the first model is to minimize the difference of utility of a_t from u_k and the objective of the second model is to maximize the difference of utility of a_t from u_k. The linear programs used in the model are as follows:

$LP_{3.5}^{t,k}$

$Min \quad U[g(a_t)] - u_k$

s.to

$(3.1) - (3.6)$

$LP_{3.6}^{\,t,k}$

$Max \quad U[g(a_t)]-u_k$

s.to

$(3.1)-(3.6)$

These models should be solved for each $a_t \in C_0$ and for each u_k for $k=1,2,...,q-1$. Let the optimal objective function value of $LP_{3.5}^{\,t,k}$ for a_t and u_k be $obj_{3.5}^{*}(t,k)$ and the optimal objective function value of $LP_{3.6}^{\,t,k}$ for a_t and u_k be $obj_{3.6}^{*}(t,k)$.

If $obj_{3.5}^{*}(t,k')\geq 0$ for $u_{k'}$, then $obj_{3.5}^{*}(t,k)\geq 0$ for u_k for $k=k'+1,k'+2,...,q-1$.

If $obj_{3.5}^{*}(t,k)\geq 0$, then $obj_{3.6}^{*}(t,k)\geq 0$.

Obtaining a positive value in the minimization problem, we know that a_t is better than u_k so the worst class of a_t is k. If we find $obj_{3.5}^{*}(t,k')\geq 0$, then there is no need to solve $LP_{3.5}^{\,t,k}$ for a_t and u_k for $k=k'+1,k'+2,...,q-1$. There is also no need to solve $LP_{3.6}^{\,t,k}$ for a_t and u_k for $k=k',k'+1,...,q-1$.

If $obj_{3.6}^{*}(t,k')<0$ for $u_{k'}$, then $obj_{3.6}^{*}(t,k)<0$ for u_k for $k=1,2,...,k'-1$.

If $obj_{3.6}^{*}(t,k)<0$, then $obj_{3.5}^{*}(t,k)<0$.

Obtaining a negative value in the minimization problem, we know that a_t is a worse alternative than u_k so the best class of a_t is $k+1$. If we find $obj_{3.6}^{*}(t,k')<0$, then there is no need to solve $LP_{3.6}^{\,t,k}$ for a_t and u_k for $k=1,2,...,k'-1$. There is also no need to solve $LP_{3.5}^{\,t,k}$ for a_t and u_k for $k=1,2,...,k'$.

Let $|C_0| = m_o$ and number of classes be q. Then the maximum number of linear programs to be solved is $2 \cdot m_o \cdot (q-1)$.

3.3.2. Probability Calculation

Uniform Distribution

We calculate the probabilities in a similar way as in the previous case. It is assumed that the difference of the utility of each alternative and each utility threshold has a uniform distribution between the calculated minimum and maximum values. Each utility difference follows $U\left[obj^*_{3.5}(t,k), obj^*_{3.6}(t,k)\right]$.

What we need in our model is $P[U[g(a_t)] - u_k \geq 0]$. Let $a = obj^*_{3.5}(t,k)$ and $b = obj^*_{3.6}(t,k)$.

Let $p_k(t) = P[U[g(a_t)] - u_k \geq 0]$. Then $p_k(t)$ is found as follows:

$$p_k(t) = \begin{cases} 1 \quad if \quad a > 0 \\ \dfrac{(b-0)}{(b-a)} \quad if \quad a \leq 0, \quad b > 0 \\ 0 \quad if \quad b \leq 0 \end{cases}$$

Symmetric Triangular Distribution

We work with the difference of utility of alternatives and utility thresholds. The values of utility thresholds depend on the utility values of alternatives. They constitute the borders of the classes and they can be minimized and maximized up to the worst and best alternatives of classes. Since they are not independent, assuming the difference will follow a symmetrical triangular distribution will only be an approximation.

Let $a = obj^{*}_{3.5}(t,k)$ and $b = obj^{*}_{3.6}(t,k)$. Let $p_k(t) = P[U[g(a_t)] - u_k \geq 0]$ and then $p_k(t)$ is found as follows:

$$p_k(t) = \begin{cases} 1 \quad if \quad a > 0 \\ 1 - 2\dfrac{a^2}{(b-a)^2} \quad if \quad a \leq 0, \quad \dfrac{(a+b)}{2} > 0 \\ 2\dfrac{b^2}{(b-a)^2} \quad if \quad \dfrac{(a+b)}{2} \leq 0, \quad b > 0 \\ 0 \quad if \quad b \leq 0 \end{cases}$$

Normal Distribution

We found that under the assumption that the utility values of subintervals follow uniform distribution within their calculated ranges; the difference of utility values of two alternatives will follow normal distribution if the sum of the utility values of subintervals is conditioned to be 1 in Section 3.2.2. We use the difference of utility of alternatives from utility thresholds. We need the conditional distribution of utility of alternatives and the distribution of utility thresholds. Then, we can find the distribution of the difference.

Distribution of Utility Values of Alternatives

The conditional distribution of the utility values of alternatives can be calculated in a similar way. . Let $m(a_j)$ be the expected value of $U[g(a_j)]$ and $s(a_j)$ be the standard deviation of $U[g(a_j)]$. $m(a_j)$ and $s^2(a_j)$ can be calculated as follows:

$$U[g(a_j)] = \sum_{i=1}^{n} \sum_{p=1}^{b_i} z_{ip}(a_j) \cdot w_{ip}$$

$$E\left[U[g(a_j)]\right] = m(a_j) = \sum_{i=1}^{n} \sum_{p=1}^{b_i} z_{ip}(a_j) \cdot m(w_{ip})$$

$$V\left[U\left[g\left(a_j\right)\right]\right] = s^{2}\left(a_j\right) = \sum_{i=1}^{n}\sum_{p=1}^{b_i}\left(z_{ip}\left(a_j\right)\right)^2 \cdot s^{2}\left(w_{ip}\right)$$

We need the distribution given that $Y = \sum_{i=1}^{n}\sum_{p=1}^{b_i} w_{ip} = 1$. We need to find the joint distribution of $U\left[g\left(a_j\right)\right]$ and Y. Let us denote $U\left[g\left(a_j\right)\right]$ as U, $m\left(a_j\right)$ as $m(U)$ and $s\left(a_j\right)$ as $s(U)$. The joint distribution of U and Y follows a bivariate normal distribution with parameters $m(U)$, $s^{2}(U)$, $m(Y)$, $s^{2}(Y)$, r. r can be found as follows:

$$Cov(U,Y) = \sum_{i=1}^{n}\sum_{p=1}^{b_i} z_{ip}\left(a_j\right)\cdot\left(s^{2}\left(w_{ip}\right) + m^{2}\left(w_{ip}\right)\right) + \sum_{i=1}^{n}\sum_{p=1}^{b_i}\sum_{i'=1}^{n}\sum_{p'=1}^{b_{i'}} z_{ip}\left(a_j\right)\cdot m\left(w_{ip}\right)\cdot m\left(w_{i'p'}\right)$$
$$- \sum_{i=1}^{n}\sum_{p=1}^{b_i} z_{ip}\left(a_j\right)\cdot m\left[w_{ip}\right]^2 - m(U)\cdot m(Y)$$

$$r = \frac{Cov(U,Y)}{s(U)s(Y)}$$

Conditional distribution of bivariate normally distributed random variables follows normal distribution. U will follow normal distribution with parameters $E[U|Y=1]$ and $V[U|Y=1]$.

$$E[U|Y=1] = m(U) - r\,\frac{s(U)}{s(Y)}(1 - m(Y))$$

$$V[U|Y=1] = \left(1 - r^{2}\right)s(Y)^2$$

Distribution of Utility Thresholds

We will treat each u_k as a random variable. We do not know the actual value and we try to find the probability of possible values being the actual value. We can find the minimum and maximum values each u_k by solving the constraint set S

with objectives the of minimizing and maximizing each u_k. To do this, we need to solve the following models for $k = 1,2,...,q-1$.

$LP_{3.7}{}^{k}$

$Min \quad u_k$

s.to

$(3.1)-(3.6)$

$LP_{3.8}{}^{k}$

$Min \quad u_k$

s.to

$(3.1)-(3.6)$

Let $obj_{3.7}^{*}(k)$ be the optimal objective value of $LP_{3.7}{}^{k}$ and $obj_{3.8}^{*}(k)$ be the optimal objective value of $LP_{3.8}{}^{k}$. We need to solve $2\cdot(q-1)$ LPs to find the ranges of utility thresholds. We fit a normal distribution to each u_k. We assume that u_k has a higher probability of being around the mean. We know the minimum and maximum values. Let $m(u_k)$ be the expected value of u_k and $s(u_k)$ be the standard deviation of u_k. We assume $m(u_k)$ is at the middle of the range.

$$E[u_k] = m(u_k) = \frac{\left(obj_{3.7}^{*}(k) + obj_{3.8}^{*}(k)\right)}{2}$$

When we assume a normal distribution, there is a positive probability that u_k is smaller than $obj_{3.7}^{*}(k)$ or greater than $obj_{3.8}^{*}(k)$. In order to have a small chance of being out of the range, we assume that standard deviation of u_k is $1/6$ times the length of the range. By assuming this, we make our fitted distribution cover the range with more than 99% probability. Let X be any normally distributed random variable with mean m and standard deviation s , then

$P(m-3s \le X \le m+3s) = f(3) - f(-3) = 0.9987 - 0.0013 = 0.9974$.

Therefore $V[u_k]$ is calculated as following.

$$V[u_k] = s^2(u_k) = \left[\frac{\left(obj^*_{3.8}(k) - obj^*_{3.7}(k)\right)}{6}\right]^2$$

Distribution of the Differences

We showed that each $U[g(a_t)] \mid Y = 1$ follows $N\left(m'(a_t), s'^2(a_t)\right)$. We assumed that each u_k follows $N\left(m(u_k), s^2(u_k)\right)$. We need the distribution of the difference of each $U[g(a_t)] \mid Y = 1$ from each u_k. Let $d_k(t)$ be the difference of $U[g(a_t)] \mid Y = 1$ from u_k and $d_k(t) = \left(U[g(a_t)] \mid Y = 1\right) - u_k$. Let the expected value of $d_k(t)$ be $m(d_k(t))$ and the standard deviation be $s(d_k(t))$.

$m(d_k(t))$ can be found by taking the difference of means.

$$E[d_k(t)] = m(d_k(t)) = E\left[\left(U[g(a_t)] \mid Y = 1\right)\right] - E[u_k] = m'(a_t) - m(u_k)$$

To find the variance of the difference, we treat $U[g(a_t)] \mid Y = 1$ and u_k as independent random variables. The alternatives that are assigned to the classes affect the values the utility thresholds can take. Assuming that they are independent is an approximation.

$$V[d_k(t)] = s^2(d_k(t)) = V\left[\left(U[g(a_t)] \mid Y = 1\right)\right] + V[u_k] = s'^2(a_t) + s^2(u_k)$$

We assume that the difference will follow a normal distribution with mean $m(d_k(t))$ and variance $s^2(d_k(t))$.

Probability Calculation

We need $P\left[\left(U[g(a_t)]\,|Y=1\right)-u_k \geq 0\right]$. We found the parameters of the distribution and we can calculate the probabilities. However, we have more information about the difference. We have found the minimum and the maximum values the difference can take. We work on the conditional distribution which is feasible subject to constraint set S.

We will calculate the conditional probability of $d_k(t)$ being greater than 0 given that $d_k(t)$ is between the minimum and the maximum value of difference.

Let $p_k(t)=P\left[d_k(t)\geq 0 \middle| obj^*_{3.5}(t,k)\leq d_k(t)\leq obj^*_{3.6}(t,k)\right]$ and then $p_k(t)$ is found as follows:

$$p_k(t)=\frac{P\left[(d_k(t)\geq 0)\cap\left(obj^*_{3.5}(t,k)\leq d_k(t)\leq obj^*_{3.6}(t,k)\right)\right]}{P\left[obj^*_{3.5}(t,k)\leq d_k(t)\leq obj^*_{3.6}(t,k)\right]}$$

If $obj^*_{3.5}(t,k)\geq 0$, the intersection is the universal set and all possible values of $d_k(t)$ is greater than 0 and $p_k(t)=1$.

If $obj^*_{3.5}(t,k)<0$ and $obj^*_{3.6}(t,k)\geq 0$,

$$p_k(t)=\frac{P\left[(d_k(t)\geq 0)\cap\left(obj^*_{3.5}(t,k)\leq d_k(t)\leq obj^*_{3.6}(t,k)\right)\right]}{P\left[obj^*_{3.5}(t,k)\leq d_k(t)\leq obj^*_{3.6}(t,k)\right]}$$

$$=\frac{P\left[0\leq d_k(t)\leq obj^*_{3.6}(t,k)\right]}{P\left[obj^*_{3.5}(t,k)\leq d_k(t)\leq obj^*_{3.6}(t,k)\right]}$$

$$p_k(t)=\frac{f\left(\frac{obj^*_{3.6}(t,k)-m(d_k(t))}{s(d_k(t))}\right)-f\left(\frac{0-m(d_k(t))}{s(d_k(t))}\right)}{f\left(\frac{obj^*_{3.6}(t,k)-m(d_k(t))}{s(d_k(t))}\right)-f\left(\frac{obj^*_{3.5}(t,k)-m(d_k(t))}{s(d_k(t))}\right)}$$

If $obj^{*}_{3.6}(t,k) < 0$, the intersection is an empty set and there does not exist any possible value of $d_k(t)$ that is greater than 0 and $p_k(t) = 0$. In summary,

$$p_k(t) = \begin{cases} 1 \quad if \quad obj^{*}_{3.5}(t,k) \geq 0 \\ \dfrac{f\left(\dfrac{obj^{*}_{3.6}(t,k) - m(d_k(t))}{s(d_k(t))}\right) - f\left(\dfrac{0 - m(d_k(t))}{s(d_k(t))}\right)}{f\left(\dfrac{obj^{*}_{3.6}(t,k) - m(d_k(t))}{s(d_k(t))}\right) - f\left(\dfrac{obj^{*}_{3.5}(t,k) - m(d_k(t))}{s(d_k(t))}\right)} \quad if \quad obj^{*}_{3.5}(t,k) < 0, \quad obj^{*}_{3.6}(t,k) \geq 0 \\ 0 \quad if \quad obj^{*}_{3.6}(t,k) < 0 \end{cases}$$

We can try to find $U[g(a_t)] - u_k \mid Y = 1$. It is difficult to find the covariance between $U[g(a_t)]$ and u_k since u_k cannot be expressed in terms of w_{ip} values. One thing we can try is to fit a normal distribution for $(U[g(a_t)] - u_k) \mid Y = 1$ on the range found for the difference. By this way, we do not need assume independence for $U[g(a_t)] \mid Y = 1$ and u_k. Normal distribution for $(U[g(a_t)] - u_k) \mid Y = 1$ will be found in the same way we find a normal distribution to u_k. Expected value will be the middle of the range and standard deviation will be $\frac{1}{6}$ times of the width of the range to have the probability of being in the range more than 99%.

3.3.3. The Classification of Alternatives

We calculate probabilities for each alternative in C_0 to be better than each u_k and denote $p_k(t)$ as the probability of a_t being better than u_k. The alternatives are classified according to the comparison of probabilities to a predefined threshold value. Let th denote the threshold value.

Exact Classification

If we set th as 0, then we will make an exact classification. If $p_k(t)=1$, then a_t is guaranteed to be better than u_k. This means that there does not exist any set of parameters making utility of a_t smaller than u_k. If $p_k(t)=0$, then a_t is guaranteed to be worse than u_k. This means that there does not exist any set of parameters making utility of a_t greater than u_k.

With these probabilities we can make an exact classification if the following conditions are satisfied:

If $p_1(t)=1$, then a_t is in C_1.

If $p_k(t)=1$ and $p_{k-1}(t)=0$, then a_t is in C_k.

If $p_{q-1}(t)=0$, then a_t is in C_q.

Probabilistic Classification

If $th>0$, we make a probabilistic classification. The assignment of alternatives to the classes is done in the following way:

If $p_1(t)\geq(1-th)$, then a_t is better than u_1 with at least $(1-th)$ probability.. Therefore a_t is assigned to C_1.

If $p_k(t)\geq(1-th)$ and $p_{k-1}(t)\leq th$, then a_t is better than u_k with at least $(1-th)$ probability and worse than u_{k-1} with at least $(1-th)$ probability.. Therefore a_t is assigned to C_k.

If $p_{q-1}(t)\leq th$, then a_t is worse than u_{q-1} with at least $(1-th)$ probability. Therefore a_t is assigned to C_q.

With the calculated probabilities, the alternatives in set C_0 which satisfy one of the above conditions are classified. If no assignments are done, then the class of one alternative is asked to the DM. With the class information taken from the DM, new probabilities are calculated. This procedure will repeat until all the alternatives are assigned to classes.

3.3.4. Selection of the Alternative to Ask the DM

To select the alternative to be asked to the DM, the following procedure is followed. For each $a_t \in C_0$;

$d_2(t) = \min_k \left(\left| p_k(t) - 0.5 \right| \right)$ for $k = 1,2,...,q-1$

$a_t^s = \arg\min_{a_t \in C_0} d_2(t)$ is selected to be asked to the DM.

We select the alternative that has the closest probability of being better than any of the utility thresholds to 0.5. Probability of 0.5 means the alternative can be better or worse than a utility threshold with equal probability. It is equally likely that the alternative can be better or worse than a utility threshold. This means we are not able to obtain any information about the position of the alternative according to that threshold. We aim to ask the alternatives that we do not obtain much information about. We think that knowing the classes of such alternatives will give more restrictive constraints and give us narrower utility ranges.

Let a_t^s denote the selected alternative to be asked to DM. We can narrow down the number of classes a_t^s can be placed as in the previous case. If $p_{k'}(t) = 1$ and $0 < p_{k'-1}(t) < 1$, then k' will be the worst class a_t^s can be placed. If $p_{k''}(t) = 0$ and $0 < p_{k''+1}(t) < 1$, then k'' will be the best class a_t^s can be placed. a_t^s can be placed to classes $k', k'+1, ..., k''$.

3.3.5. The Algorithm

As in the previous case, we suggest two different ways to implement the algorithm. One is taking the class information of assigned alternatives as given and the other one is using only the class information coming from the DM.

Step 0: Separate the alternatives into two sets C_0 and C_r. Put the alternatives to be classified to C_0 and put the alternatives whose classes are known to C_r. Let h be used to count the alternatives assigned in an iteration. Let k denote the index for utility thresholds.

Step 1: Set $k=1$ and $h=0$. Go to step 2.

Step 2:

For each alternative in C_0;

Step 2.1:

Solve $LP_{3.5}{}^{t,k}$ for a_t and u_k.

If $obj^*_{3.5}(t,k) \geq 0$, go to Step 2.2.

If $k \neq q-1$ and $obj^*_{3.5}(t,k) < 0$, set $k = k+1$ and go to Step 2.1

If $k = q-1$ and $obj^*_{3.5}(t,k) < 0$, go to Step 2.2.

Step 2.2:

Solve $LP_{3.6}{}^{t,k}$ for a_t and u_k.

If $obj^*_{3.6}(t,k) < 0$, go to Step 2.3.

If $k \neq 1$ and $obj^*_{3.6}(t,k) \geq 0$, set $k = k-1$ and go to Step 2.2.

If $k = 1$ and $obj^*_{3.6}(t,k) \geq 0$, go to Step 2.3.

Step 2.3:

Find $p_k(t)$ for each a_t and u_k for which both $obj^*_{3.5}(t,k)$ and $obj^*_{3.6}(t,k)$ is found. Go to Step 2.4.

Step 2.4 :

If $p_1(t) \geq (1-th)$, take a_t from C_0 and assign to C_1. Set $h = h+1$.

If $p_k(t) \geq (1-th)$ and $p_{k-1}(t) \leq th$, take a_t from C_0 and assign to C_k. Set $h = h+1$.

If $p_{q-1}(t) \leq th$, take a_t from C_0 and assign to C_q. Set $h = h+1$.

Step 3:

If $|C_0| = 0$, go to Step 4.

If $|C_0| > 0$ and $h > 0$, go to step 1.

If $|C_0| > 0$ and $h = 0$, find $a_t^s = \arg\min_{a_t \in C_0} d_2(t)$ and present the DM the possible classes of a_t^s. Ask the DM to place a_t^s to one of the presented classes. If the DM assigns this alternative to class $k*$, take it from C_0 and place in C_{k*}. If $|C_0| > 0$, go to step 1. If $|C_0| = 0$, go to Step 4.

Step 4: Present alternatives in C_k as in class k to the DM for $k = 1,2,...,q$.

If only the guaranteed correct information is to be considered, the following modifications will be done to the algorithm.

Let C_k' denote the set of alternatives assigned by the model to class k.

Step 2.4 :

If $p_1(t) \geq (1-th)$, take a_t from C_0 and put a_t to C_1' and $h = h+1$.

If $p_k(t) \geq (1-th)$ and $p_{k-1}(t) \leq th$, take a_t from C_0 and put a_t to C_k' and $h = h+1$.

If $p_{q-1}(t) \leq th$, take a_t from C_0 and put a_t to C_q' and $h = h+1$.

Step 3:

If $|C_0| = 0$, go to Step 4.

If $|C_0| > 0$, find $a_t^s = \arg\min_{a_t \in C_0} d_2(t)$ and present the DM the possible classes of a_t^s. Ask the DM to place a_t^s to one of the presented classes. If the DM assigns this alternative to class $k*$, take it from C_0 and place in C_{k*}. If $|C_0| > 0$, go to step 1. If $|C_0| = 0$, go to Step 4.

Step 4: Present alternatives in C_k and C_k' as in class k to the DM for $k = 1,2,...,q$.

3.4. DISCUSSIONS ON TWO DIFFERENT APPROACHES

We propose two approaches. One is to compare the unassigned alternatives with already assigned alternatives. The other is to compare unassigned alternatives with utility thresholds. In LPs we used, the utility of alternatives are restricted by utility thresholds. We observed that two approaches do not give the same results.

$z_1 = Min \quad u_k$
s.to
$(3.1) - (3.6)$

$z_2(l) = Min \quad U[g(a_l)]$
s.to
$(3.1) - (3.6)$

$z_3 = Max \quad u_k$
s.to
$(3.1) - (3.6)$

$z_4(l) = Max \quad U[g(a_l)]$
s.to
$(3.1) - (3.6)$

u_k can be decreased more than utility values of $a_l \in C_k$ up to the utility values of $a_l \in C_{k+1}$ and this results in $z_1 \le \underset{a_l \in C_k}{Min} \quad z_2(l)$.

u_k can be increased more than utility values of $a_l \in C_{k+1}$ up to the utility values of $a_l \in C_k$ and this results in $z_3 \ge \underset{a_l \in C_{k+1}}{Max} \quad z_4(l)$.

In the approach we compare alternatives with already assigned ones, we use the objectives of $Min \quad U[g(a_t)]-U[g(a_l)]$ and $Max \quad U[g(a_t)]-U[g(a_l)]$ for every $a_l \in C_k$ and for each k. In the approach we compare with utility thresholds, we use the objectives of $Min \quad U[g(a_t)]-u_k$ and $Max \quad U[g(a_t)]-u_k$. Since u_k can be increased more than the utility values of $a_l \in C_{k+1}$, we obtain smaller values for the minimum of difference in the pairwise comparison approach. Since u_k can be decreased more than the utility values of $a_l \in C_k$, we obtain greater values for the maximum of difference in the pairwise approach. Subject to same constraints, the range we obtained in the approach that compares alternatives with utility thresholds is a subset of the range obtained in the pairwise approach. This means we work with larger ranges for difference in the pairwise approach. Larger ranges means we have less information about the differences and it is harder to place alternatives than the approach we compare alternatives with utility thresholds.

In the pairwise approach, the maximum number of linear programs to be solved is $2 \cdot |C_0| \cdot |C_r|$ at one iteration. Assume that there are $|A| = m$ alternatives and we start with one alternative assigned. At the worst case, we assume that we are not able to assign any alternatives and each time we ask the DM to place one alternative. Then, the total number of LPs needed to be solved throughout the algorithm can be found as follows.

$$2 \cdot \sum_{i=1}^{m} (m-i) \cdot i = 2 \cdot \sum_{i=1}^{m} m \cdot i - 2 \cdot \sum_{i=1}^{m} i^2 = 2 \cdot m \cdot \frac{m^2+m}{2} - 2 \cdot \frac{m \cdot (m-1) \cdot (2m+1)}{6}$$
$$= \frac{m^3}{3} + \frac{4m^2}{3} + \frac{m}{3}$$

.

There are a polynomial number of LPs $(O(m^3))$. The algorithm will end in polynomial time since each LP is solvable in polynomial time. This will be a worst case behavior and when the algorithm assigns alternatives, we will solve fewer LPs. When we compare the utility of alternatives with utility thresholds, the maximum number of linear programs to be solved is $2 \cdot (q-1) \cdot |C_0|$ at one

iteration. At the worst case, the total number of LPs needed to be solved throughout the algorithm is $2\cdot(q-1)\cdot\sum_{i=1}^{m} i = 2\cdot(q-1)\cdot\frac{m^2+m}{2} = (q-1)\cdot(m^2+m).$

There are a polynomial number of LPs $(O(q\cdot m^2))$ and the second algorithm will also end in polynomial time. The numbers of LPs are calculated for worst case behaviors. When the algorithm assigns alternatives, we will solve fewer LPs. As m increases, number of LPs needed to solve for the pairwise algorithm increases more than the algorithm we compare alternatives with utility thresholds.

3.5. DISCUSSIONS ON TWO DIFFERENT IMPLEMENTATIONS

We suggest two different ways to implement the two different approaches we proposed. One way is using the class information of assigned alternatives by the algorithm to classify alternatives. When alternatives are assigned by the model, the class information of newly assigned alternatives is used even though they may be classified incorrectly. The probabilities calculated in this model are with respect to both correct and incorrect information. Therefore, the probabilities cannot be interpreted as the probability of being in class k. For instance, if an alternative has a 0 probability of being in class k that does not necessarily mean that it cannot be in class k, because the information used in the calculation of this probability may be incorrect.

The other way is using the class information given by the DM only. Here, we use the class constraints of alternatives whose classes are given to us by the DM. The class constraints of the assigned alternatives are not added to LPs. Probabilities are calculated using only correct information. If an alternative has 0 probability of being in class k, then this means that it cannot be in class k for sure.

Theorem: If alternative j is placed to class k by using only correct information at 0 probability threshold, then the class constraints of this alternative $U[g(a_j)] - u_{k-1} \leq -d$ and $U[g(a_j)] - u_k \geq 0$ are redundant.

Proof: In order to place alternative j, we solve LPs with objectives $Min \;\; U[g(a_j)] - u_k$ and $Max \;\; U[g(a_j)] - u_k$ for each class k. If we obtain $(Min \;\; U[g(a_j)] - u_k) \geq 0$ and $(Max \;\; U[g(a_j)] - u_{k-1}) < 0$, then alternative j is placed in class k. This implies that $u_k \leq U[g(a_j)] < u_{k-1}$. According to the results of the solved LPs, when all constraints are satisfied, $u_k \leq U[g(a_j)] < u_{k-1}$ will be automatically satisfied. This means $U[g(a_j)] - u_{k-1} \leq -d$ and $U[g(a_j)] - u_k \geq 0$ do not restrict the current feasible region. Therefore, class constraints of alternative j are redundant. □

When probability threshold we used in our model is 0, then that does not make any difference to add class constraints of the alternatives assigned by the model. Therefore, there is no need to solve LPs and calculate probabilities when alternatives are classified by the model. The classification of these alternatives does not give any new information to us. The LPs should be solved and probabilities should be calculated again when the class of an alternative is asked to the DM.

When we work with probability thresholds greater than 0, adding the class constraints of assigned alternatives makes a difference. If an alternative is classified probabilistically by using only the correct information, then the class constraints of this alternative are not redundant. If an alternative is classified probabilistically to class k, we write constraints $U[g(a_j)] - u_{k-1} \leq -d$ and $U[g(a_j)] - u_k \geq 0$ to the model; although we obtained $(Min \;\; U[g(a_j)] - u_k) < 0$

or $\left(Max \quad U\left[g\left(a_j\right)\right]-u_{k-1}\right)\geq 0$. This means there are points in the current feasible region that do not satisfy these constraints. By writing these class constraints, we eliminate some points in the feasible region. Therefore, LPs should be solved and probabilities should be calculated again because their results have been changed with the assignments of new alternatives.

3.6. INFEASIBLE CASES

In the algorithm using the class information of assigned alternatives, infeasibilities may occur. When we ask the DM the class of a selected alternative, we present the DM the possible classes it can be placed to. When we ask the DM among the possible classes the alternative can be placed according to the available information, this information includes the class information of the misclassified alternatives. So we can face a situation such that the DM places an alternative to a class which is not a member of the narrowed set. It becomes infeasible to place the alternative to its correct class because of using the class constraints of the misclassified alternatives. In such a case, we suggest the following procedure.

We can detect infeasibility after we ask the DM to place an alternative. If we detect infeasibility, the alternative is assigned to the class the DM placed but we let violation of the class constraints for that alternative. We solve an LP to minimize the violation of the class constraints. We use the results of this LP as given in the next iteration. The constraints for that alternative are added to the constraint set with the minimum violation value used as a parameter.

Let a_g denote the alternatives which we detected infeasibility while placing to the class given by the DM. Let $C_{k'}$ denote the set of alternatives which are originally in C_k but it is infeasible to place them to C_k after assignments. If the DM places a_g to class k, we put a_g to set $C_{k'}$ instead of C_k. For a particular alternative $a_{g'}$, let $e_{g'}$ be the minimum amount of violation of class constraints of

$a_{g'}$. If the DM places $a_{g'}$ to class $k*$, then the minimum amount of violation is found by solving the following LP.

$LP_9^{g'}$

$Min \quad e_{g'}$

$s.to$

$(3.1)-(3.6)$

$$U[g(a_g)]-u_k \geq -V_g, \quad \forall a_g \in C_{k'}, \quad k'=1,2,...,q-1 \qquad (3.7)$$

$$U[g(a_g)]-u_{k-1} \leq V_g - d, \quad \forall a_g \in C_{k'}, \quad k'=2,3,...,q \qquad (3.8)$$

$$U[g(a_{g'})]-u_{k*} + e_{g'} \geq 0 \qquad (3.9)$$

$$U[g(a_{g'})]-u_{k*-1} - e_{g'} \leq -d \qquad (3.10)$$

Constraints (3.7) and (3.8) are written for the alternatives which are previously placed to $C_{k'}$. V_g denotes the minimum amount of violation previously found for a_g and it is used as a parameter in $LP_9^{g'}$. After solving $LP_9^{g'}$, $a_{g'}$ is assigned to $C_{k'}$. The objective value of $LP_9^{g'}$ is taken and it is used as parameter $V_{g'}$ to modify the constraint set S we used to solve all the mentioned LPs. All the mentioned LPs will be solved subject to constraints $(3.1)-(3.8)$.

Another way to deal with the infeasibilities is not to use the class information of the alternatives that we detect infeasibility while placing them to their original classes. If it is infeasible to place a_g to the class the DM placed, we do not add the class constraints of a_g to constraint set S. We put a_g to set $C_{k'}$ and do not write any constraints on $a_g \in C_{k'}$. We select another alternative to ask to the DM and continue with information coming from that alternative. In the end, we present the DM the alternatives both in C_k and $C_{k'}$ as in class k.

We worked on ways to deal with infeasibilities. We looked for a way to find the misclassified alternatives causing infeasibility. The class constraints of some of the misclassified alternatives conflict with the class constraints of the alternatives

assigned by the DM. We tried to solve an LP to eliminate the misclassified alternatives causing infeasibility. We relaxed class constraints of each alternative except for the ones that are placed by the DM. We added decision variables corresponding to violation of each class constraint. We solved this model with objective of minimizing total violation. We discarded the class constraints of the alternatives that have a positive violation value. After discarding them, the LPs used become feasible again. We treated them as unassigned alternatives and put into classification procedure again. By this way, we aim to eliminate some of the misclassified alternatives. However, correctly classified alternatives can have positive violation values. It is not guaranteed that only misclassified alternatives will be discarded because LP solved to minimize the violation can find a feasible combination of decision variables by relaxing the correctly classified alternatives. When we use the objective of minimizing the number of constraints to be violated instead of minimizing total violation, we end up with similar results. With that approach, we handle infeasibility but we lose information of some of the correctly classified alternatives. Therefore, these two approaches are not guaranteed to improve the results of the model.

Infeasible cases can occur when we use the class information of the assigned alternatives. If we work with only correct information, we do not face with an infeasible case. While working with the correct information, it can happen that there does not exist a utility function that can generate the classification we suggest to the DM. However, this situation does not cause any difficulty to us and does not cause any problems while the algorithm is working.

CHAPTER 4

IMPLEMENTATION AND RESULTS

We proposed an interactive multi-criteria sorting algorithm. We proposed two different approaches and two different ways of implementation. We showed how probabilities can be calculated when the difference follows uniform distribution, symmetric triangular distribution and normal distribution.

We implement our algorithm on a data set used by Köksalan and Özpeynirci (2009). Financial times publishes its ranking of top 100 global MBA programs every year. The programs are ranked based on three main criteria which are alumni career progress, diversity and idea generation. Köksalan et. al. (2009b) study the MBA ranking problem for 81 MBA programs using 2005 data and generate numerical scores based on the ranking provided by Financial Times. Köksalan and Özpeynirci (2009) experiments with this data to sort 81 MBA programs using 2005 data into 3 groups. We implement our algorithm with the same data set.

The alternatives are grouped into 3 classes. Each criterion range is divided into 3 subintervals. The limits of the subintervals can be seen in Table A.1. Köksalan and Özpeynirci (2009) assume the DM has an underlying model with parameter values u_k^* and w_{ip}^*, $k = 1,2$, $i = 1,2,3$ and $p = 1,2,3$. They use $u_1^* = 0.65$, $u_2^* = 0.40$ and the w_{ip}^* values in Table A.2. The marginal utility function of the DM on each criterion can be seen in Figure 4.1.

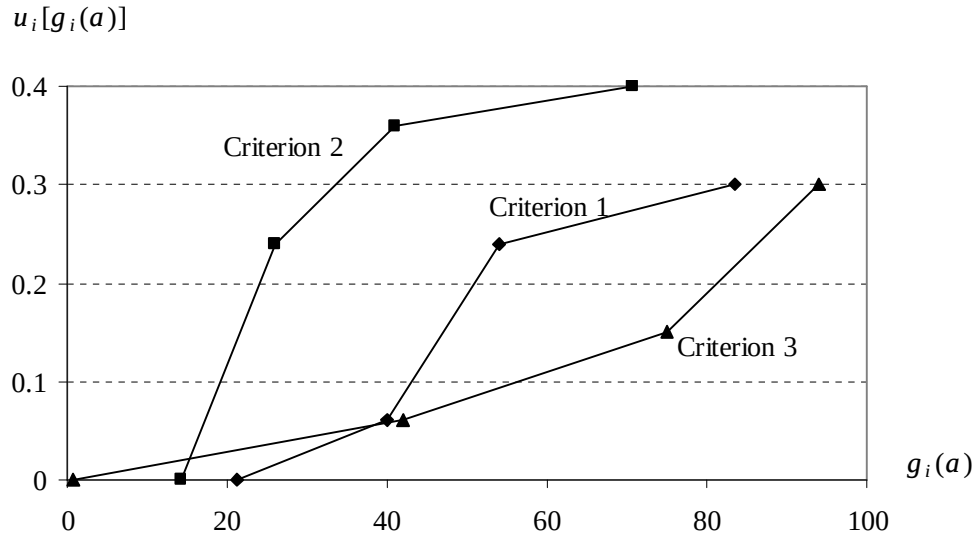

Figure 4.1: The Marginal Utility Functions on Each Criterion

The utility of the alternatives are calculated and the classes are assigned according to this underlying preference structure. 15 of the alternatives are in class 1, 47 of the alternatives are in class 2 and 19 of the alternatives are in class 3. We will use the same data and the same underlying model they assumed for the DM.

We evaluate our algorithm based on the number of misclassified alternatives and the number of alternatives whose classes are asked to the DM. We will try our algorithm for different probability threshold values. Probability threshold in the algorithm can be seen as a measure of the confidence level of the DM. When probability threshold is 0, we make an exact classification and we do not misclassify any of the alternatives. When probability threshold is small, the DM is expected to give more information but most of the alternatives will be correctly classified. When probability threshold is large, the DM does not give much information but accepts a relatively high number of misclassified alternatives. As probability threshold increases, we expect our algorithm to ask fewer questions and misclassify more alternatives. However, we can get unexpected results as well. Since there is randomness in the process, we can end up with fewer questions asked and fewer alternatives misclassified or with more questions asked

and more alternatives misclassified as probability threshold increases just by chance.

Sometimes, even if we cannot specify the class of an alternative, we may be able to reduce the possible number of classes it may belong to. To account for such situations, we also look at the number of classes presented to the DM to place the alternatives. We present the DM the possible classes the alternative can be placed and the DM can select the class from a narrowed set.

4.1. COMPARISON OF TWO DIFFERENT APPROACHES

We proposed two different approaches. One suggests the pairwise comparison of the utility of alternatives and the other suggests comparison of the utility of alternatives with utility thresholds. We see that it is harder to place the alternatives in the pairwise approach. We made experiments to compare two different approaches under the assumption that the differences will a follow uniform distribution. We used the mentioned data set and we used 4 more samples by changing the assumed underlying utility thresholds. Changing utility thresholds caused the change of the class of alternatives. We tested our data on these 5 samples. The utility thresholds and the number of alternatives in each class can be seen from Table 4.1.

Table 4.1: Utility Thresholds and the Number of Alternatives in Each Class

	u1	u2	Class 1	Class 2	Class 3
Sample 1	0.40	0.65	15	47	19
Sample 2	0.45	0.60	25	25	31
Sample 3	0.43	0.58	27	27	27
Sample 4	0.52	0.65	15	24	42
Sample 5	0.40	0.55	35	27	19

The detailed results of the two approaches on 5 different samples can be seen in Tables 4.2 and 4.3.

Table 4.2: Number of Misclassifications and Questions of Pairwise Model on Different Samples

		PAIRWISE COMPARISON										
	Threshold	**0.00**	**0.05**	**0.10**	**0.15**	**0.20**	**0.25**	**0.30**	**0.35**	**0.40**	**0.45**	**0.50**
Sample 1	**Misclassification**	0	0	0	0	0	0	3	3	3	22	14
	Question	47	43	39	36	31	30	27	26	18	10	0
Sample 2	**Misclassification**	0	0	0	0	0	2	4	4	4	6	16
	Question	57	56	53	48	43	42	27	35	22	11	0
Sample 3	**Misclassification**	0	0	0	0	0	0	0	1	6	8	24
	Question	58	57	54	48	38	49	35	24	14	4	0
Sample 4	**Misclassification**	0	0	0	0	0	0	0	0	1	3	10
	Question	64	60	54	48	37	34	31	26	17	8	0
Sample 5	**Misclassification**	0	0	0	0	0	1	0	4	4	8	27
	Question	57	54	50	45	39	44	31	35	15	7	0

Table 4.3: Number of Misclassifications and Questions of the Model Comparing with Utility Threshold

		COMPARING WITH UTILITY THRESHOLDS										
	Threshold	**0.00**	**0.05**	**0.10**	**0.15**	**0.20**	**0.25**	**0.30**	**0.35**	**0.40**	**0.45**	**0.50**
Sample 1	**Misclassification**	0	0	0	0	0	0	5	8	12	15	29
	Question	33	32	32	31	27	21	15	9	6	4	0
Sample 2	**Misclassification**	0	0	0	0	2	4	6	10	12	18	22
	Question	32	32	30	26	23	21	16	12	10	6	0
Sample 3	**Misclassification**	0	0	0	0	1	6	8	13	17	22	28
	Question	34	33	30	27	24	19	15	12	7	3	0
Sample 4	**Misclassification**	0	0	0	0	0	0	2	5	9	9	13
	Question	32	29	26	21	19	18	14	11	5	3	0
Sample 5	**Misclassification**	0	0	0	1	2	5	12	15	17	19	21
	Question	34	32	29	28	21	16	10	8	5	3	0

When we make an exact classification, comparing with utility thresholds performs better. It is able to classify asking fewer questions to the DM. This is an expected result because it is harder to place the alternatives in the pairwise approach. When probability threshold is small, both approaches make few misclassifications. Since comparing with utility thresholds asks fewer questions, it is preferred over the pairwise approach when the probability threshold is small. As probability threshold increases, the pairwise approach seems to make fewer misclassifications and ask more questions in our problems.

We compare CPU times for two approaches for different values of probability threshold. We give the results for 81 alternative MBA data for the original underlying model $u_1^* = 0.65$, $u_2^* = 0.40$ and the w_{ip}^* values in Table A.2. We implement our algorithm in C++ programming language. We use Microsoft Visual C++ 6.0 to implement our algorithm and use Cplex 10.1 solver to find the optimal solutions of the linear programs. Computational tests are made on a Intel Pentium D 3.00 GHz, 512 MB RAM computer running Microsoft Windows XP. The results can be seen in Table 4.4.

Table 4.4: CPU Times for Two Approaches

Threshold	Pairwise comparison	Comparing with utility thresholds
0.00	1702.66	105.44
0.05	1624.73	106.80
0.10	1482.05	85.98
0.15	1380.61	86.03
0.20	1177.30	62.72
0.25	1325.84	43.25
0.30	1185.28	23.67
0.35	537.67	13.97
0.40	525.14	9.02
0.45	343.14	5.63
0.50	198.78	3.58

We see that the implementations take much more CPU times at the pairwise approach compared to the one we compared the alternatives with utility thresholds. This is because the number of LPs solved at pairwise approach is much greater. As the probability threshold decreases, the CPU times decreases for both approaches. Since more alternatives are assigned in one iteration, fewer LPs are solved. We see that the pairwise approach is computationaly disadventageous compared to the other algorithm at all probability threshold values. We will compare the utility of the alternatives with utility thresholds in our next experiments.

4.2. COMPARISON OF DIFFERENT IMPLEMENTATIONS

We mentioned two different implementations. First one is using the information coming from the assigned alternatives. In this implementation, the information coming from misclassified alternatives are also used. The other one is using only the information coming from the DM. This implementation uses only correct information. We made an experiment on the data set mentioned for two different implementations.

When we ask the DM to place an alternative, we sometimes have information about the alternative and we are able to narrow down the number of classes the alternative can be placed in. For this three class problem, we present the number of alternatives that the classes they can be placed in are narrowed down to two and are not narrowed down. The number of questions asked to the DM will be sum of these numbers. We give detailed information for the alternatives that are placed by the DM. In general, the number of questions asked to the DM will be sum of the number of alternatives whose classes are narrowed down to $2,3,...,q-1$ classes and the number of alternatives not narrowed down. The detailed results can be seen in Tables 4.5 and 4.6.

Table 4.5: Number of Misclassifications and Questions of the Model Using Information of Assigned Alternatives

Threshold	Misclassification	Questions asked	Narrowed down to 1 class	Narrowed down to 2 classes	Not narrowed down
0.00	0	33	48	31	2
0.05	0	32	49	30	2
0.10	0	30	51	29	1
0.15	3	24	57	22	5
0.20	24	14	67	11	3
0.25	17	14	67	12	2
0.30	23	9	72	9	0
0.35	23	6	75	6	0
0.40	29	2	79	2	0
0.45	33	0	81	0	0
0.50	29	0	81	0	0

Table 4.6: Number of Misclassifications and Questions of the Model Using Correct Information

Threshold	Misclassification	Questions asked	Narrowed down to 1 class	Narrowed down to 2 classes	Not narrowed down
0.00	0	33	48	31	2
0.05	0	32	49	30	2
0.10	0	32	49	30	2
0.15	0	31	50	29	2
0.20	0	27	54	25	2
0.25	0	21	60	19	2
0.30	5	15	66	13	2
0.35	8	9	72	7	2
0.40	12	6	75	3	3
0.45	15	4	77	2	2
0.50	29	0	81	0	0

Both implementations have the same results when we make an exact classification. As the probability threshold increases, the first implementation asks fewer questions to the DM and results in more misclassified alternatives

since it can use incorrect information. The second implementation has better results in terms of number of misclassified alternatives and the number of questions asked to the DM is not considerably higher than the first implementation. We will make our experiments with the second implementation that uses only correct information.

4.3. COMPARISON WITH THE INTERACTIVE APPROACH

We obtain the same results with the interactive algorithm proposed by Köksalan and Özpeynirci (2009) when we make an exact classification. They check if it is feasible for an alternative to have a larger or smaller utility value than a utility threshold. In our algorithm when we make an exact classification, we look at the probability of an alternative being better than a utility threshold. If we get 0 as probability, it means that the utility of alternative is smaller than the utility threshold and there is no feasible solution that makes the utility of the alternative larger than that threshold. If we get 1 as probability, then the utility of alternative is larger than the utility threshold and there is no feasible solution to make it smaller. Therefore, two algorithms do the same job when our algorithm use 0 probability threshold.

Köksalan and Özpeynirci (2009) suggest to solve integer programs to check the feasibility of assigning an alternative to classes. Integer variables are needed to find a class to each unassigned alternative. Integer constraints are written to place unassigned alternatives. We observe that these constraints are redundant and there is no need to solve integer programs in this problem. If there is a feasible solution, utility of each alternative can be calculated with the estimated parameters. Then the alternatives can be assigned to any of the classes by comparing with the estimated utility thresholds. Trying to find a class to each of the unassigned alternatives does not restrict the feasible region and does not help to find a more restricted solution. For a particular case, solving LP always gives the same results

with solving IP. Since it is harder and more time consuming to solve IPs, LPs should be solved to get the results.

4.4. TAKING ORTHOGONAL ALTERNATIVES AS THE REFERENCE SET

Weber (1985) suggests creating hypothetical alternatives coming from an orthogonal design. He suggests to ask the utility ranges of the hypothetical alternatives to the DM. We worked on how orthogonal design concept can be applied to our case. We check if we can benefit from using orthogonal alternatives in our algorithm.

We observed that if we are given same length utility ranges for a set of alternatives, the constraints coming from orthogonal alternatives restrict the feasible region more than the constraints of other alternatives. We tried to show this on an example. Assume that we have 3 criteria and the marginal utility of each criterion is linear. This means we have 2 levels on each criterion where the first level, denoted by 0, corresponds to the minimum value of each criterion and the second, level denoted by 1, corresponds to the maximum value of each criterion. The orthogonal array with 3 factors and 2 levels in each factor can be seen in Table 4.7.

Table 4.7: Orthogonal Array with 3 Factors and 2 Levels

	Criterion 1	**Criterion 2**	**Criterion 3**
Alternative 1	0	0	0
Alternative 2	0	1	1
Alternative 3	1	0	1
Alternative 4	1	1	0

We ask the DM utility ranges for the last three orthogonal alternatives since alternative 1 has the lowest scores in all criteria and it is known that it has a 0 utility value. Let w_i denote the marginal utility of the maximum value of criterion i. In the example, assume that $w_1 = 0.4$, $w_2 = 0.3$ and $w_3 = 0.3$. Assume that the highest score in all criteria is 10 and the lowest score in all criteria is 0.

Assume that the DM has given us the utility ranges for each alternative as in Table 4.8.

Table 4.8: Criterion Values and Utility Ranges for Orthogonal Alternatives

	Criterion 1	Criterion 2	Criterion 3	Actual Utility Value	Utility Range
Alternative 1	0	10	10	0.6	(0.5 - 0.7)
Alternative 2	10	0	10	0.7	(0.6 - 0.8)
Alternative 3	10	10	0	0.7	(0.6 - 0.8)

The constraints we can write according to orthogonal alternatives are as follows.

$0.5 \le w_2 + w_3 \le 0.7$

$0.6 \le w_1 + w_3 \le 0.8$

$0.6 \le w_1 + w_2 \le 0.8$

Since it is known that $w_1 + w_2 + w_3 = 1$, we can write $1 - w_1 - w_2$ instead of w_3. The constraint set can be written as follows.

$w_1 \ge 0.3$ (4.1)

$w_1 \le 0.5$ (4.2)

$w_2 \ge 0.2$ (4.3)

$w_2 \le 0.4$ (4.4)

$w_1 + w_2 \ge 0.6$ (4.5)

$w_1 + w_2 \le 0.8$ (4.6)

The feasible weight space can be seen in Figure 4.2.

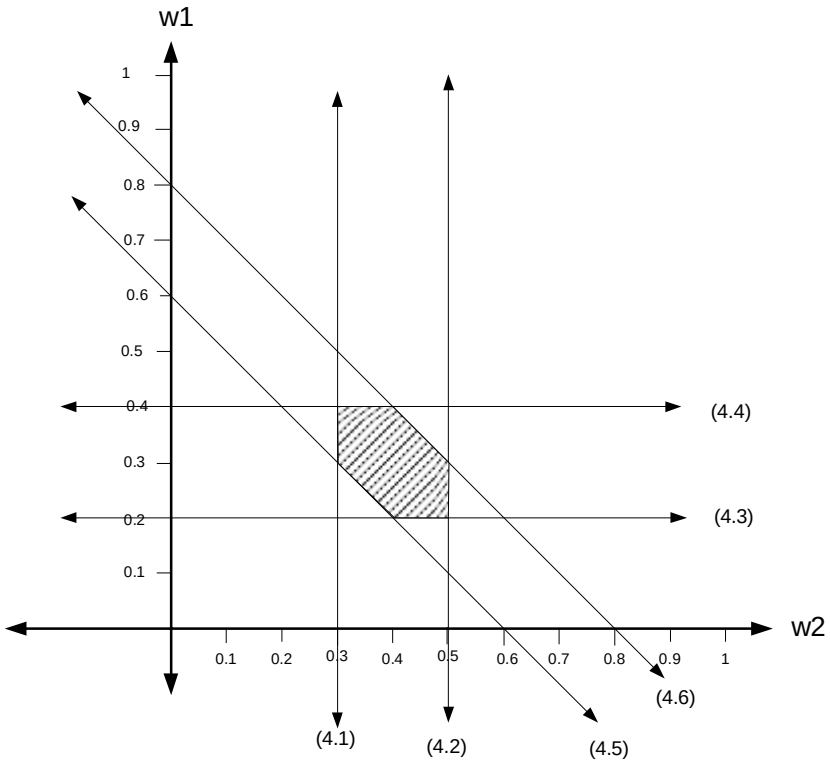

Figure 4.2: Feasible Weight Space of the Orthogonal Alternatives

To compare how weight space is restricted by constraints coming from alternatives, we created 2 different set of alternatives. In each case, we asked the DM the utility ranges of 3 alternatives that have the same utility value with orthogonal alternatives and assume that the DM has given us the same utility ranges with the range of orthogonal alternatives.

Criterion values of the first set of alternatives are given in Table 4.9.

Table 4.9: Criterion Values and Utility Ranges for the First Set

	Criterion 1	Criterion 2	Criterion 3	Actual Utility Value	Utility Range
Alternative 1	6	8	4	0.6	(0.5 - 0.7)
Alternative 2	5.5	8	8	0.7	(0.6 - 0.8)
Alternative 3	7	8	6	0.7	(0.6 - 0.8)

The constraint set defined by the alternatives in Table 4.8 is as follows.

$$0.5 \le 0.6w_1 + 0.8w_2 + 0.4w_3 \le 0.7$$
$$0.6 \le 0.55w_1 + 0.8w_2 + 0.8w_3 \le 0.8$$
$$0.6 \le 0.7w_1 + 0.8w_2 + 0.6w_3 \le 0.8$$

When we write $1 - w_1 - w_2$ instead of w_3, the constraint set becomes

$$0.2w_1 + 0.4w_2 \ge 0.1 \quad (4.7)$$
$$0.2w_1 + 0.4w_2 \le 0.3 \quad (4.8)$$
$$w_1 \ge 0 \quad (4.9)$$
$$w_1 \le 0.8 \quad (4.10)$$
$$0.1w_1 + 0.2w_2 \ge 0 \quad (4.11)$$
$$0.1w_1 + 0.2w_2 \le 0.2 \quad (4.12)$$

The feasible weight space can be seen in Figure 4.3.

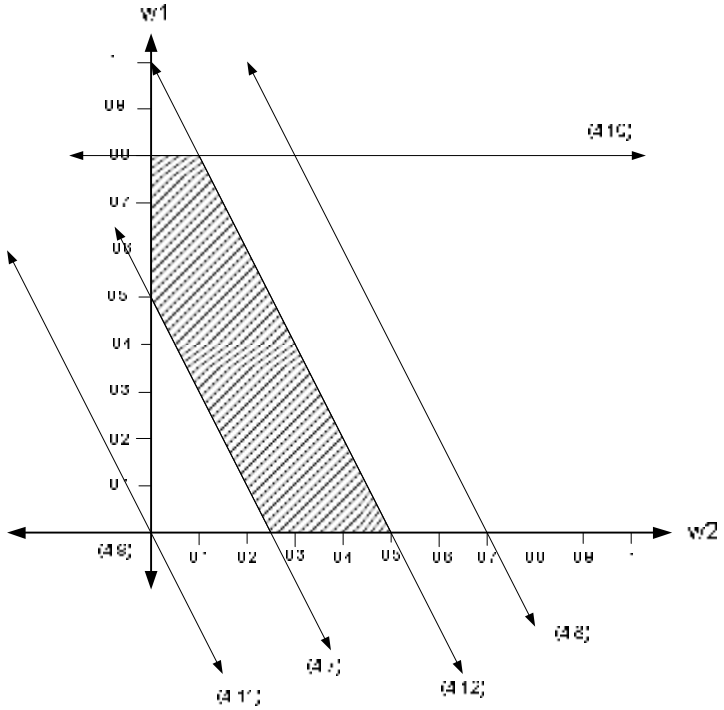

Figure 4.3: Feasible Weight Space of the First Set of Alternatives

Criterion values of the second set of alternatives are given in Table 4.10.

Table 4.10: Criterion Values and Utility Ranges for the Second Set

	Criterion 1	Criterion 2	Criterion 3	Actual Utility Value	Utility Range
Alternative 1	9	4	4	0.6	(0.5 - 0.7)
Alternative 2	7	4	10	0.7	(0.6 - 0.8)
Alternative 3	4	10	8	0.7	(0.6 - 0.8)

The constraint set defined by the alternatives in Table 4.9 is as follows.

$$0.5 \le 0.9w_1 + 0.4w_2 + 0.4w_3 \le 0.7$$
$$0.6 \le 0.7w_1 + 0.4w_2 + w_3 \le 0.8$$
$$0.6 \le 0.4w_1 + w_2 + 0.8w_3 \le 0.8$$

When we write $1-w_1-w_2$ instead of w_3, the constraint set becomes

$$w_1 \geq 0.2 \tag{4.13}$$
$$w_1 \leq 0.4 \tag{4.14}$$
$$0.3w_1 + 0.6w_2 \geq 0.2 \tag{4.15}$$
$$0.3w_1 + 0.6w_2 \leq 0.4 \tag{4.16}$$
$$0.4w_1 + 0.2w_2 \geq 0 \tag{4.17}$$
$$0.4w_1 + 0.2w_2 \geq 0 \tag{4.18}$$

The feasible weight space can be seen in Figure 4.4.

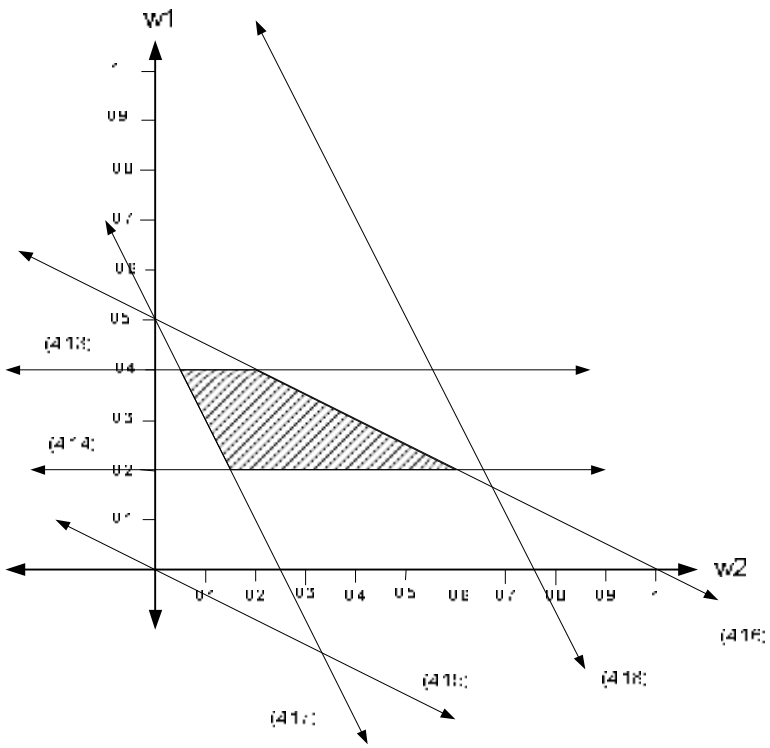

Figure 4.4: Feasible Weight Space of the Second Set of Alternatives

We obtained the most restricted region in the orthogonal design case. We observed that if we are given same length utility ranges, then the constraints coming from orthogonal alternatives are more restrictive compared to the ones

coming from other alternatives. We obtain the smallest area when the constraints are orthogonal to each other.

We tried to use the concept of orthogonality and tried to take hypothetical alternatives as reference set of our algorithm. We tried to see if starting with orthogonal alternatives will result asking fewer questions to the DM. In our data set, we have 3 criteria and 4 levels in each of the criterion. We will use 16 orthogonal alternatives. The created orthogonal alternatives are given in Appendix A.3. The first alternative is discarded because it has the minimum scores in all criteria and we know that it has a utility value of 0. We added the remaining 15 orthogonal alternatives to our data set.

We made an experiment to see the effect of orthogonality. In the experiment, we compared the utility of alternatives with utility thresholds and we assumed uniform distribution to calculate the probabilities. We tried our algorithm in two different ways on the data set including orthogonal alternatives. In the first one, the algorithm selects the alternative to ask the DM in the way we proposed at the previous chapter. We find the probability of each alternative being better than each utility threshold and the alternative that has the closest probability of being better than any of the utility thresholds to 0.5 is selected to ask the DM. In the second one, when we ask the class of an alternative to the DM, orthogonal alternatives have priority over the other alternatives. After all orthogonal alternatives are assigned, we select the alternatives that has closest probability to 0.5. The results of the experiment can be seen in Tables 4.11. and 4.12.

Table 4.11: Number of Misclassifications and Questions on the Data Set Orthogonal Alternatives Included

Threshold	Misclassification	Questions asked	Narrowed down to 1 class	Narrowed down to 2 classes	Not narrowed down
0.00	0	48	48	36	12
0.05	0	46	50	31	15
0.10	0	45	51	30	15
0.15	0	45	51	30	15
0.20	0	38	58	23	15
0.25	3	31	65	17	14
0.30	8	25	71	11	14
0.35	20	20	76	4	16
0.40	26	13	83	1	12
0.45	39	10	86	0	10
0.50	31	3	93	0	3

Table 4.12 Number of Misclassifications and Questions on the Data Set Orthogonal Alternatives Asked First

Threshold	Misclassification	Questions asked	Narrowed down to 1 class	Narrowed down to 2 classes	Not narrowed down
0.00	0	48	48	36	12
0.05	0	47	49	35	12
0.10	0	47	49	35	12
0.15	0	42	54	30	12
0.20	2	37	59	25	12
0.25	16	27	69	14	13
0.30	20	22	74	9	13
0.35	38	15	81	4	11
0.40	53	13	83	2	11
0.45	57	9	87	2	7
0.50	31	3	93	0	3

The two experiments result in the same number of questions asked to the DM when the probability threshold is 0. When probability threshold is equal to 0.05 and 0.10, the one started with orthogonal alternatives have asked more questions to the DM.

We see that under the constraints of our model, we do not obtain improved results when we first asked orthogonal alternatives to the DM. We have observed that using orthogonal alternatives restricts the weight space more than other alternatives if the same length utility ranges are given. In our algorithm, utility values of the alternatives are restricted by utility thresholds which are also our decision variables. The distribution of orthogonal alternatives to the classes is also important in the sorting problem. According to the utility function of the DM, if none of utility values of the orthogonal alternatives falls in a class, then we are not able to obtain information about this class. This can make classification harder. In our case, 7 orthogonal alternatives are in class 1, 4 are in class 2 and 4 are in class 3.

4.5. COMPARISON OF DIFFERENT DISTRIBUTIONS

We made experiments to observe how fitting different distributions to the difference of utility of alternatives from utility thresholds affects our results. We implemented our algorithm under the assumption that the difference follows uniform distribution, symmetric triangular distribution and normal distribution. We implemented our algorithm by using only the information coming from the DM.

We implemented our algorithm on the data set mentioned for different probability threshold values and we implemented on 2 randomly generated data sets used by Köksalan and Özpeynirci (2009). The first randomly generated data set have 100 alternatives evaluated in 3 criteria. The alternatives will be assigned to 5 classes. Each criterion range is divided into 3 subintervals. It is assumed that the DM has an underlying model with parameter values u_k^* and w_{ip}^*, $k = 1,2,3,4$, $i = 1,2,3$ and $p = 1,2,3$. The limits of subintervals can be seen in Table B.1. They use $u_1^* = 0.70$, $u_2^* = 0.60$, $u_3^* = 0.50$, $u_4^* = 0.35$ and the w_{ip}^* values in Table B.2. The marginal utility function of the DM on each criteria can be seen from Figure 4.5.

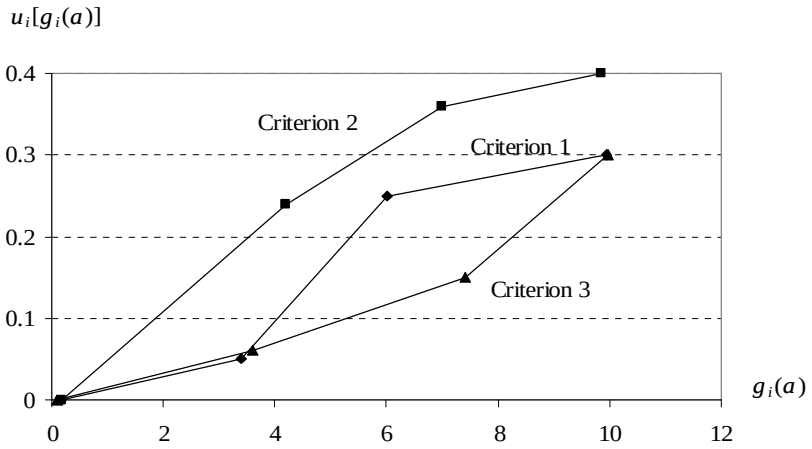

Figure 4.5: The Marginal Utility Functions on Each Criterion for the Data Set 1

The second randomly generated data set has 100 alternatives evaluated in 5 criteria. The alternatives will be assigned to 5 classes. Each criterion range is divided into 3 subintervals. The limits of subintervals can be seen in Table C.1. It is assumed that the DM has an underlying model with parameter values u_k^* and w_{ip}^*, $k = 1,2,3,4$, $i = 1,2,3,4,5$ and $p = 1,2,3$. They use $u_1^* = 0.70$, $u_2^* = 0.60$, $u_3^* = 0.50$, $u_4^* = 0.40$ and the w_{ip}^* values in Table C.2. The marginal utility function of the DM on each criteria can be seen from Figure 4.6.

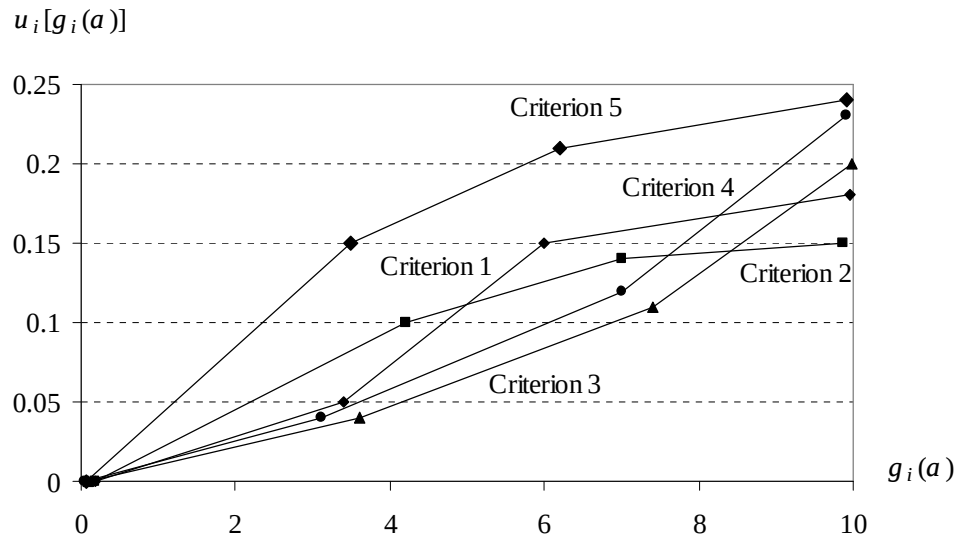

Figure 4.6: The Marginal Utility Functions on Each Criterion for the Data Set 2

We have created 15 orthogonal alternatives for both of the data sets. The orthogonal alternatives created for each of the randomly generated data set can be seen in Tables B.3 and C.3. We did the same experiments by assuming that the DM has given the class of orthogonal alternatives.

The results of the algorithm applied for the MBA data for uniform, symmetric triangular and normal distribution can be seen in Tables 4.13, 4.14 and 4.15, respectively.

Table 4.13: Number of Misclassifications and Questions for the MBA Data under Uniform Distribution Assumption

Threshold	Misclassification	Questions asked	Narrowed down to 1 class	Narrowed down to 2 classes	Not narrowed down
0.00	0	42	39	31	11
0.05	0	39	42	28	11
0.10	0	38	43	27	11
0.15	0	36	45	25	11
0.20	5	32	49	21	11
0.25	11	23	58	12	11
0.30	21	17	64	6	11
0.35	26	13	68	2	11
0.40	31	11	70	1	10
0.45	37	5	76	0	5
0.50	20	3	78	0	3

Table 4.14: Number of Misclassifications and Questions for the MBA Data under Triangular Distribution Assumption

Threshold	Misclassification	Questions asked	Narrowed down to 1 class	Narrowed down to 2 classes	Not narrowed down
0.00	0	42	39	31	11
0.05	1	35	46	24	11
0.10	8	28	53	16	12
0.15	12	21	60	10	11
0.20	22	14	67	4	10
0.25	27	13	68	2	11
0.30	30	11	70	1	10
0.35	34	8	73	0	8
0.40	37	6	75	0	6
0.45	38	4	77	0	4
0.50	20	3	78	0	3

Table 4.15: Number of Misclassifications and Questions for the MBA Data under Normal Distribution Assumption

Threshold	Misclassification	Questions asked	Narrowed down to 1 class	Narrowed down to 2 classes	Not narrowed down
0.00	0	37	44	29	8
0.05	0	36	45	28	8
0.10	0	33	48	25	8
0.15	0	32	49	24	8
0.20	0	31	50	23	8
0.25	1	26	55	18	8
0.30	4	20	61	12	8
0.35	11	15	66	8	7
0.40	16	11	70	2	9
0.45	22	8	73	0	8
0.50	39	3	78	0	3

The results for the data set with orthogonal alternatives included and randomly generated data sets can be seen in Appendix D.

When we make an exact classification, we obtain almost the same results in all 3 distributions. The differences in the results come from the difference in the selection of the alternatives placed by the DM. We find the probability of each alternative being better than each utility threshold. We ask the DM the alternative that has the closest probability to 0.5. Since different probabilities are calculated under different distributions, the alternatives asked the DM can be different. When probability threshold increases, the distribution used makes a difference. According to the results, none of the distributions perform significantly better than the other ones. In the above tables, in symmetric triangular distribution misclassification begins at smaller thresholds than the other ones. In uniform and normal distribution, there is no misclassification while threshold have increased up to 0.15. We see that most of the alternatives placed by the DM have been narrowed down to 2 classes. Even when the algorithm is not able to place the alternatives, it is able to narrow down the classes the alternative can be placed.

When we start with orthogonal alternatives to exact classification, we obtain improved results in terms of the number of questions asked to the DM. However, we ask 15 more questions to the DM at the beginning and this should be taken into consideration while evaluating the results. As probability threshold increases, orthogonal alternatives result in fewer misclassified alternatives and fewer questions to the DM. But again the extra number of questions asked to the DM should be taken into consideration. In normal distribution, the results have improved more by starting with orthogonal alternatives.

As the number of criteria increases, it becomes harder to place the alternatives and the number of alternatives placed by the DM increases. In the results of randomly generated data set 2, most of the alternatives are placed by the DM. We see that most of the alternatives placed by the DM have been narrowed down to two classes. The DM places most of the alternatives but we considerably reduce the number of classes the alternatives can be placed in.

CHAPTER 5

CONCLUSIONS

In this study, we proposed algorithms to multi-criteria sorting problem. We assumed an underlying additive utility function for the DM. Marginal utility function on each criterion is thought to be piecewise linear. Using the available information, we calculated probabilities for each unassigned alternative being in each class. We suggested a classification method by comparing the calculated probabilities with a given probability threshold value.

We implemented our algorithm on a data set used by Köksalan and Özpeynirci (2009) to assign 81 MBA programs to preference ordered groups. We also used two randomly generated data sets used by Köksalan and Özpeynirci (2009).

We evaluated our algorithm based on the number of alternatives placed by the DM and the number of misclassified alternatives. The algorithm can result in exact classification or probabilistic classification depending on the given probability threshold. When probability threshold is small, the DM is required to place many alternatives but most of the alternatives will be correctly classified. When probability threshold is large, we classify alternatives with less information but many alternatives can be misclassified. As probability threshold increases, the general trend is to ask fewer questions and misclassify more alternatives.

In our algorithm, we see that it is harder to place the alternatives in pairwise comparison of alternatives. When we make an exact classification, comparing with utility thresholds performs better and is able to classify with asking fewer

questions to the DM. When we make probabilistic classification, the pairwise approach asks more questions to the DM but misclassifies fewer alternatives.

We proposed two different implementations of the algorithm. We suggest to work with the one using the correct information given by the DM. It is expected that it will misclassify fewer alternatives compared to one using the class information of assigned alternatives by the algorithm.

In our study, we have thought of creating hypothetical orthogonal alternatives and asking the DM to place them at the beginning of the algorithm. We made experiments on our algorithm starting with the class information of orthogonal alternatives. This requires asking questions to the DM at the beginning. The number of questions asked depends on the number of criteria and the number of subintervals each criterion range has divided. We observed that constraints of orthogonal alternatives are more restrictive than the constraints of the other alternatives if the same length utility ranges are given for the alternatives. According to our results, although constraints written on them are more restrictive, they did not show a significant improvement in our results when we consider the additional questions asked to place these hypothetical alternatives.

We found the range for the difference of utility of alternatives from utility thresholds. We fit different distributions to the difference and tried to find the probability of difference being greater than 0. We tried uniform distribution, symmetric triangular distribution and normal distribution. In the experiments, we obtained nearly similar results for all distributions. We cannot say that fitting one distribution performs better than the other.

Different procedures can be used while selecting the alternatives to ask to the DM. The alternative placed by the DM affects the feasible region and we aim to ask the alternatives that will lead us to write more restrictive constraints. In the algorithm, we selected the alternative to ask the DM by looking at the

probabilities of being better than utility thresholds. We asked the alternative that has the closest probability of being better than any of the thresholds to 0.5. We aimed to ask the alternative that we do not have much information about. As a future research, a different procedure using orthogonality can be tried. Alternatives that can give the constraints closest to the orthogonal constraints of previously asked alternative can be selected to ask to the DM. If we search for the constraints orthogonal to the previously assigned one, we hope to have a reduced feasible region. However, the alternative set may not include an alternative that can give orthogonal constraints to the previously assigned alternative. Therefore, we suggest to select the alternative that can give closest constraints to the ones we search. By searching for orthogonal constraints, it is aimed to reduce feasible region more than the constraints derived from other alternatives.

REFERENCES

Addelman, S. and Kempthorne, O. (1962) Some main effect plans and orthogonal arrays of strength two. *The Annals of Mathematical Statistics,* Vol. 32, No. 4, 1167-1176.

Damart, S., Dias, L. C. and Mousseau, V. (2007) Supporting groups in sorting decisions: Methodology and use of a multi-criteria aggregation/disaggregation DSS. *Decision Support Systems*. Vol. 43, Issue 4, 1464-1175.

Dias, L. C. and Mousseau, V. (2003) IRIS: A DSS for multiple criteria sorting problems. *Journal of Multi-Criteria Decision Analysis*, Vol. 12, 285-298.

Diakoulaki, D., Zopounidis, C., Mavrotas, G. and Doumpos, M. (1999) The use of a preference disaggregation method in energy analysis and policy making. *Energy*, Vol. 24, No. 2, 157-166.

Doumpos, M. and Zopounidis, C. (2001) A preference disaggregation decision support system for financial classification problems. *European Journal of Operational Research*, Vol.130, Issue 2, 402-413.

Doumpos M. and Zopounidis C. (2004a) A multicriteria classification approach based on pairwise comparisons. *European Journal of Operational Research*, Vol. 158, Issue 2, 378-389.

Doumpos, M. and Zopounidis, C. (2004b) Developing sorting models using preference disaggregation analysis: An experimental investigation. *European Journal of Operational Research*, Vol. 154, Issue 3, 585-598.

Edwards, W. (1977) How to use multiattribute utility measurement for social decision making. *IEEE Transactions on Systems Man and Cybernetics,* SMC-7, 326-340.

Edwards, W. and Barron, F.H. (1994) Smarts and smarter: improved simple methods for multiattribute utility measurement. *Organizational Behavior and Human Decision Processes,* Vol. 60, Issue 3, 306–325.

Greco, S., Mousseau, V. and Slowinski R. (2008) Ordinal regression revisited: Multiple criteria ranking using a set of additive utility functions. *European Journal of Operational Research,* Vol. 191, Issue 2, 416-436.

Hedayat, A. S., Pu. K. and Stufken, J. (1992) On the construction of asymmetrical orthogonal arrays. *The Annals of Statistics,* Vol 20. No 4. 2142-2152.

Hines, W. W., Montgomery, D. C., Goldsman, D. M. and Borror, C. M. (2003) Probability and Statistics in Engineering, Chapter 7, The Normal Distribution, 160-164, *John Wiley & Sons, Inc., Fourth Edition*.

Jacquet-Lagzere, E. and Siskos, J. (1982) Assessing a set of additive utility functions for multicriteria decision making, the UTA method. *European Journal of Operational Research*, Vol. 10, Issue 2, 151-164.

Jacquet-Lagzere, E. and Siskos, J. (2001) Preference disaggregation: 20 years of MCDA experience. *European Journal of Operational Research*, Vol. 130, Issue 2, 233-245.

Köksalan, M., Mousseau, V., Özpeynirci, Ö. and Özpeynirci, S. (2009a) A new outranking-based approach for assigning alternatives to ordered classes. *Naval Research Logistics,* Vol.56, Issue 1, 74-85.

Köksalan, M. and Ulu, C. (2003) An interactive approach for placing alternatives in preference classes. *European Journal of Operational Research,* Vol. 144, No. 2, 429-439.

Köksalan, M. and Özpeynirci S. (2009) An Interactive Sorting Method for Additive Utility Functions. *Computers and Operations Research.* Vol.36, Issue 9, 2565-2572.

Köksalan, M., Büyükbaşaran, T., Özpeynirci, Ö. and Wallenius, J. (2009b) A flexible approach to ranking with an application to MBA programs. *European Journal of Operational Research,* doi: 10. 1016/j.ejor.2009.02.034

Larichev, O. I. and Moshkovich, H. M. (1994) An approach to ordinal classification problems. *International Transactions in Operational Research,* Vol. 1, Issue 3, 375-385.

Mousseau, V., Slowinski, R. and Zielniewicz, P. (2000) A user-oriented implementation of the ELECTRE-TRI method integrating preference elicitation support. *Computers and Operations Research,* Vol. 27, Issue 7-8, 757–777.

Mousseau, V., Figueira, J. and Naux, J.-Ph. (2001) Using assignment examples to infer weights for ELECTRE-TRI method: Some experimental results. *European Journal of Operational Research,* Vol. 130, Issue 2, 263–275.

Pasiouras, F., Gaganis, C. and Zopounidis, C. (2007) Multicriteria decision support methodologies for auditing decisions: The case of qualified audit reports in the UK. *European Journal of Operational Research,* Vol. 180, Issue 3, 1317-1330.

Pöyhönen, M. and Hamalainen R. P. (2001) On the convergence of multiattribute weighting methods. *European Journal of Operational Research*, Vol. 129, Issue 3, 569-585.

Ulu, C. and Köksalan, M. (2001) An interactive procedure for selecting acceptable alternatives in the presence of multiple criteria, *Naval Research Logistics*, Vol. 48, Issue 7, 592-606.

von Winterfeldt, D. and Edwards, W. (1986) Decision Analysis and Behavioral Research. *Cambridge University Press, Cambridge*.

Weber, M. (1985) A method of multiattribute decision making with incomplete information. *Management Science*, Vol, 31, No. 11, 1365-1371.

Zopounidis, C. (1999) Multicriteria decision aid in financial management. *European Journal of Operational Research*, Vol. 119, Issue 2, 404-415.

Zopounidis, C. and Doumpos, M. (2000) PREFDIS: A multicriteria decision support system for sorting decision problems. *Computers and Operations Research*, Vol. 27, No. 7, 779–797.

Zopounidis, C. and Doumpos, M. (2001) Assessing financial risks using a multicriteria sorting procedure: The case of country risk assessment. *Omega - The International Journal of Management Science*, Vol. 29, Issue 1, 97-109.

Zopounidis, C. and Doumpos, M. (2002) Multicriteria classification and sorting methods: A literature review. *European Journal of Operational Research*, Vol. 138, Issue 2, 229-246.

APPENDIX A

UNDERLYING MODEL OF THE MBA DATA

Table A.1: Limits of Subintervals for the MBA Data

	g_i^1	g_i^2	g_i^3	g_i^4
Criterion 1	21.18	40.00	54.00	83.44
Criterion 2	14.31	26.00	41.00	70.74
Criterion 3	0.71	42.00	75.00	93.91

Table A.2: w_{ip}^* of the Underlying Model for the MBA Data

i \ p	1	2	3
1	0.06	0.18	0.06
2	0.24	0.12	0.04
3	0.06	0.09	0.15

Table A.3: Orthogonal Alternatives Created for the MBA Data

Alumni Carrer Progress	Diversity	Idea Generation
21.18	14.31	0.71
21.18	26.00	42.00
21.18	41.00	75.00
21.18	70.74	93.91
40.00	14.31	42.00
40.00	26.00	0.71
40.00	41.00	93.91
40.00	70.74	75.00
54.00	14.31	75.00
54.00	26.00	93.91
54.00	41.00	0.71
54.00	70.74	42.00
83.44	14.31	93.91
83.44	26.00	75.00
83.44	41.00	42.00
83.44	70.74	0.71

APPENDIX B

UNDERLYING MODEL OF THE DATA SET 1

Table B.1: Limits of Subintervals for the Data Set 1

	g_i^1	g_i^2	g_i^3	g_i^4
Criterion 1	0.19	3.40	6.00	9.95
Criterion 2	0.19	4.20	7.00	9.87
Criterion 3	0.10	3.60	7.40	9.98

Table B.2. w_{ip}^* of the Underlying Model for the Data Set 1

i \ p	1	2	3
1	0.05	0.20	0.05
2	0.24	0.12	0.04
3	0.06	0.09	0.15

Table B.3 Orthogonal Alternatives for the Data Set 1

Criterion 1	Criterion 2	Criterion 3
0.19	4.20	3.60
0.19	7.00	7.40
0.19	9.87	9.98
3.40	0.19	3.60
3.40	4.20	0.10
3.40	7.00	9.98
3.40	9.87	7.40
6.00	0.19	7.40
6.00	4.20	9.98
6.00	7.00	0.10
6.00	9.87	3.60
9.95	0.19	9.98
9.95	4.20	7.40
9.95	7.00	3.60
9.95	9.87	0.10

APPENDIX C

UNDERLYING MODEL OF THE DATA SET 2

Table C.1: Limits of Subintervals for the Data Set 2

	g_i^1	g_i^2	g_i^3	g_i^4
Criterion 1	0.19	3.40	6.00	9.95
Criterion 2	0.19	4.20	7.00	9.87
Criterion 3	0.10	3.60	7.40	9.98
Criterion 4	0.05	3.10	7.00	9.91
Criterion 5	0.07	3.50	6.20	9.91

Table C.2: w_{ip}^* of the Underlying Model for the Data Set 2

i \ p	1	2	3
1	0.05	0.10	0.03
2	0.10	0.04	0.01
3	0.04	0.07	0.09
4	0.04	0.08	0.11
5	0.15	0.06	0.03

Table C.3: Orthogonal Alternatives for the Data Set 2

Criterion 1	Criterion 2	Criterion 3	Criterion 4	Criterion 5
0.19	4.20	3.60	3.10	3.50
0.19	7.00	7.40	7.00	6.20
0.19	9.87	9.98	9.91	9.91
3.40	0.19	3.60	7.00	9.91
3.40	4.20	0.10	9.91	6.20
3.40	7.00	9.98	0.05	3.50
3.40	9.87	7.40	3.10	0.07
6.00	0.19	7.40	9.91	3.50
6.00	4.20	9.98	7.00	0.07
6.00	7.00	0.10	3.10	9.91
6.00	9.87	3.60	0.05	6.20
9.95	0.19	9.98	3.10	6.20
9.95	4.20	7.40	0.05	9.91
9.95	7.00	3.60	9.91	0.07
9.95	9.87	0.10	7.00	3.50

APPENDIX D

RESULTS

Table D.1: Number of Misclassifications and Questions for the MBA Data under Uniform Distribution Assumption and Orthogonal Alternatives Included

Threshold	Misclassification	Questions asked	Narrowed down to 1 class	Narrowed down to 2 classes	Not narrowed down
0.00	0	33	48	31	2
0.05	0	32	49	30	2
0.10	0	32	49	30	2
0.15	0	31	50	29	2
0.20	0	27	54	25	2
0.25	0	21	60	19	2
0.30	5	15	66	13	2
0.35	8	9	72	7	2
0.40	12	6	75	3	3
0.45	15	4	77	2	2
0.50	29	0	81	0	0

Table D.2: Number of Misclassifications and Questions for the MBA Data under Triangular Distribution Assumption and Orthogonal Alternatives Included

Threshold	Misclassification	Questions asked	Narrowed down to 1 class	Narrowed down to 2 classes	Not narrowed down
0.00	0	33	48	31	2
0.05	0	31	50	29	2
0.10	0	23	58	21	2
0.15	3	20	61	18	2
0.20	6	14	67	12	2
0.25	9	9	72	7	2
0.30	12	6	75	3	3
0.35	12	5	76	2	3
0.40	13	4	77	2	2
0.45	23	4	77	2	2
0.50	29	0	81	0	0

Table D.3: Number of Misclassifications and Questions for the MBA Data under Normal Distribution Assumption and Orthogonal Alternatives Included

Threshold	Misclassification	Questions asked	Narrowed down to 1 class	Narrowed down to 2 classes	Not narrowed down
0.00	0	29	52	27	2
0.05	0	27	54	25	2
0.10	0	26	55	24	2
0.15	0	24	57	22	2
0.20	0	18	63	16	2
0.25	0	12	69	10	2
0.30	6	12	69	10	2
0.35	9	8	73	6	2
0.40	12	6	75	4	2
0.45	16	3	78	1	2
0.50	18	0	81	0	0

Table D.4: Number of Misclassifications and Questions for the Data Set 1 under Uniform Distribution Assumption

Threshold	Misclassification	Questions asked	Narrowed down to 1 class	Narrowed down to 2 classes	Narrowed down to 3 classes	Narrowed down to 4 classes	Not narrowed down
0.00	0	63	37	37	11	5	10
0.05	0	60	40	34	11	5	10
0.10	0	57	43	31	11	5	10
0.15	1	51	49	27	9	5	10
0.20	2	44	56	21	7	6	10
0.25	5	42	58	15	6	11	10
0.30	11	35	65	10	9	7	9
0.35	19	21	79	3	1	5	12
0.40	30	16	84	2	0	3	11
0.45	40	9	91	1	0	0	9
0.50	43	5	95	0	0	0	5

Table D.5: Number of Misclassifications and Questions for the Data Set 1 under Triangular Distribution Assumption

Threshold	Misclassification	Questions asked	Narrowed down to 1 class	Narrowed down to 2 classes	Narrowed down to 3 classes	Narrowed down to 4 classes	Not narrowed down
0.00	0	63	37	37	11	5	10
0.05	1	50	50	27	9	5	10
0.10	3	45	55	18	8	9	10
0.15	8	38	62	11	6	11	10
0.20	13	30	70	8	5	8	9
0.25	19	23	77	4	2	4	13
0.30	27	17	83	2	1	1	13
0.35	32	14	86	1	0	2	11
0.40	39	9	91	1	0	0	8
0.45	41	8	92	0	0	0	8
0.50	43	5	95	0	0	0	5

Table D.6: Number of Misclassifications and Questions for the Data Set 1 under Normal Distribution Assumption

Threshold	Misclassification	Questions asked	Narrowed down to 1 class	Narrowed down to 2 classes	Narrowed down to 3 classes	Narrowed down to 4 classes	Not narrowed down
0.00	0	63	37	38	7	7	11
0.05	0	60	40	35	7	7	11
0.10	2	53	47	28	7	7	11
0.15	4	51	49	24	8	9	10
0.20	9	52	48	18	10	7	7
0.25	13	38	62	13	12	6	7
0.30	20	32	68	8	8	9	7
0.35	32	23	77	4	7	3	9
0.40	39	16	84	2	3	3	8
0.45	44	11	89	2	0	1	8
0.50	50	5	95	0	0	0	5

Table D.7: Number of Misclassifications and Questions for the Data Set 1 under Uniform Distribution Assumption and Orthogonal Alternatives Included

Threshold	Misclassification	Questions asked	Narrowed down to 1 class	Narrowed down to 2 classes	Narrowed down to 3 classes	Narrowed down to 4 classes	Not narrowed down
0.00	0	46	54	35	10	1	0
0.05	0	43	57	32	10	1	0
0.10	0	40	60	29	10	1	0
0.15	1	37	63	26	10	1	0
0.20	1	34	66	23	10	1	0
0.25	2	26	74	15	10	1	0
0.30	10	18	82	9	6	3	0
0.35	18	12	88	3	6	3	0
0.40	21	10	90	1	8	1	0
0.45	26	5	95	0	3	1	1
0.50	26	0	100	0	0	0	0

Table D.8: Number of Misclassifications and Questions for the Data Set 1 under Triangular Distribution Assumption and Orthogonal Alternatives Included

Threshold	Misclassification	Questions asked	Narrowed down to 1 class	Narrowed down to 2 classes	Narrowed down to 3 classes	Narrowed down to 4 classes	Not narrowed down
0.00	0	46	54	35	10	1	0
0.05	1	36	64	25	10	1	0
0.10	1	32	68	21	10	1	0
0.15	6	24	76	13	10	1	0
0.20	14	14	86	6	5	3	0
0.25	19	11	89	2	6	3	0
0.30	18	10	90	2	5	3	0
0.35	25	7	93	0	4	3	0
0.40	26	5	95	0	3	1	1
0.45	33	3	97	0	2	1	0
0.50	26	0	100	0	0	0	0

Table D.9: Number of Misclassifications and Questions for the Data Set 1 under Normal Distribution Assumption and Orthogonal Alternatives Included

Threshold	Misclassification	Questions asked	Narrowed down to 1 class	Narrowed down to 2 classes	Narrowed down to 3 classes	Narrowed down to 4 classes	Not narrowed down
0.00	0	55	45	45	10	0	0
0.05	6	46	54	36	10	0	0
0.10	10	41	59	30	11	0	0
0.15	12	38	63	28	10	0	0
0.20	20	28	72	19	9	0	0
0.25	23	24	76	15	8	1	0
0.30	27	20	80	11	8	1	0
0.35	30	13	87	4	8	1	0
0.40	38	8	92	3	4	1	0
0.45	40	3	97	0	2	1	0
0.50	45	0	100	0	0	0	0

Table D.10: Number of Misclassifications and Questions for the Data Set 2 under Uniform Distribution Assumption

Threshold	Misclassification	Questions asked	Narrowed down to 1 class	Narrowed down to 2 classes	Narrowed down to 3 classes	Narrowed down to 4 classes	Not narrowed down
0.00	0	86	14	41	11	18	16
0.05	0	82	18	37	11	18	16
0.10	0	77	23	34	9	18	16
0.15	0	73	27	30	9	18	16
0.20	0	68	32	26	8	18	16
0.25	0	64	36	22	8	18	16
0.30	2	54	46	12	8	18	16
0.35	9	47	53	6	10	15	16
0.40	26	28	72	1	2	7	18
0.45	43	16	84	0	0	1	15
0.50	53	5	95	0	0	0	5

Table D.11: Number of Misclassifications and Questions for the Data Set 2 under Triangular Distribution Assumption

Threshold	Misclassification	Questions asked	Narrowed down to 1 class	Narrowed down to 2 classes	Narrowed down to 3 classes	Narrowed down to 4 classes	Not narrowed down
0.00	0	86	14	41	11	18	16
0.05	0	73	27	30	9	18	16
0.10	0	67	33	25	8	18	16
0.15	0	59	41	18	7	18	16
0.20	3	54	46	11	9	18	16
0.25	9	47	53	6	10	15	16
0.30	21	33	67	0	3	12	18
0.35	32	20	80	0	1	3	16
0.40	42	16	84	0	0	1	15
0.45	50	10	90	0	0	0	10
0.50	53	5	95	0	0	0	5

Table D.12: Number of Misclassifications and Questions for the Data Set 2 under Normal Distribution Assumption

Threshold	Misclassification	Questions asked	Narrowed down to 1 class	Narrowed down to 2 classes	Narrowed down to 3 classes	Narrowed down to 4 classes	Not narrowed down
0.00	0	86	14	38	12	15	21
0.05	0	83	17	35	12	15	21
0.10	0	77	22	31	10	15	21
0.15	1	72	28	22	13	16	21
0.20	5	66	34	18	10	17	21
0.25	12	60	40	9	20	13	18
0.30	23	45	55	1	13	13	19
0.35	30	35	65	0	7	9	19
0.40	35	27	73	0	3	4	20
0.45	50	12	88	0	0	0	12
0.50	56	5	95	0	0	0	5

Table D.13: Number of Misclassifications and Questions for the Data Set 2 under Uniform Distribution Assumption and Orthogonal Alternatives Included

Threshold	Misclassification	Questions asked	Narrowed down to 1 class	Narrowed down to 2 classes	Narrowed down to 3 classes	Narrowed down to 4 classes	Not narrowed down
0.00	0	80	20	51	18	8	3
0.05	0	75	25	46	18	8	3
0.10	0	69	31	40	18	8	3
0.15	0	60	40	31	18	8	3
0.20	0	54	46	26	17	8	3
0.25	1	46	54	20	15	8	3
0.30	5	38	62	12	15	8	3
0.35	16	29	71	2	15	9	3
0.40	21	17	83	1	5	7	4
0.45	30	8	92	1	2	4	1
0.50	44	0	100	0	0	0	0

Table D.14: Number of Misclassifications and Questions for the Data Set 2 under Triangular Distribution Assumption and Orthogonal Alternatives Included

Threshold	Misclassification	Questions asked	Narrowed down to 1 class	Narrowed down to 2 classes	Narrowed down to 3 classes	Narrowed down to 4 classes	Not narrowed down
0.00	0	80	20	51	18	8	3
0.05	0	58	42	29	18	8	3
0.10	1	51	49	25	15	8	3
0.15	3	46	54	19	16	8	3
0.20	6	34	66	9	13	9	3
0.25	16	28	72	2	14	9	3
0.30	19	19	81	1	6	9	3
0.35	22	15	85	1	6	7	1
0.40	29	9	91	1	2	3	3
0.45	36	4	96	0	1	2	1
0.50	44	0	100	0	0	0	0

Table D.15: Number of Misclassifications and Questions for the Data Set 2 under Normal Distribution Assumption and Orthogonal Alternatives Included

Threshold	Misclassification	Questions asked	Narrowed down to 1 class	Narrowed down to 2 classes	Narrowed down to 3 classes	Narrowed down to 4 classes	Not narrowed down
0.00	0	78	22	48	19	6	5
0.05	0	73	27	43	19	6	5
0.10	0	63	37	34	18	6	5
0.15	1	53	47	24	18	6	5
0.20	5	46	54	19	16	6	5
0.25	6	41	59	17	15	4	5
0.30	17	35	65	4	16	10	5
0.35	31	21	79	1	10	5	5
0.40	39	13	87	0	2	6	5
0.45	46	8	92	0	3	2	3
0.50	52	0	100	0	0	0	0