A PROBABILISTIC AND INTERACTIVE APPROACH TO MULTIPLE CRITERIA SORTING

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

BY

SİNEM MUTLU

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

JUNE 2015

ABSTRACT

A PROBABILISTIC AND INTERACTIVE APPROACH TO MULTIPLE CRITERIA SORTING

Mutlu, Sinem
M.S., Department of Industrial Engineering
Supervisor : Prof. Dr. Murat Köksalan
Co-Supervisor: Prof. Dr. Yasemin Serin

June 2015, 94 pages

In this thesis, we develop a method in order to assign alternatives that are evaluated by multiple criteria into preference ordered classes probabilistically. Our motivation is that; when there are large sets of alternatives, placement could be realized in a fast and effective way based on a reasonable misclassification ratio. We assume that the underlying utility function of Decision Maker (DM) is additive. We first ask DM to place reference alternatives into the classes. We develop an interactive probabilistic sorting approach that calculates the probability of each alternative being in each class. If all the alternatives can be placed into a class at the same time by evaluating probabilities, the procedure ends, if not DM is asked to place an unassigned alternative into a class and the procedure is repeated by utilizing this new information. The procedure ends when all the alternatives are placed.

We test the performance of the algorithms on four different problems. The performance of algorithm is evaluated by number of misclassified alternatives, expected number of misclassified alternatives, and number of alternatives that are asked to the DM for placement.

Keywords: Multiple Criteria Decision Making, Probabilistic Sorting, Interactive Methods.

ÖZ

ÇOK ÖLÇÜTLÜ SIRALAMA PROBLEMLERİNE OLASILIKSAL VE ETKİLEŞİMLİ YAKLAŞIMLAR

Mutlu, Sinem
Yüksek Lisans, Endüstri Mühendisliği Bölümü
Tez Yöneticisi : Prof. Dr. Murat Köksalan
Ortak Tez Yöneticisi: Prof. Dr. Yasemin Serin

Haziran 2015, 94 sayfa

Bu tezde, çok ölçütle değerlendirilen alternatifleri sıralı sınıflara yerleştirmek amacıyla bir yöntem geliştirmeye çalıştık. Motivasyonumuz, makul bir hata payıyla çok büyük setlerden oluşan alternatiflerin hızlı ve etkili bir şekilde yerleştirilebilmesiydi. Karar vericinin eklemeli bir fayda fonksiyonuna sahip olduğu varsayılmıştır. İlk adımda karar vericiden referans alternatifleri sınıflara yerleştirmesini istiyoruz. Her alternatifin her sınıfta bulunma olasılığını hesaplayan etkileşimli ve olasılıksal bir yaklaşım geliştirdik. Hesaplanan olasılıklarla tüm alternatifler sınıflara aynı anda yerleşebiliyorsa yöntem sona erer. Eğer tüm sınıflardaki alternatifler aynı anda yerleştirilemiyorsa karar vericiden bir alternatifi uygun sınıfa ataması istenir. Yeni elde edilen bilgi ile alternatifler tekrar sınıfa yerleştirilmeye çalışılır. Yöntem, tüm alternatifler yerleşince sona erer.

Algoritmanın performansı dört farklı problem üzerinde test edilmiştir. Algoritmanın performansı, yanlış sınıflandırılan alternatiflerin sayısı, beklenen yanlış sınıflandırma sayısı ve Karar verici tarafından yerleştirilen alternatiflerin sayısıdır.

Anahtar Kelimeler: Çok Amaçlı Karar Verme, Olasılıksal Sıralama, Etkileşimli Yöntemler

To My Parents

ACKNOWLEDGMENTS

I would like to thank my supervisors Prof. Dr. Murat Köksalan and Prof. Dr.Yasemin Serin for their guidance and comments throughout this study.

I am also grateful to Gökhan Ceyhan who is always with me when I need him and who always makes me happy.

I would like to thank my mother Türkan Mutlu, my father Niyazi Mutlu and my brother Sinan Mutlu for their faith in me throughout my life.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Multi-criteria problems that consider a set of discrete alternatives which are evaluated by many criteria can be examined under different categories. Choice problems try to find the best alternative or a set of good alternatives. Ranking problems on the other hand, try to order alternatives from best to the worst. Classification and sorting problems try to place alternatives into a set of classes. The difference between classification and sorting problems is in the way they define the classes. In sorting problems classes are preference ordered, whereas they are nominal in classification problems.

Multi criteria sorting problems aim to place a discrete set of alternatives into the preference ordered classes. Placing alternatives into preference ordered classes can be encountered in many different situations in practice such as placing students applying for graduate programs into accepted, rejected or wait-listed classes, classifying academic programs, categorizing countries into different risk groups, deciding on credit applications of bank customers according to their credit risk groups and selecting students for different types of scholarship programs.

Multi criteria sorting methods can be analyzed under three main categories: utility function-based methods, outranking-based methods and interactive methods that are based on either outranking or utility function-based methods and require the decision maker input throughout the solution process.

A well-known method for partitioning alternatives into the categories is the estimated utility function method. The estimated utility function represents the preferences of the decision maker (DM). Once the utility value of an alternative is specified, the alternatives can be sorted from best to the worst. Then, by using some reference alternatives whose preferences are previously provided by the DM and by considering the threshold values of different classes, alternatives can be assigned to the correct classes. Typically, in order to estimate the utility function many parameters are needed to be estimated. The reliability of these parameters is very important since they indicate the form of the utility function, in other words the preference structure of the DM.

Another well-known class for multi-criteria sorting problems is the outranking-based approaches. The outranking relation is a binary relation that provides the outranking degree of an alternative over another alternative. The outranking relation can result in the conclusion that an alternative outranks the other in a pairwise comparison. This means that there are enough arguments to conclude that the alternative is at least as good as the one it is compared with, while there is no strong argument to refuse this situation. Sometimes incomparability may also arise if an alternative has an outstanding value in some criteria but have a very low performance in another.

In both of these methods, in order to sort the alternatives lots of parameters are needed to be estimated. The reliabilities of both approaches depend on how well these parameters are estimated. Interactive methods on the other hand, obtain preference information from DM during the solution process. Throughout the process the DM is presented with improved solutions based on his/her responses. This facilitates a learning process for the DM both about the extent of available solutions' preferences and trade-offs.

We develop a probabilistic and interactive approach in order to sort alternatives. We develop two different solution approaches in order to place alternatives into preference ordered classes. We have tested the performance of our algorithm by utilizing different probability distributions and different data sets.

In Chapter 2, we review the literature developed for multi-criteria sorting problems. We consider the estimated utility function-based, outranking-based and interactive methods in the literature. In Chapter 3 we present the solution approach we developed which is an interactive and probabilistic approach. We present our experiments and their results in Chapter 4. We conclude in Chapter 5.

CHAPTER 2

LITERATURE REVIEW ON MULTI-CRITERIA SORTING PROBLEMS

In this chapter, we briefly review the literature on multi-criteria sorting problems. We review the literature on outranking-based sorting methods; estimated utility function-based sorting methods and interactive sorting methods.

The UTA (UTilitès Additives) method is developed by Jacquet-Lagrèze and Siskos [1]. The method aims to estimate additive utility functions in order to obtain as consistent as possible rankings based on a reference set of alternatives. It estimates one or more additive utility functions from a given reference set of alternatives which are already assigned to classes by the DM. Then based on this additive utility function, it tries to rank alternatives from best to the worst. Hence, the type of problem the UTA method deals with is a ranking problem.

The UTADIS method (UTilities Additives DIScriminates) is derived from the UTA method. The method is developed first by Devaud et al. (see Kosmido et al. [4]). Different variations are proposed by Jacquet Lagreze and Siskos, and Zopounidis and Doumpos. (see for instance Zopounidis and Doumpos [10]). It provides the sorting of alternatives through an additive utility function. It utilizes the structure of the UTA method for sorting the alternatives into the preference ordered classes rather than ranking them. The marginal utility functions are piecewise linear. While defining the piecewise linear marginal utility functions, range of each criterion is divided into subintervals. The utility values of each subinterval on each criterion are estimated based on a reference set in such a way that the possible misclassification errors for the reference set of the DM is minimized. Using the piecewise linear marginal utility functions on each criterion, the global utility for each alternative is tried to be estimated through a linear interpolation. Sorting is done by comparing the global utility of each alternative with utility thresholds corresponding to the lower bound of each class.

Köksalan and Ulu [5] develop an interactive approach for multi-criteria sorting problems assuming an underlying linear utility function of the DM. There are some alternatives which are already classified by the DM. In order to find the true class (or possible class ranges) of the remaining alternatives, dominance relations are checked first. Then to narrow down the possible classes an alternative can be placed, convex dominance relationships are utilized. After that, to compare the relative utilities of unplaced alternatives with the placed ones, a weight space reduction technique is utilized. If no feasible weights are found, then the class range obtained is tighter. Parameters of the linear utility function are based on past preferences of the DM. The algorithm is applied for the problem of placing students who applied to the master's degree program of the Industrial Engineering Department, Middle East Technical University.

Ulu and Köksalan [6] propose another interactive method that considers distributing alternatives into acceptable and unacceptable sets. They develop interactive procedures to deal with this problem when the DM's underlying utility function is linear, quasi-concave and general monotone. For the linear utility function case, in order to decide whether an alternative is acceptable or not they utilize the dominance relationships, convex domination and weight space reduction techniques as well as previous preferences of DM. They define five classes namely; "acceptable", "barely acceptable", "unacceptable", "not in the acceptable set" (either barely acceptable or unacceptable) and "not in the unacceptable set" (either barely acceptable or acceptable). Then, they ask the DM to place some alternatives into these sets initially. For quasi-concave function case they use dominance relationships in addition to properties of a quasi-concave utility function. They propose and use the idea that convex combinations of acceptable alternatives are also acceptable by using the property of the quasi-concave utility function. They use the idea that alternatives that are inefficient with respect to the convex cones (see Korhonen et al. [7]). For the general monotone case they propose using dominance relations.

Köksalan and Özpeynirci [8] demonstrate that the UTADIS method and its variations may misclassify many alternatives even if the DM places many of the alternatives into the classes directly. They argue that it is because of the fact that UTADIS estimates many parameters of an additive utility function and places alternatives based on these parameters which result in many alternative optimal solutions. Hence, classification of alternatives depends on which alternative optimal solution is selected. They propose an interactive method which guarantees to classify all alternatives correctly. In contrast to UTADIS, they do not estimate the values of the parameters. They try to partition alternatives into classes by utilizing underlying an additive utility function of the DM and by using past preferences. The algorithm ends when all alternatives are classified. In order to demonstrate that the procedure works well, they use the data set of ranking global MBA programs by "Financial Times." In addition, they utilize several randomly generated problems.

Soylu [11] develops a Tchebycheff utility function-based approach for multi criteria sorting problems. By using a Tchebycheff utility function, all the efficient alternatives could be found even if they are not on the convex part of the efficient frontier. Like in the UTADIS method some reference alternatives are assigned into the classes and the remaining alternatives are placed into the classes by comparing their utilities with these reference alternatives. These comparisons are made based on a Tchbeycheff utility function. In the algorithm, if the weights are not specified by the DM, then each alternative selects its own weight which makes that alternative closest to the ideal point. The aim of this procedure is to place each alternative to the best possible class.

Ishizaka et al. [12] proposes a method called AHPSort that is used for classifying alternatives into preference-ordered categories. They argue that AHPSort method requires much fewer comparisons than classical AHP, which facilitates decision making within large scale problems. In order to place each alternative into the correct class, the profiles of classes are defined. This is done either by defining a local limiting profile (minimum performance needed on each criterion to belong to a class) or with a local central profile which is defined by a representative alternative belonging to that class. Then by using eigenvalue method of AHP, importance of each criterion is found. By utilizing these values global priority of each alternative is calculated. In order to assign alternatives into the classes these global priorities are compared with limiting profiles or central profiles. In order to control the validity of the method, it is demonstrated for a real case of supplier selection problem.

Köksalan et al. [9] propose a model to "flexibly rank" alternatives defined by multiple-criteria into preference-ordered classes by using mixed integer programming. They suggest sorting (which they call as flexible ranking) is more suitable for many problem types compared to ranking. In order to rank the alternatives flexibly, they consider a linear aggregation method. They develop a procedure that places the alternatives into distinct classes that are "sufficiently" different. Although they consider a linear aggregation model, they permit changes in criterion weights in pre-defined ranges. The difference of this model is that, unlike the similar procedures that heavily involve the DM, it is based on "reasonable" weight ranges that can express preferences of many distinct parties. They develop a mathematical model in order to assign best possible weight to each alternative. Then, they modify and apply this model to the problem of ranking MBA programs based on FT Global MBA program rankings.

Zopounidis and Doumpos [2] make a literature review on multi-criteria classification and sorting methods. They examine multi criteria sorting problems under four categories: outranking-based approaches, estimated utility function-based approaches, rough set approaches and neural networks. They mention different methods from literature for each of the categories. They explain how these methods work as well as model development techniques for different kinds of models. Finally, they give information about developed decision support systems and real-world applications.

Köksalan et al [3] develop an outranking-based multi-criteria sorting method to assign alternatives into preference–ordered classes. Instead of using explicit reference profiles, they ask the DM to place reference alternatives to each category. In order to place unassigned alternatives into categories, they compare them with these reference alternatives. They assume that preference and indifference thresholds are known but criteria weights are unknown. The method tries to place the alternatives by eliminating circles. A circuit occurs when two alternatives in different categories outrank each other. The approach is demonstrated for the evaluation problem of MBA programs and a problem from the banking sector.

Mousseau et al. [15] explain ELECTRE TRI method based on the studies of Yu and Roy & Bouyssou. ELECTRE TRI uses outranking relations in order to assign alternatives into classes. It builds an outranking relation S in order to justify or reject the claim that aSb_h (a is at least as good as b_h) ELECTRE TRI places the alternatives into the predefined classes. An alternative is placed into a class based on the comparison of the alternative's value on each criterion with reference profiles defining the limits of a class. To compare an alternative a with a reference profile, b_h, partial concordance indices, are calculated first by using indifference and preference thresholds. Then, global concordance indices are computed. Global concordance reveals the extent to which a and b_h are concordant with the assertion that "a outranks b_h" ("b_h outranks a", respectively). After this step partial discordance index, which reveals the extent of opposition of criterion to the assertion "a outranks b_h" is calculated. ("b_h outranks a", respectively.) The criterion opposes a veto if the difference is greater than veto threshold.

Finally, the credibility index of the outranking relation is calculated. The degree of the credibility of the outranking relation reveals the extent to which "a outranks b_h" ("b_h outranks a," respectively) according to global concordance and discordance index.

The outranking relation S is made by means of a λ-cut which is called a cutting level. λ is considered as the smallest value of the credibility index compatible with the assertion that "a outranks b_h. Then there are two possible assignment procedures: optimistic procedure and pessimistic procedure. Alternatives are assigned to the classes based on these procedures.

Mousseau et al. [15] also develop software called ELECTRE TRI Assistant. They provide explanations about how to use this software. It estimates the parameters based on the past assignment preferences of the DM. It allows user to give his/her preferences about the weight and the cutting level by giving limits or providing comparisons. They state that in the next version they will include similar features for reference profiles and thresholds as well. For estimation, it applies a nonlinear programming based on pessimistic procedure without veto from assignment examples. The aim of the program is to find parameters that lead credibility indices that are substantially far from the cutting level. That is, the model tries to maximize the minimum distance.

Mousseau et al. [15] consider that the nonlinear program of ELECTRE TRI can be solved as a linear program if reference profiles and threshold values are provided. Hence, they only take into consideration inferring weights vector which makes the problem to be solved linear. The objective of the model is to find assignment which is as consistent as possible with the assignment examples given by the DM. Hence, the objective functions of NLP and LP are same. They argue that as the objective function value increases the model becomes more stable. In order to control whether the model can find inconsistencies, they assign some alternatives into wrong classes and they conclude that the model demonstrates a good ability. In addition they try some different objectives but they conclude that his does not bring significant improvement.

Zopounidis and Doumpos [13] develop a system called FINCLASS (FINancial CLASSification) which incorporates UTADIS method and its three variants in order to classify the alternatives (firms) into the classes. Furthermore, the financial/credit analyst can decide competitive level between the alternatives of the same class with regard to their global utility values via UTADIS method or any of its variants. The FINCLASS system integrates preference disaggregation system of MCDA approach with decision support systems in order to provide financial/credit analysts with a tool to study financial classification decision problems. Current form of FINCLASS is oriented towards the analysis and assessment of corporate performance and viability, in addition to credit risk evaluation. They use the data of previous years as a reference set in UTADIS and estimate the parameters of the model. They utilize these parameters in order to classify the alternatives of current year. With FINCLASS a wide range of financial classification problems can be studied, namely bankruptcy risk evaluation, credit granting, and assessment of corporate performance. Furthermore, the system could be adapted to study some other classification topics such as country risk assessment, and portfolio selection and management

Diakoulaki et al. [14] use the UTADIS method in order to sort 14 countries to 3 classes by means of their energy intensities which is defined as energy consumption per unit output. 13 factors are decided to be effective for reaching a desired level of energy efficiency. The objective in this study is to observe the relative importance of criteria and its change as time passes. They try to specify most important factors that affect energy efficiency.

Dehnokhlaji et al. [16] assume that the underlying utility function of the DM is quasi-concave. They extend Prasad et al.'s model [22] in order to obtain a strict partial order for a set of alternatives evaluated by multiple criteria. Based on underlying preference information, the model will classify the alternatives with respect to the cone and the polyhedron defined based on a preference subset. Alternatives are classified with respect to vertex of the cone as: surely better, surely worse or possibly better/worse than the vertex. The last situation provides a measure about the related alternative’s closeness to being dominated by the cone and the amount that dominates the part of the polyhedron. In order to classify the alternative heuristically two threshold values are defined that measures how much alternatives better or worse beyond the thresholds.

Hokkanen et al. [19] propose use of SMAA (Stochastic multi-objective acceptability analysis) as a decision support system for multi-criteria alternatives and multiple decision makers. They try to find the weight space based on the underlying utility function of the decision maker. They try to find favorable weight vectors that make the utility of an alternative better than the alternative it is compared with. In this method decision makers do not directly involved in the procedure. They calculate an acceptability index; in order find the probability of that alternative being the best based on its calculated favorable weight and central weight. The alternatives are chosen based on this acceptability index. They represent the use of SMAA technique for both deterministic and stochastic cases. In the stochastic case, they also calculate a confidence factor in order to measure the accuracy of the criteria data.

Dias et al. [20] propose a new method called SMAA-TRI for multi-criteria sorting problems. SMAA-TRI utilizes SMAA in order to evaluate the stability of some parameters used in ELECTRE-TRI. Hence, SMAA-TRI is a combination of ELECTRE-TRI and SMAA methods. SMAA-TRI makes use of ELLECTRE-TRI with arbitrarily distributed weights, cutting level and class profiles. The other variables are assumed to be deterministic and known. In order to calculate the share of each alternative, namely acceptability index from a finite set of arbitrarily distributed parameter values, Monte Carlo simulation is utilized.

Kadziński and Tervonen [21] consider the use of SMAA-TRI method for general monotone utility functions. SMAA-TRI method applies the SMAA procedure to the ELECTRE-TRI method. The method utilizes Monte Carlo Simulation to calculate acceptability indices by which alternatives are assigned to the classes. They examine both threshold-based and example-based sorting procedures. In the threshold based approach, preference model of the DM is represented by an additive utility function and a vector of thresholds which describes the lower and upper bounds of classes. In contrast, in the example based approach classes are defined with the example alternatives that are already placed into the classes.

Kadziński and Slowiński [23] develop a method called DIS-CARD for multi-criteria sorting problems. The method aims to assign a specified number of alternatives to each class or unions of some classes. The method uses ordinal regression procedure to represent preferences of DM based on some reference alternatives. They develop a mathematical model that takes into account both the preference information and the cardinality constraints of the classes. They implement the model on an additive value function and on an outranking based relation. They generate different constraints for both of the approaches. They illustrate the method in order to sort a company's international sales managers into four classes.

Rough set approach is claimed to be useful for problems with inconsistencies such as sorting problems. However the original rough set approach cannot be directly applied to the sorting problems. The first change proposed by Greco et al. [24] is change of the data table by a pairwise comparison table. In addition, discernibility relation was needed to be changed by dominance relation. At the end, the sorting problem will end with a set of decision rules that represent the preference model.

Dias and Mousseu [25] present software for multi-criteria sorting problems: IRIS (**I**nteractive **R**obustness analysis and parameters' **I**nference for multi-criteria **S**orting problems). The method is based on ELECTRE TRI method. They assume that reference profiles and class thresholds are known. It does not require the DM to specify any precise value for the parameters needed for the ELECTRE TRI method. DM interactively denotes some reference alternatives or constraints on the parameter values. DM may accept the sorting suggested by the method or continue by adding more constraints. If the constraints defined by the DM are inconsistent, IRIS presents ways in order to get rid of this inconsistency. They represent the user interface for all of the iterations on an example problem.

Larichev and Moshkovich [26] propose a sorting method based on dominance relationships. They check the dominance situation between assigned and unassigned alternatives. The scale of each criterion is verbal and can be updated according to the preferences of the DM.)

Zopounidis and Doumpos [27] present a system called PREFDIS (PREFerence DIScrimination) for multi-criteria sorting problems. The system is based on UTADIS, UTADIS I, UTADIS II and UTADIS III. DM can prefer the model that meets his/her preferences best. During the process DM can change marginal utilities. They illustrate their system on two examples.

Brans and Vincke [17] propose PROMETHEE methods for outranking based relations for ranking problems. PROMETHEE I, provides a partial ranking and PROMETHEE II provides a total ranking for a set of alternatives. The basic idea of the method is to extend the notion of criterion where the classical notion refers to preference and indifference relations. For some of the criteria, intransitivity of the difference relation is utilized, for others indifference relation can be turned into preference relation. In order to extend the criterion a preference function is introduced that provides the relation between each alternative in pairwise comparison. They propose use of six types of preference functions for six different criterion types.

Buğdacı et al. [18] propose an interactive probabilistic sorting method based on the probability of an alternative being in each class and restricting incorrect assignments below a previously determined threshold value. They assume that the DM has an additive utility function and marginal utility of each alternative on each criterion is piecewise linear. Initially, the DM is needed to place an alternative into a class. Then by using this information and the class information of already assigned alternatives, the probability of belonging to each class, for the unassigned alternatives is calculated. Then alternatives are either assigned probabilistically (by comparing their probability of being in each class with a threshold value) or exactly. If there are any remaining unassigned alternatives, they ask the DM to place another chosen alternative into a class and the probabilities are updated accordingly. Then the remaining alternatives are tried to be assigned. The algorithm terminates when all alternatives are assigned into classes. In order to reduce the number of classes an alternative may belong to, they solve two linear programs. One of them minimizes and the other maximizes the difference between the utility of an alternative and the threshold of a class. Hence, the optimal solutions of these models provide the worst and best possible classes respectively. The assignment to a class is realized if the probability of making an incorrect assignment is sufficiently small. This acceptable error level is a parameter and can be specified by the user. Therefore, the approach can also be used as an exact sorting algorithm if the acceptable error probability is zero.

We also develop an interactive and probabilistic sorting method. Like Buğdacı et al. we assume that the underlying utility function on DM is additive which is the only common point of two methods. However, the way we calculate the probability of belonging each class is different. In addition, we solve fewer linear models in order to calculate the probabilities which make the method more effective. The assignment procedure and the selection method for the alternative to ask the DM are different.

CHAPTER 3

BACKGROUND AND SOLUTION APPROACH

In this chapter, we first introduce the multi-criteria sorting problem. Later we explain three different multi-criteria sorting methods that we based our study on. The first is the UTADIS method. The second is an interactive sorting that is based on the UTADIS method. The third is developed by using the ideas in the second method and it employs a probabilistic approach in order to assign the alternatives into the classes. Lastly, we introduce the procedure proposed in the present study.

3.1. Multi-criteria Sorting Problem

We consider a set A of m alternatives $A = \{a_1, a_2, \dots, a_m\}$ evaluated by n criteria. The alternatives are to be placed into q preference classes $C_1, C_2, \dots, C_q$ where C_1 is the most and C_q is the least-preferred class.

Let $g_i(a_j)$ be the score of alternative a_j on criterion i, $u_i[g_i(a_j)]$ be the marginal utility of alternative a_j on criterion i and $U[g(a_j)]$ be the global utility of alternative a_j. We assume that the marginal utility $u_i[g_i(a_j)]$ is piecewise linear for $i = 1,2,\dots,n$. For criterion i, piecewise linear utility is defined as follows: Range of criterion i is divided into b_i subintervals $[g_i^p, g_i^{p+1}]$, $p \in \{1,2,\dots,b_i\}$. The contribution of subinterval $[g_i^p, g_i^{p+1}]$ of criterion i to the marginal utility $u_i[g_i(a_j)]$ is w_{ip}; that is, $w_{ip} = u_i[g_i^p] - u_i[g_i^{p+1}] \geq 0$, $p \in \{1,2,\dots,b_i\}$ Let r_{ji} satisfy $g_i^{r_{ji}} \leq g_i(a_j) \leq g_i^{r_{ji}+1}$ for $r_{ji} = 1, 2, \dots, b_i$, $i = 1,2,\dots,n$ and $j = 1,2,\dots,m$. Then, the marginal utility of alternative a_j on criterion i, $u_i[g_i(a_j)]$ can be written as:

$$u_i[g_i(a_j)] = \sum_{p=1}^{r_{ji}-1} w_{ip} + \frac{g_i(a_j) - g_i^{r_{ji}}}{g_i^{r_{ji}+1} - g_i^{r_{ji}}} w_{ir_{ji}}$$

We assume also that the global utility function is obtained by

$$U[g(a_j)] = \sum_{i=1}^{n} u_i[g_i(a_j)]$$

which can also be represented as

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

Since the utility function of the DM is unknown, w_{ip} values are unknown. We also assume that the DM has some unknown class thresholds that separate his/her preference classes. These thresholds are also unknown to the DM although he/she can classify the alternatives when asked. Let u_k be the class threshold between C_k and C_{k-1} for $j = 1,2,\ldots,q$. Then;

$$U[g(a_j)] \geq u_1 \Rightarrow a_j \in C_1$$

$u_k \leq U[g(a_j)] \leq u_{k-1} \Rightarrow a_j \in C_k$ for k = 2,…,q-1

$$U[g(a_j)] \leq u_{q-1} \Rightarrow a_j \in C_q$$

3.2. UTADIS Method

UTADIS method is a utility function-based multi-criteria sorting method developed first by Devaud et al. [4]. Different variations and examples are presented by Jacquet Lagreze and Siskos and Zopounidis and Doumpos [10]. The method assumes that the underlying utility function of the DM is additive and marginal utilities are piecewise linear. It tries to estimate marginal utilities of each criterion first. Then, the global utility function, which is a weighted sum of marginal utilities, and the class thresholds separating different classes are estimated. The estimation is based on some initial assignments made by the DM. These initial assignments are used as reference points for the method. While introducing the method the notation of Zopounidis and Doumpos [10] is utilized.

UTADIS estimates the additive utility function using the information from the alternatives that are already assigned to the classes by the DM initially. In order to estimate the utility function, a linear program is constructed that minimizes the maximum misclassification errors of alternatives that are placed by the DM. There are two kinds of possible misclassification errors: upper bound violation and lower bound violation. If an alternative that is placed into class C_k by the DM is placed into C_{k+1} by the estimated utility function (violates class threshold u_k), then this is a lower bound violation. On the other hand, if an alternative that is placed into class C_k by the DM is placed into C_{k-1} by the estimated additive utility function (violates class threshold u_{k-1}), then this is an upper bound violation.

The possible violations are measured by:

$$\sigma_j^+ = \max\,\{0, u_k - U[g(a_j)]\}, \qquad \forall a_j \in C_k$$

$$\sigma_j^- = \max\,\{0, U[g(a_j)] - u_{k-1}\}, \qquad \forall a_j \in C_k$$

where σ_j^+ is the amount of lower bound violation and σ_j^- is the amount of upper bound violation of the alternative $a_j \in C_k$.

In other words, $\sigma_j^+ > 0$ denotes that the actual class of misclassified alternative a_j is class C_k instead of class C_{k+1}. In order for alternative a_j to be placed in class C_k, its global utility should be increased by $u_k - U\left[g(a_j)\right]$. If $\sigma_j^- > 0$ then this denotes that the actual class of misclassified alternative a_j is class C_k instead of class C_{k-1}. In order for alternative a_j to be placed in class C_k, its global utility should be decreased by $U\left[g(a_j)\right] - u_{k-1}$.

The linear program P1 that estimates the coefficients, w_{ip}, of the utility function and the class thresholds, u_k, is given below:

$$\text{(P1)}: Min\ f = \sum_{k=1}^{q} \left[\frac{\sum_{\forall a_j \in C_k} \left(\sigma_j^+ + \sigma_j^-\right)}{|C_k|} \right]$$

s. t.

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

$$U\left[g(a_j)\right] - u_1 + \sigma_j^+ \ \geq\ \delta_1 \quad \forall a_j \in C_1 \tag{2}$$

$$\left.\begin{array}{l} U\left[g(a_j)\right] - u_k + \sigma_j^+ \ \geq\ \delta_1 \\ \\ U\left[g(a_j)\right] - u_{k-1} + \sigma_j^- \ \leq\ -\delta_2 \end{array}\right\} \ \forall a_j \in C_k\,,\ k = 2,3,\dots,q-1 \tag{3}$$

$$U\left[g(a_j)\right] - u_{q-1} + \sigma_j^- \ \leq -\delta_2 \qquad \forall a_j \in C_q \tag{4}$$

$$\sum_{i=1}^{n} \sum_{p=1}^{b_{i-1}} w_{ip} = 1 \qquad i = 1,2,\dots,n \qquad p = 1,2,\dots,b_{i-1} \tag{5}$$

$$u_k - u_{k+1} \geq s \qquad \forall k = 1,2,\dots,q-2 \tag{6}$$

$w_{ip} \geq 0 \qquad \forall i = 1, \ldots, n \quad \forall p = 1, \ldots, b_{i-1}$

$\sigma_j^+ \geq 0\,, \ \sigma_j^- \geq 0 \qquad \forall j = 1, \ldots, m$

$\delta_1 \ , \delta_2 \ , s \ are\ user\ defined\ small\ positive\ constants$

This LP minimizes a weighted sum of classification errors σ$^+$ and σ$^-$. The weights are determined according to the size of the class, $|C_k|$ where $k = 1,2, \ldots, q$. It is assumed that the number of reference alternatives for each class that are placed by the DM are close to each other.

Constraint (1) represents the DM's additive utility function $U[g(a_j)]$.

Constraints (2)-(4) describe classification errors, $\sigma_j^+$and σ_j^-, defined before.

Constraint (5) normalizes the additive utility function to [0, 1] interval.

Constraint (6) guarantees the ordering of classes by making sure that lower bound u_k of class C_k is higher than lower bound u_{k+1} of class C_{k+1}.

Zopounidis and Doumpos [10] proposed a post optimality analysis. The optimal solution gives a placement of the alternatives some of which are possibly misclassified. After the optimal objective function value is found a post optimality analysis is needed to be employed since there are alternative optimal solutions in most of the cases. During the post optimality step, criteria weights $\sum_{p=1}^{b_{i-1}} w_{ip}$, $i = 1,2, \ldots, n$ and the utility thresholds $u_k, k = 1,2, \ldots, q-1$, are tried to be maximized which may result in $n + q - 1$ alternative optimal or near optimal solutions. The first n solutions maximize the importance of each criterion $g_1(a_j)$, $g_2(a_j), \ldots$, $g_n(a_j)$ and the remaining $q - 1$ solutions maximize utility thresholds $u_1, u_2, \ldots, u_{q-1}$. The final additive utility function is formed from the average of all above.

3.3. Interactive Methods

3.3.1. An Interactive Sorting Method for Additive Utility Functions by Köksalan and Özpeynirci [8]

Köksalan and Özpeynirci [8] show that UTADIS method may misclassify many alternatives even if a significant amount of preference information is taken from the DM. They argue that the reason for high number of misclassifications in UTADIS method is the high number of parameters needed to be estimated in order to form the additive utility function. Classification of alternatives depends on which alternative optimal solution is selected. The method of Köksalan and Özpeynirci [8] requires from DM to place some alternatives into the classes iteratively. It guarantees to place all the alternatives into the classes correctly for a DM whose preferences are consistent with an additive utility function.

In order to show that a lot of alternative solutions are obtained with the UTADIS method they have made experiments with the original method and with different secondary objectives while preserving the main objective of minimizing misclassification of alternatives. Furthermore, they work on variations of the original UTADIS by adding some constraints on the possible values of parameters. They make the experiments on a three-class example with 30 reference alternatives out of 81 alternatives.

Köksalan and Özpeynirci [8] adopt the linear approach developed by Köksalan and Ulu [5] to a more general case of additive utility function. The aim of the method is to ask only small number of alternatives to the DM while warranting sorting all the alternatives correctly. Instead of estimating the parameters, the method tries to place alternatives into the classes by introducing the preferences of the DM to the model. They solve an integer programming model for each alternative in order to find the possible classes for that alternative. If the worst and best possible classes of an alternative are the same, then the alternative is placed to that class. If there is more than one possible class for an alternative, then they show the possible classes to the DM, and ask DM to place the alternative. After an alternative is placed by the DM, the procedure repeats until all the alternatives are placed.

Let A be the set of alternatives that will be classified and let C_0 be the set of alternatives that are not assigned to any classes yet. Let C_r represent the set of alternatives that are known to be in class C_k for $k = 1,2,\dots,q$. Suppose $a_j \in C_0$. Let C_j^W be the worst and C_j^B be the best category a_j can be placed in.

They use the notation presented in UTADIS. They define the binary variable x_{jk} to represent the assigned class of an alternative;

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

The model $IP1_{a,t}$ represented below controls whether the worst class that the alternative a_j can be placed in is class t.

$(IP1_{a,t})$: Max ε

s.t.

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

$$u_k \leq \mathrm{U}[\mathrm{g}(a_j)] + M(1 - x_{jk}) + \varepsilon \qquad k = 1,2,\dots,q-1 \qquad \forall a_j \in C_0 - \{a_j\}$$

$$u_{k-1} \geq \mathrm{U}[\mathrm{g}(a_j)] - M(1 - x_{jk}) + \varepsilon \qquad k = 2,\dots,q \qquad \forall a_j \in C_0 - \{a_j\}$$

$$\sum_{i=1}^{n} \sum_{p=1}^{b_{i-1}} w_{ip} = 1 \qquad i = 1,2,\dots,n \qquad p = 1,2,\dots,b_{i-1}$$

$u_k - u_{k+1} \geq s$ $\qquad k = 1,2, \dots, q-2$

$\sum_{k=1}^{q} x_{jk} = 1$ $\qquad \forall a_j \in C_0 - \{a_j\}$

$u_1 \leq \mathrm{U}[\mathrm{g}(a_j)]$ $\qquad \forall a_j \in C_1$

$u_k \leq \mathrm{U}[\mathrm{g}(a_j)] \leq u_{k-1} - \varepsilon$ $\qquad k = 2, \dots, q-1 \qquad \forall a_j \in C_k$

$\mathrm{U}[\mathrm{g}(a_j)] \leq u_{q-1} - \varepsilon$ $\qquad \forall a_j \in C_q$

$\mathrm{U}[\mathrm{g}(a_j)] \leq u_t - \varepsilon$ $\qquad \forall a_j \in C_q$

$\varepsilon \geq 0$

$w_{ip} \geq 0$ $\qquad \mathrm{i} = 1,2, \dots, \mathrm{n} \qquad \mathrm{p} = 1,2, \dots, b_i$

$x_{jk} \in \{0,1\}$ $\qquad \forall a_j \in C_0 - \{a_j\} \quad k = 1,2, \dots, q$

They argue that if $IP1_{a,t}$ is infeasible than the worst class of alternative a_j is t. That is C_j^w is t. If constraint $\mathrm{U}[\mathrm{g}(a_j)] \leq u_t - \varepsilon$ is altered with $\mathrm{U}[\mathrm{g}(a_j)] \geq u_{t-1}$ then model $IP2_{a,t}$ is formed. This model checks whether the best class of alternative a_j is t. They also solve the LP relaxations of the models $IP1_{a,t}$ and $IP2_{a,t}$ which are formed by relaxing the binary variable x_{jk}. Integer models are stronger than the linear models since there can be feasible solutions for LP which may not be feasible for IP. On the other hand, it is easier to solve a linear problem. Therefore, for the large problem sizes, initially solving LP and solving the IP only if LP is feasible will be a reasonable solution approach.

They solve $IP1_{a,t}$ for a selected alternative. If $IP1_{a,t}$ is infeasible then the worst class the alternative can be assigned is class 1. If it is feasible for class 1, then $IP1_{a,t}$ is solved till t=q or until an infeasible result is obtained. If it is feasible for all t, then they conclude that the worst class of the alternative is q. After that, by starting with the worst class of alternative, $IP2_{a,t}$ is solved until an infeasible solution is obtained to determine the best possible class for alternative. If $IP2_{a,t}$ is feasible for all t, then the best possible class of the alternative is class 1. If the best and worst class of an alternative is the same, the alternative is placed to that class. Otherwise, the DM is asked to place the alternative to one of the classes between its worst and best class. They also suggest a variation in case DM does not want to place the alternative to a single category. The algorithm ends when all the alternatives are assigned to a category.

They make an experiment with the example they use while testing the UTADIS method. They use a real data set that is published by Financial Times. They make the experiment with 81 MBA programs. They try to place he programs into three categories. They present the results according to the number of narrowed-down classes. For the case there is no information about w_{ip} and u_k the algorithm places 31 of the alternatives out of 81 (38% of all alternatives). If they assume that they have information about range of parameters these values improve. In order to test the performance of alternative they solve some additional problems with 50, 100 alternatives and 3 to 5 classes.

3.3.2. Interactive Probabilistic Sorting Methods for Additive Utility Functions

3.3.2.1. Problem Setting

In this section overview of two different interactive probabilistic procedures are presented. The first method is proposed by Buğdacı et al. [18]. The second method is the procedure that we develop. In this section, we give of the common problem setting of these methods.

It is assumed that the DM has an unknown additive utility function and marginal utility of each alternative on each criterion is piecewise linear. In addition, the utility thresholds that partition different classes are unknown. If an alternative is placed in a different class than it really belongs to “a misclassification” occurs. In order to assign alternatives into the classes, probability of an alternative to be placed in each class is calculated. In order to calculate these probabilities linear models are solved. Misclassification threshold determined by the DM represents the maximum acceptable error level. If the threshold is 0, then DM does not allow any misclassifications. An alternative is placed into a class if the probability of being in that class greater than the acceptable misclassification threshold; that is, if the probability of an incorrect assignment is acceptably small. As the value of misclassification threshold increases, the number of misclassified alternatives is expected to increase as well.

Initially, the DM is needed to place an alternative into a class. Then by using this information, for each of the unassigned alternatives, the probability of belonging to each class is calculated. Then alternatives are either assigned probabilistically (by comparing their probability of being in each class with misclassification threshold value) or exactly (with probability one). If there are some unassigned alternatives, then the DM is asked to place another alternative into a class and the probabilities are updated accordingly. Then the remaining alternatives are tried to be assigned. The algorithm terminates when all alternatives are assigned into classes.

At an iteration of the algorithm, C_0 refers to the set of alternatives that are not assigned to any classes yet, C_k refers to the set of alternatives which are known to be in class k, $C_r = C_1 \cup C_2 \cup \ldots \cup C_q$ refers to the set of alternatives which are placed into a class at the current iteration; that is, A= $C_r \cup C_0$. The algorithm starts with $C_r = \emptyset$. DM is asked to place an alternative j' into a class. Then $C_r = \{j'\}$ while $C_0 = A - \{j'\}$. By using the class information of alternatives in C_r, alternatives in C_0 are assigned to appropriate classes either exactly or probabilistically. The assigned alternatives are removed from C_0 and placed to the set they are assigned at the end of the iteration. If there are unassigned alternatives at the end of the iteration, the DM is asked to place a selected alternative to a class. The selection of this alternative will be discussed later. Utilizing this new class information, the remaining alternatives are tried to be assigned iteratively. The algorithm terminates when every alternative is assigned to a class.

Alternatives are placed into the classes by comparing global utility values with utility thresholds that separate consecutive classes. Parameters of the utility function and utility thresholds are unknown. The class information of alternatives in C_r and previous placements by the DM are utilized in order to assign remaining alternatives. In all of the iterations, alternatives should satisfy constraints (7) to (12). Using the notation of Zopounidis and Doumpos [10], a feasible set for the unknown utility thresholds and the utility function parameters is defined by (7) - (12) at every iteration.

$$U[g(a_j)] = \sum_{i=1}^{n} \left(\sum_{p=1}^{r_{ji}-1} w_{ip} + \frac{g_i(a_j) - g_i^{r_{ji}}}{g_i^{r_{ji}+1} - g_i^{r_{ji}}} w_{ir_{ji}} \right) \tag{7}$$

$$U\left[g\left(a_j\right)\right] - u_k \ \geq \ 0 \qquad \forall a_j \in C_k \tag{8}$$

$$U\left[g\left(a_j\right)\right] - u_{k-1}{}^{+} \leq \ -\delta \qquad \forall a_j \in C_k \quad k = 2,3,\dots,q-1 \tag{9}$$

$$\sum_{i=1}^{n} \sum_{p=1}^{b_{i-1}} w_{ip} = 1 \qquad i = 1,2,\dots,n \qquad p = 1,2,\dots,b_{i-1} \tag{10}$$

$$u_k - u_{k+1} \geq s \qquad \forall k = 1,2,\dots,q-2 \tag{11}$$

$$w_{ip} \ \geq 0 \qquad \forall i = 1,2,\dots,n \quad \forall p = 1,2 \dots ,b_{i-1} \tag{12}$$

Constraint (7) defines the global utility of each alternative. Constraints (8) and (9) are written for the alternatives that are already placed by the decision maker or placed with certainty and indicate respectively that an alternative in class C_k, should have greater utility than the threshold u_k (which is the lower bound of class C_k) and should be smaller than u_{k-1}(which is upper bound of class C_k). Constraint (10) implies that subintervals of utility values of all alternatives should sum up to 1. Constraint (11) indicates that the upper utility threshold of a class must be greater than its lower utility threshold. Finally, constraint (12) implies nonnegative weights for each utility subinterval.

In order to place alternatives into the classes linear programs are solved to calculate the probability of each alternative being in each class.

Every iteration starts with selecting an alternative and asking the DM to place it. This means that a constraint of type (8) and a constraint of type (9) are added to the above set of constraints. Solutions of some linear programs with appropriate objectives under constraints (7) - (12) give us information about the ranges of the unknown parameters at the current iteration. Using these ranges, we come up with some estimators of these unknown parameters. With additional assumptions about the distributions of these estimators, we compute the probability of each alternative being in each class. In the next iteration, new constraints of type (8) and (9) are added to this linear program. The new optimal solution possibly changes the parameters of these distributions. So, the probabilities are re-calculated.

The approaches of Buğdacı et al. [18] and our approach in this study differ in the following aspects:

1. The estimated parameters, hence, the estimators and the assumed distributions.

2. The selection method of the alternative to ask the DM to be placed.

3. The stopping condition of the algorithm.

3.3.3.1. Parameter estimation and probability computation in Buğdacı et al. [18]

At every iteration, the unknown parameters are estimated using the current information obtained by the solutions of six linear programs. The first model aims to minimize the difference between utility value of alternative a_j and utility threshold u_k and the second model tries to maximize this difference. The third and fourth models minimize and maximize possible w_{ip} values respectively. Finally, the fifth and sixth models minimize and maximize the range of class threshold u_k values. The linear programs and the random variables used for probability calculations are explained throughout this section.

$LP_1^{j,k}$: min $U\big[g(a_j)\big] - u_k$

$s.t.$

$Constraints$ (7) to (12).

$LP_2^{j,k}$: max $U\big[g(a_j)\big] - u_k$

$s.t.$

$Constraints$ (7) to (12).

Let the optimal objective value of $LP_1^{j,k}$ for alternative a_j and class threshold u_k be $obj_1^*\,(j,k)$ and the optimal objective value of $LP_2^{j,k}$ be $obj_2^*\,(j,k)$.

If $obj_1^*\,(j,k') \geq 0$ for u_k then $obj_1^*\,(j,k) \geq 0$ for k =k'+1, k'+2,…,q-1 and there is no need to solve the minimization problem for k= k'+1, k'+2,…,q-1. It is also true that if $obj_1^*\,(t,k') \geq 0$ then $obj_2^*\,(t,k') \geq 0$ and there is also no need to solve the maximization problem for k= k'+1, k'+2,…,q-1.

If the objective value of maximization problem is negative for alternative a_j and utility threshold u_k, then the best class of a1ternative a_j is k+1. If $obj_2^* \, (j, k') < 0$, there is no need to solve $LP_2^{j,k}$ for alternative a_j and class threshold u_k for k=1,2,…, k'-1. There is also no need to solve the minimization problem for k=1,2,…, k'.

In order to assign the alternatives into the classes without asking the DM, the estimated utility of each alternative as well as the thresholds between the classes are necessary. Both the utility value of the alternative and utility thresholds should be estimated. These unknown quantities are estimated by some estimators obtained from the linear programs with feasible region (7)-(12) as follows: For each parameter, a range is obtained by minimizing and maximizing it over the feasible set of the current iteration. Then the midpoint of this range is used as the estimator of the parameter. A distribution for this estimator is assumed and the probability of an alternative being in each class is computed.

In order to estimate the utility of an alternative, w_{ip} values are estimated first and they are used to estimate the utilities using (7). The ranges for w_{ip} values are obtained solving the following linear programs:

$LP_3^{i,p}$: min w_{ip}

$s.t.$

Constraints (7) to (12)

$LP_4^{i,p}$: max w_{ip}

$s.t.$

Constraints (7) to (12)

Ranges for the threshold values are estimated in the same manner:

LP_5^k: min u_k

$s.t.$

Constraints (7) to (12)

LP_6^k: max u_k

$s.t.$

Constraints (7)to (12)

Assume that the optimal objective value of $LP_3^{i,p}$ is $\underline{W}_{ip}$ and the optimal objective value of $LP_4^{i,p}$ is $\overline{W}_{ip}$. These upper and lower bounds on w_{ip} can be considered as random variables since they are determined by the current set of placed alternatives and that they could change if the current set of placed alternatives were different. An estimator of w_{ip} at the current iteration is

$$\widehat{W}_{ip} = \frac{\underline{W}_{ip} + \overline{W}_{ip}}{2}.$$

If $\widehat{W}_{ip}$ is assumed to be uniformly distributed in that interval $[\underline{W}_{ip}, \overline{W}_{ip}]$ then its variance is estimated as:

$$V(\widehat{W}_{ip}) = \frac{(\overline{W}_{ip} - \underline{W}_{ip})^2}{12}$$

It is further assumed that $\widehat{W}_{ip}$ values are independent for $\forall i = 1,2,\dots,n$ and $\forall p = 1,2 \dots, b_{i-1}$.

As a result, the utility of the alternative a_j can be estimated as:

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

where $z_{ip}(a_j)$ represents :

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

Buğdacı et al. [18] argued that the unbiased estimator of utility $\widehat{U}[g(a_j)]$ is normally distributed, assuming that $\sum_{i=1}^{n} b_i$ is large enough due to the Central Limit Theorem.

As a result of independence of the estimator of $\widehat{W}_{ip}$ and the normality of $\widehat{U}[g(a_j)]$, the variance of unbiased estimator of utility is estimated as:

$$\widehat{V}[\widehat{U}[g(a_j)]] = \sum_{i=1}^{n}\sum_{p=1}^{b_i} z_{ip}(a_j)^2\, V(\widehat{W}_{ip})$$

In order to specify the distribution of $\widehat{U}[g(a_j)]$, it is conditioned on $Y = 1$ where $Y = \sum_{i=1}^{n}\sum_{p=1}^{b_i} \widehat{W}_{ip}$. Due to Central Limit Theorem, Y is assumed to have a normal distribution.

$$E[Y] = \sum_{i=1}^{n}\sum_{p=1}^{b_i} w_{ip}$$

$$\hat{V}[Y] = \sum_{i=1}^{n} \sum_{p=1}^{b_i} \hat{V}(\widehat{W}_{ip})$$

Since the distributions of $\widehat{U}[g(a_j)]$ and Y are normal, their joint distribution is bivariate normal. The marginal variances and means are already estimated. In order to estimate the mean and variance of bivariate distribution firstly covariance and as a result correlation coefficient are calculated. The mean and variance of the bivariate normal distribution are as follows:

$$E\left[\left(\widehat{U}\left[g(a_j)\right] \,\middle|\, Y = 1\right)\right] = E\left[\widehat{U}[g(a_j)]\right] - \rho \frac{V\left(\widehat{U}\left[g(a_j)\right]\right)}{V(Y)} (1 - E(Y))$$

$$V\left[\left(\widehat{U}\left[g(a_j)\right] \,\middle|\, Y = 1\right)\right] = (1 - \rho^2) V(Y)$$

The estimators for class thresholds are found by utilizing linear programs LP_5^k and LP_6^k. Assume that the optimal objective value of LP_5^k is $\underline{u}_k$ and the optimal objective value of LP_6^k is $\overline{u}_k$. An unbiased estimator of u_k at the current iteration can be:

$$\hat{u}_k = \frac{\underline{u}_k + \overline{u}_k}{2}$$

The distribution of $\hat{u}_k$ is assumed to be normal and the variance is assumed to cover 6 standard deviations.

$$V(\hat{u}_k) = \frac{(\overline{u}_k - \underline{u}_k)^2}{36}$$

It is further assumed that $\widehat{W}_{ip}$ and $\hat{u}_k$ are independent for ∀i =1,2, …, n, ∀p =1,2, …, b_i and k= 1,2,…,q.

$D_{kj} = \left(\widehat{U}\left[g(a_j)\right] \,\middle|\, Y = 1\right) - \hat{u}_k$ is an unbiased estimator of $U\left[g(a_j)\right] - u_{k.}$ Let d_{kj} be the current estimate for $D_{kj.}$

To assign alternatives into the classes, probabilities are calculated as follows:

$p(j,k)\colon \hat{P}\left(\left[\left(\widehat{U}\left[g(a_j)\right] \,\middle|\, Y = 1\right) - \hat{u}_k\right] \;\middle|\; obj_1^*\,(j,k) \le \left[\left(\widehat{U}\left[g(a_j)\right] \,\middle|\, Y = 1\right) - \hat{u}_k\right] \le obj_2^*\,(j,k)\right)$

If this probability is sufficiently large, then we assume $U\left[g\left(a_j\right)\right] > u_k$. Whether the probability is sufficiently large is decided by a predetermined critical probability value, τ. If $\tau = 0$, alternatives are placed without any error. If $\tau > 0$, alternatives are placed into the classes probabilistically.

If $p(\mathrm{j}, 1) \ge (1 - \tau)$, then a_j is assigned to C_1.

If $p(\mathrm{j}, \mathrm{k}) \ge (1 - \tau)$, $p(\mathrm{j}, \mathrm{k} - 1) \le \tau$. It is assigned to C_k.

If $p(\mathrm{j}, \mathrm{q} - 1) \le \tau$ then a_j is assigned to C_q.

After alternatives are assigned exactly or probabilistically, they ask the DM to place the alternative about whose class we have the least information. Therefore, they choose this alternative whose calculated probabilities are closest to 0. The algorithm continues till all the alternatives are placed.

3.3.3.2. Parameter estimation and probability computation in our study

In order to assign an alternative into a class, we need to find out between which class thresholds its utility value falls. Alternatives are placed into the classes as follows:

If $U\left[g\left(a_j\right)\right] \ge u_1$, then a_j is in class C_1.

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

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

However, neither utility values nor class thresholds are known. Therefore, we generate an estimator only for $U[g(a_j)] - u_k$, the utility of the alternative a_j and the threshold of class k, class threshold differences and compute the probability of above events using the assumed distribution of the estimator.

One of the most important differences of this study from the study of Buğdacı et al. [18] is what we estimate and which distribution we use for the estimators. In their algorithm they solve three minimization and three maximization problems in order to find the minimum and maximum values of $w_{ip},\ u_k$ and $U[g(a_j)] - u_k$. They assume a distribution between the maximum and minimum range of w_{ip} values in order to find an unbiased estimator for the utility value of each alternative. Then they use these values in order to form the utilities of each alternative. They also solve separate models in order to estimate the class threshold value. Finally, in order to shrink the possible ranges that $U[g(a_j)]$ and u_k may take; they solve the linear models that minimize and maximize the difference between the alternative and the class threshold.

We now introduce how we generate the random variables;

At every iteration, we solve a minimization and a maximization problem for the difference of the utility of each alternative and each class threshold $U[g(a_j)] - u_k$. Then, we consider the estimator of this unknown difference. We assume a distribution for the estimator over the range. Finally, we compute the probability that $U[g(a_j)] - u_k > 0$ using distribution parameters at the current iteration. Alternatives are placed into the classes by comparing these values with misclassification thresholds and make the assignments according to the procedures of the algorithm we use. In addition the positivity/negativity of these estimators will also give an idea about the possible classes of the alternatives. Since, to place an alternative into a class its utility value should be greater than or equal to the class threshold value.

Now we present the computations in detail. Suppose $a_j \in C_0$. To compare the utility of each alternative with each class threshold, the following two models are solved for each unassigned alternative a_j and each class threshold u_k.

LP$_{a,1}$

max $U[g(a_j)] - u_k$

$s.t.$

$$U[g(a_j)] = \sum_{i=1}^{n} \left[\sum_{p=1}^{r_{ji}-1} w_{ip} + \frac{g_i(a_j) - g_i^{r_{ji}}}{g_i^{r_{ji+1}} - g_i^{r_{ji}}} w_{ir_{ji}} \right] \quad \forall a_j \in A \qquad (13)$$

$$U[g(a_j)] - u_k \geq 0 \qquad \forall\, a_j \in C_r \;\; k = 1, 2, \dots, q-1 \qquad (14)$$

$$U[g(a_j)] - u_{k-1} \leq -\delta \qquad \forall\, a_j \in C_r \;\; k = \; 2\,, 3\,, \; \dots, q \qquad (15)$$

$$\sum_{i=1}^{n} \sum_{p=1}^{b_i - 1} w_{ip} = 1 \qquad i = 1,2, \dots, n \qquad p = 1,2, \dots, b_{i-1} \qquad (16)$$

$$u_k - u_{k+1} \geq s \qquad \forall k = 1, 2, \dots, q-2 \qquad (17)$$

$$w_{ip} \geq 0 \qquad \forall i = 1, 2, \dots, n \qquad \forall p = 1, 2, \dots, b_i \qquad (18)$$

LP$_{a,2}$

min $U\left[g(a_j)\right] - u_k$

$s.t.$

(13) - (18)

Let the optimal objective value of $LP_{a,1}$ be $obj *_{a,1}$ and the optimal objective value of $LP_{a,2}$ be $obj *_{a,2}$.

If $obj *_{a,1} < 0$ is satisfied for an alternative-class threshold pair then $obj *_{a,2} < 0$ for the same pair.

Obtaining a negative objective value in the maximization problem LP$_{a,1}$ implies that the utility of alternative does not satisfy the threshold of the class it is compared with, even at its maximum possible value. Hence the best possible class for that alternative to be placed in is k-1.

Obtaining a positive objective value in the minimization problem LP$_{3,2}$ implies that the utility of the alternative satisfies the threshold of the class it is compared with even at its minimum possible value. Hence the worst possible class for that alternative to be placed is class k.

3.3.3.3. Probability computation under different distributions

Consider the programs we present in the previous section.

LP$_{a,1}$

max $U\left[g(a_j)\right] - u_k$

$s.t.$

(13)- (18)

LP$_{a,2}$

max $U\left[g(a_j)\right] - u_k$

$s.t.$

(13)- (18)

Let the optimal objective value of $LP_{a,1}$ be $\underline{U\left[g\left(a_j\right)\right] - u_k}$ and the optimal objective value of $LP_{a,2}$ be $\overline{U\left[g\left(a_j\right)\right] - u_k}$. These values are assumed to be observations of a random variable since new constraints of type (14) and (15) are added to the model representing the placement done by the DM at the current iteration and different alternatives asked to the DM may result in different optimal solutions. We assume that the random estimator $\widehat{U\left[g(a_j)\right]} - u_k$ of $U\left[g(a_j)\right] - u_k$ has a certain distribution in the interval $[\underline{U\left[g\left(a_j\right)\right] - u_k}, \overline{U\left[g\left(a_j\right)\right] - u_k}]$ and compute the probability that $\widehat{U\left[g(a_j)\right]} - u_k \geq 0.$. We consider trapezoidal, uniform and triangular distributions as three alternatives.

3.3.3.3.1. Trapezoidal Distribution

To calculate the probability of each alternative being in each class, we fit a trapezoidal distribution for the maximum difference between the utility value of an alternative and class threshold of each class and the minimum difference between the utility value of an alternative and class threshold of each class, i.e. for alternative a_j and class C_k we assume a trapezoidal distribution over $[\underline{U\left[g\left(a_j\right)\right] - u_k}, \overline{U\left[g\left(a_j\right)\right] - u_k}]$.

We assume a symmetric trapezoidal distribution. The assumptions that we have made and the calculation method is represented below. The notation that we use is represented in the figure below.

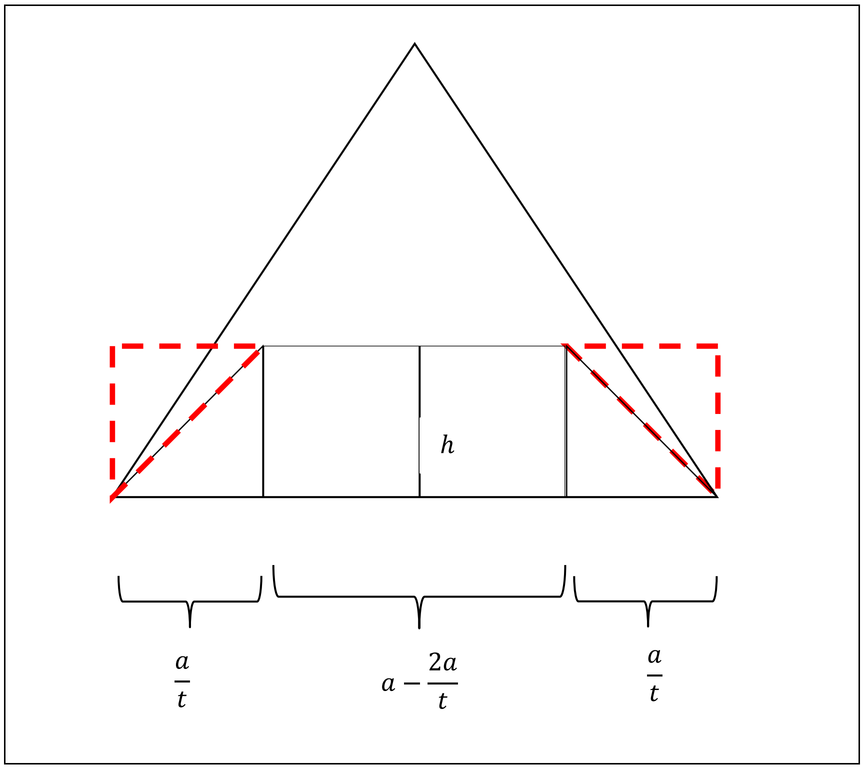

Figure 3.1. Trapezoidal Distribution

For t ≥ 2 (For t =2 it transforms into triangular distribution)

$$a - \frac{2a}{t} . h + h . \frac{a}{t} = 1$$

$$h = \frac{t}{a(t-1)}$$

For our calculations we assume t = 3 and utilize a symmetric trapezoidal distribution. The calculation of probabilities is shown below:

Let x = $\overline{U[g(a_j)] - u_k}$. and y=$U[g(a_j)] - u_k$

$$\begin{cases} 1 & if \quad x \geq a \\ 1 - \dfrac{t^2x^2}{2t(y-x)^2} & if \quad x \leq 0 \text{ and } tx + \dfrac{y-x}{t} \geq 0 \\ 1 - \dfrac{x-y-2tx}{2(y-x)(t-1)} & if \quad tx + \dfrac{y-x}{t} \leq 0 \text{ and } \dfrac{y(t-1)+x}{t} > 0 \\ \dfrac{t^2y^2}{2(t-1)(y-x)^2} & if \quad y > 0 \text{ and } \dfrac{y(t-1)+x}{t} \leq 0 \\ 0 & if \quad y \leq 0 \end{cases}$$

3.3.3.3.2. Uniform Distribution

We now assume that the difference between utility of each alternative and utility threshold for each class is distributed uniformly between its maximum and minimum values.

In order to calculate the probability $p(j,k)$= $\hat{P} = [U[g(a_j)] - u_k \geq 0]$

let x =$U[g(a_j)] - u_k$ and y=$\overline{U[g(a_j)] - u_k}$].

$$p(j,k) = \begin{cases} 1 & if\ x > 0 \\ \dfrac{(y-0)}{(y-x)} & if\ x \leq 0, \quad y > 0 \\ 0 & if\ y \leq 0 \end{cases}$$

We show that when we assume a uniform distribution the maximum difference increases and the minimum difference decreases. Hence, the values coincide at some point and it is guaranteed that all alternatives are assigned to a class.

In Figure 3.2., the relations between maximum and minimum differences of utilities and class thresholds for a three category case are represented with respect to the different positions of “0”where:

$a = Min(U[g(a_j)] - u_1)$

$b = Min(U[g(a_j)] - u_2)$

$c = Max(U[g(a_j)] - u_1)$

$d = Max(U[g(a_j)] - u_2)$

and the numbered areas are used in order to represent possible positions of zero.

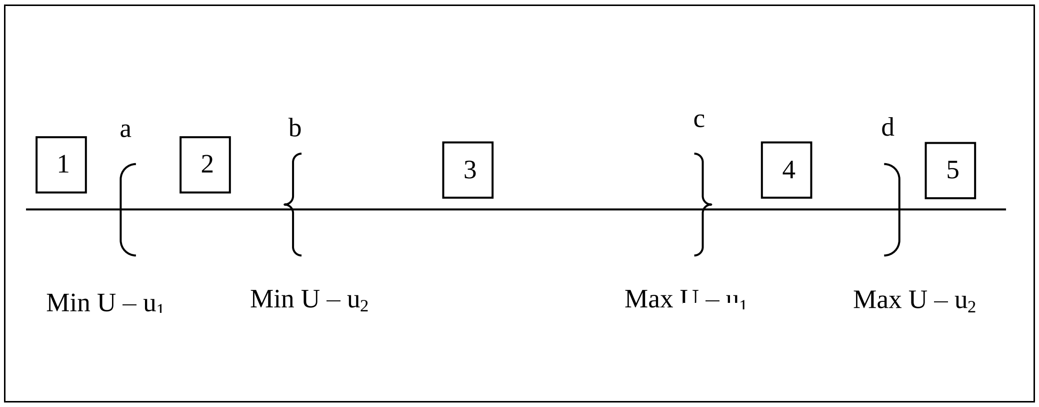

Figure 3.2. Positions of $\boldsymbol{a, b, c, d}$ and 0

In addition, the following probabilities are defined:

$$p(j,1) = P\ (\widehat{U[g(a_j)]} - u_1 \geq 0)$$

$$p(j,2) = P\ (U[g(\widehat{a_j})] - u_2 \geq 0)$$

$$p(j,3) = P\ \ (U[g(\widehat{a_j})] - u_2\ \leq 0)$$

Before presenting the propositions the change of probabilities with respect to five possible places of “0” is represented below:

Property 1: If position of 0 is $\boxed{1}$ then,

a and b are both greater than zero. Since the minimum possible value for $U\big[g\big(a_j\big)\big] - u_1$ is $0,$ it is for sure that $P\ (U\big[\widehat{g\big(a_j\big)\big] -}\, u_1 \geq 0) = 1$. Hence, a_j is placed into class 1 exactly.

Property 2: If position of 0 is $\boxed{2}$ then,

b is greater than zero. Since the minimum possible value for $U\big[g\big(a_j\big)\big] - u_2$ is $0,$ it is for sure that $P\ (U\big[\widehat{g\big(a_j\big)\big] -}\, u_2 \geq 0) = 1$. Hence, a_j is placed into class 2 exactly.

Property 3: If position of 0 is $\boxed{3}$ then,

a_j could be placed in one of the three classes with probabilities $p(j,1) > 0,\ p(j,2) > 0$ and $p(j,3) > 0$

Property 4: If position of 0 is $\boxed{4}$ then,

c is less than zero. Since the maximum possible value for $U\big[g\big(a_j\big)\big] - u_1 \leq 0$, it is for sure that $P\ (U\big[\widehat{g\big(a_j\big)\big] -}\, u_1 \geq 0) = 0$. Hence, a_j is placed into either class 2 or class 3 with $p(j,2) > 0$ and $p(j,3) > 0$

Property 5: If position of 0 is $\boxed{5}$ then,

a and d are both less than zero. Since the maximum possible value for $U\big[g\big(a_j\big)\big] - u_2 \leq 0$, it is for sure that $P\ (U\big[\widehat{g\big(a_j\big)\big] -}\, u_1 \geq 0) = 0$. Hence, a_j is placed into class 3 exactly.

Proposition 1: Given $d' > d$, where the position of "0" is given as $\boxed{3}$

$d = Max(U\big[g\big(a_j\big)\big] - u_2)$

$d' = Max(U[g(a_k)] - u_2)$

$P\ (U[g(\widehat{a_j)]} - u_2 \geq 0)\ >\ P\ (U[g(\widehat{a_k)]} - u_2 \geq 0)$

Proof:

$$\widehat{\mathrm{P}}\ (\ U[g(a_k)] >\ \mathrm{u}_2) -\ \widehat{\mathrm{P}}\ \left(\ U\left[g\left(a_j\right)\right] >\ \mathrm{u}_2\right) = \frac{d'}{d' - b} - \frac{d}{d - b}$$

$$= \frac{dd' - d'b - dd' + bd}{(d - b)(c - a)}\ \geq 0$$

□

Proposition 2: Given $a' < a$, where the position of "0" is given as $\boxed{3}$

$a = Min(U\left[g\left(a_j\right)\right] - u_1)$

$a' = Min(U[g(a_k)] - u_1)$

$\hat{P}(U[g(a_k)] - u_1 \geq 0)\ >\ \widehat{\mathrm{P}}\ (U\left[g\left(a_j\right)\right] - u_1 \leq 0)$

Proof:

$$\widehat{\mathrm{P}}\ (\ U[g(a_k)] >\ \mathrm{u}_1) -\ \widehat{\mathrm{P}}\ \left(\ U\left[g\left(a_j\right)\right] >\ \mathrm{u}_1\right) = \frac{c}{c - a'} - \frac{c}{c - a}$$

$$= \frac{-ac + a'c}{(c - a')(c - a)}\ \leq 0$$

□

Proposition 3: Given that the position of "0" is given as $\boxed{3}$

$\widehat{\mathrm{P}}\ (\ \mathrm{U} >\ \mathrm{u}_2) -\ \widehat{\mathrm{P}}\ (\ \mathrm{U} >\ \mathrm{u}_1)\ \geq 0$

Proof:

$$\widehat{\mathrm{P}}\,(\mathrm{U} > \mathrm{u}_2) = \frac{d}{d-b}$$

$$\widehat{\mathrm{P}}\,(\mathrm{U} > \mathrm{u}_1) = \frac{c}{c-a}$$

$$\widehat{\mathrm{P}}\,(\mathrm{U} > \mathrm{u}_2) - \widehat{\mathrm{P}}\,(\mathrm{U} > \mathrm{u}_1) = \frac{d}{d-b} - \frac{c}{c-a}$$

$$= \frac{dc - ad + dc - bc}{(d-b)(c-a)}$$

$$= \frac{2dc - ad - bc}{(d-b)(c-a)} \geq 0$$

□

3.3.3.3.3. Triangular Distribution

If the differences between maximum and minimum utility value of an alternative and utility threshold of each class are independent, then this difference is either in triangular or trapezoidal form.

Let x = max $(U_i - u_k)$ and $y = \min\ (U_i - u_k)$

$P[U[g\widehat{(a_j)}] - u_k\ \geq 0]$ is found as follows:

$$p(j,k) = \begin{cases} 1 & \text{if } \mathrm{x} > 0 \\ 1 - 2\dfrac{\mathrm{x}^2}{(\mathrm{y}-\mathrm{x})^2} & \text{if } \mathrm{x} \leq 0, \quad \dfrac{\mathrm{x}+\mathrm{y}}{2} > 0 \\ 2\dfrac{\mathrm{y}^2}{(\mathrm{y}-\mathrm{x})^2} & \text{if } \dfrac{\mathrm{x}+\mathrm{y}}{2} \leq 0, \quad \mathrm{y} > 0 \\ 0 & \text{if } \mathrm{y} \leq 0 \end{cases}$$

Proposition 4:

Given $u_1 > \ u_2$

Max $(U - u_1)$ < Max $(U - u_2)$

Min $(U - u_1)$ < Max $(U - u_2)$

Proof:

Suppose that X and Y are random variables. X refers to maximum or minimum possible values for $(U - u_1)$ and Y refers to maximum or minimum possible values for $(U - u_2)$. Y is obtained from X by shifting the interval of X [a,b] to [a’, b’] as shown below. The ranges of both variables are defined below by (1). Both the lower and the upper bound of Y is greater than the lower and upper bound of X respectively which is represented in (2)

$$\mathrm{X} \in [\mathrm{a}, \mathrm{b}] \qquad \mathrm{Y} \in [\mathrm{a}', \mathrm{b}'] \qquad (1)$$

$$a' > a \qquad b > \ a \qquad b' > b \qquad b' > a' \qquad (2)$$

$$\mathrm{Y} = \frac{X + A}{B} \qquad \mathrm{a}' = \frac{a + A}{B} \qquad \mathrm{b}' = \frac{b + A}{B}$$

$$\mathrm{B} = \frac{b - a}{b' - a'} \qquad A = a'\left(\frac{b - a}{b' - a'}\right) - a$$

It is assumed that A and B is greater than 0 for the cases we consider.

Since, $b > \ a$ and $b' > a'$ $\ \mathrm{B} > 0$ is always satisfied.

In order for $A > 0$ to be satisfied $a'b - ab' > 0$ should be satisfied.

All possible values for a, b, a', b' are represented in the table below.

Due to the relations between a, b, a' and b' provided in (2), cases 3, 4, 7, 8, 9, 10, 11, 14 and 16 (see Table 1) are not possible. In addition, we will calculate the probabilities for the intervals at least one of which include 0. If none of the intervals include 0, then the corresponding alternatives will not be placed probabilistically but they are placed exactly. Hence we do not need to examine cases 1, 6 and 13 (see Table 1)

For the cases 2, 5 and 12 we will show that $A > 0$ is satisfied. (It is already shown that $B > 0$ is satisfied for all a, b, a', b'

Case 2:

$a < 0,\ a' > 0,\ a' > a$

$b > 0,\ b' > 0,\ b' > b$

In order for $A > 0$ to be satisfied $a'b - ab' > 0$ should be satisfied.

For Case 2,

$a'b > 0$ and $-ab' > 0$

Hence, $A > 0$ is always satisfied.

Case 5:

$a < 0,\ a' < 0,\ a' > a$

$b > 0,\ b' > 0, b' > b$

In order for $A > 0$ to be satisfied $a'b - ab' > 0$ should be satisfied.

For Case 5,

$a'b < 0$ and $-ab' > 0$

Since $|-ab'| > \ |a'b| \ \ A > 0$ is always satisfied.

Case 12:

$a < 0, \ a' < 0, \ a' > a$

$b < 0, \ b' > 0, \ b' > b$

In order for $A > 0$ to be satisfied $a'b - ab' > 0$ should be satisfied.

For Case 5,

$a'b > 0$ and $-ab' > 0$

Hence, $A > 0$ is always satisfied.

Therefore, $A > 0$ is satisfied for all possible cases.

Table 3.1: Possible Values of a, b, a', b'

Cases	a	b	a'	b'	**Positivity**	**Suitability**
Case 1	-	-	-	-	✓	Since none of the intervals include zero, probability will not be calculated.
Case 2	-	+	+	+	✓	Positivity of A should be examined
Case 3	-	+	-	-	**x**	Lower and upper limits do not satisfy the relationship between a, b , a’, b’
Case 4	-	+	+	-	**x**	Lower and upper limits do not satisfy the relationship between a, b , a’, b’
Case 5	-	+	-	+	✓	Positivity of A should be examined
Case 6	-	-	+	+	✓	Since none of the intervals include zero, probability will not be calculated
Case 7	-	-	+	-	**x**	Lower and upper limits do not satisfy the relationship between a, b , a’, b’
Case 8	+	+	-	-	**x**	Lower and upper limits do not satisfy the relationship between a, b , a’, b’
Case 9	+	-	-	+	**x**	Lower and upper limits do not satisfy the relationship between a, b , a’, b’
Case 10	+	-	+	-	**x**	Lower and upper limits do not satisfy the relationship between a, b , a’, b’
Case 11	+	-	+	+	**x**	Lower and upper limits do not satisfy the relationship between a, b , a’, b’
Case 12	-	-	-	+	✓	Positivity of A should be examined
Case 13	+	+	+	+	✓	Since none of the intervals include zero, probability will not be calculated
Case 14	+	+	+	-	**x**	Lower and upper limits do not satisfy the relationship between a, b , a’, b’
Case 15	+	+	-	+	**x**	Lower and upper limits do not satisfy the relationship between a, b , a’, b’
Case 16	+	+	-	-	**x**	Lower and upper limits do not satisfy the relationship between a, b , a’, b’

In the light of this information, we claim that the cumulative distribution function of every Y value is below that of every X value.

$$F_Y(y) = P\ (Y \leq y) = P\left(\frac{x + a'\dfrac{b-a}{b'-a'} - a}{\dfrac{b-a}{b'-a'}} \ \leq y\right)$$

$$P\left(X \ \leq \frac{b-a}{b'-a'}y - a\frac{b-a}{b'-a'} + a \leq y\right)$$

$$F_X\left(\frac{b-a}{b'-a'}y - a\frac{b-a}{b'-a'} + a\right)$$

$$F_Y(0) = \ F_X\ \left(-a\frac{b-a}{b'-a'} + a\right) = \ F_X(-A) \leq \ F_X(0)$$

$$F_Y(y) \leq F_X(y)$$

□

It is hard to say that one distribution is better over the other. It depends on the data set. However, as a general trend we expect that the probability of being in the middle is higher than probability of being at the extremes of the allowable ranges. Therefore, we suggest using Trapezoidal Distribution.

3.3.3.4. Computation of the probabilities and assigning alternatives into classes

Up to this point, we computed the probability that alternative a_j is in class C_k for all $j = 1,2,\dots,m$ and $k = 1,2,\dots,q$. After probabilities are calculated alternatives are placed into the best, the intermediate or the worst classes.

The method that we use to calculate probabilities for assigning alternatives into the classes is also different from Buğdacı et al. [18]. In order to place the alternatives into the intermediate classes they compare the utility of being in a better or worse class separately. We calculate the probability of being in an intermediate class in a different manner. The details are explained below using the notation $p(j,k)$ = $\hat{P}[U[g(a_j)] - u_k \geq 0]$:

-Probability that the alternative a_j is in the best class C_1 is $p(j,1)$.

- Probability that the alternative a_j is in an intermediate class C_k is

$\hat{P}[u_k \leq U[g(a_j)] \leq u_{k-1}]$ = $\hat{P}[U[g(a_j)] - u_k \geq 0]$- $\hat{P}[U[g(a_j)] - u_{k-1} \geq 0]$

= $p(j,k) - p(j,k-1)$.

- Probability that the alternative a_j is in the worst class C_q is $1 - p(j,q-1)$.

Alternatives are placed into the classes if the probability of an alternative being wrongly placed into a class is less than the misclassification threshold *th*. By decreasing misclassification threshold *th* to 0, alternatives can be placed into their true classes with probability one without any misclassification. If $p(j,k) = 1$, alternative a_j is better than class threshold u_k for all set of possible parameters. On the other hand if $p(j,k) = 0$, then alternative a_j is worse than class threshold u_k for all set of possible parameters.

Alternatives can be placed exactly with $p(j,k) = 1 \; or \; p(j,k) = 0$ if the following conditions are satisfied.

If $p(j,1) = 1$, then a_j is in C_1. Since C_1 is the best possible class an alternative can be placed in and alternative a_j is warranted to be better than u_1 for all set of possible parameters.

If $p(j,k) = 1$ and $p(j,k-1) = 0$, then a_j is in C_k. Since a_j is warranted to be better than u_{k-1} and there is no set or parameters that make a_j worse than u_k.

If $p(j, q-1) = 0$, then a_j is in C_q. Since C_q is the worst class an alternative can be placed and the probability for alternative a_j to be in a better class is 0, it is warranted to be in C_q.

3.3.3.5. Selection of Alternatives to Ask the Decision Maker in Buğdacı et al. [18]

They select the alternative whose probability of being in a class is closest to 0.5. Buğdacı et al. [18] proposed the following method for selection of alternatives to select the alternative that is asked to the DM. $d_{j,k}$ represents the difference between $p(j,k)$ and 0.5. a_j^s represents the alternative selected.

$$d_{j,k} = \min_k |p(j,k) - 0.5| \; for \; k = 1,2,\dots,q-1$$

$$a_j^s = \{\operatorname*{argmin}_i \; d_{j,k}\} \qquad \forall\, k = 1,2,\dots,q-1, \qquad \forall a_i^s \in C_0$$

They suggest that the closest probability value to 0.5 is the one we have the least information above. This method fails to compare the overall probability. It just compares the probabilities of classes on the edge with 0.5, failing to analyze middle classes.

Hence, we worked on several methods in order to select the alternative that we ask the DM to place. Our aim is to ask the most “ambiguous” alternative or the alternative whose placement provides “valuable” information. In that manner, we are going to ask the alternative that we have least information about. By doing so we expect to ask for fewer alternatives compared to asking the DM to place an alternative into a class randomly. We explain all the methods we use briefly below:

<u>*Method 1: Asking the Alternative whose successive probabilities are close to 0.5*</u>

For each $a_j \in C_0$ we compare the absolute difference between probability of an alternative to be placed in successive classes and 0.5. If the probability that utility of an alternative exceeds class threshold for two adjacent classes is close to 0.5, then the class that alternative will be placed is very unclear. By selecting the alternative about which we have less information we aim to ask fewer number of questions to the DM. Let

$p(j,k) =$ Probability that utility value of alternative a_j is greater than threshold u_k

$d_{j,k} = |p(j,k) - 0.5|$

$d_{j,k+1} = |p(j,k+1) - 0.5|$

$a_j^s = \{\min_i(|d_k|, |d_{k+1}|)\} \qquad \forall\, k = 1,2,\dots,q-1, \qquad \forall a_j^s \in C_0$

<u>*Method 2: Asking the alternative if all class probabilities are close to 0.5*</u>

We suggest that to place an alternative into a class, not only the probability of adjacent classes but also for all possible classes we may not have enough information. Hence we also suggest whether the probability that utility of an alternative exceeds class threshold for all classes is close to 0.5. For instance for a three class case the probabilities of 0.33 for all three classes will be the most unclear case. To consider this case, we select the alternative a_i^s as follows:

$d_{j,k,k+1,..,q} = (|(p(j,k) - 0.5)|, |(p(j,k+1) - 0.5)|, \dots, |(p(j,q) - 0.5)|)$

$a_j^s = \min_j\left(d_{j,k,k+1,..,q}\right) \qquad \forall a_j^s \in C_0$

<u>*Method 3: Asking the Alternative whose minimum distance to 1 is Maximum*</u>

If $p(j,k) = 1$ we know that alternative a_i is placed to class k exactly. Furthermore, if $p(j,k) = 0$ we know that alternative a_i is not placed to class k exactly. However, if $0 < p(j,k) < 1,$ then alternatives are placed probabilistically. Since, our aim is to ask the alternative for which we have the least information we ask the alternative whose probability of exceeding a class threshold is not close to 1. Hence, we minimize the maximum distance from 1 as follows:

$$a_j^s = \min\left(\max_j\left(d_{j,k} = \left(1 - p(j,k)\right)\right)\right) \qquad \forall a_j^s \in C_0$$

3.3.3.6. Stopping Condition Comparison

We utilize the notation of Buğdacı et al. [18] for our algorithm. We changed the algorithm proposed in several ways to improve the results. There are two versions of the algorithm. Buğdacı et al. [18] proposed the first one. We developed the second version. This constitutes the third difference between two studies.

In both versions, an iteration starts by asking DM to place an alternative. Then, we try to place as many alternatives as possible into classes either with certainty (with probability 1) or with some positive probability. Initially, we ask the DM to place at least one alternative. By using this information we try to assign the remaining alternatives into the classes either with certainty (with probability 1) or probabilistically (with a probability less than 1). We place the alternatives by comparing these probabilities with a predetermined critical probability value (maximum acceptable misclassification level). If there are still alternatives that are not placed then the next iteration starts. In the first version, once an alternative is placed in a class, it stays in that class in the remaining iterations. The algorithm stops when all the alternatives are placed. In the second version, the alternatives are also placed into the classes iteratively but the alternatives that are placed probabilistically can change class in the following iterations. Clearly, the alternatives placed by the DM or the alternatives placed with certainty stay in the same class during the iterations. If all the alternatives cannot be placed into a class in an iteration, the DM is asked to place another alternative into a class and probabilities are recalculated. The procedure continues until all the alternatives can be placed into classes either exactly or probabilistically at the same time. With this approach we expect to make fewer misclassifications but possibly more iterations compared to Method 1.

3.3.3.7. Calculating the Expected Number of Misclassifications

In order to evaluate the performance of algorithms we calculated the expected number of misclassifications and compared them with the realized number of misclassifications. Expected number of misclassifications is calculated as follows:

For each alternative that is not placed by the DM, we first calculate the probability that the alternative is not in the class it is actually placed by the algorithm. Let k_i be the class number that the alternative a_j is assigned by the algorithm and $p(i, k_i) = \widehat{\text{P}}\ (U[g(a_j)] - u_{k_i} \geq 0)$ be the probability of alternative a_i to the class it is placed. Then,

Probability of misclassification of alternative $a_i = 1 - p(i, k_i)$

E[Number of misclassifications] $=\sum_i (1 - p(i, k_i))$.

3.3.3.8. Placing Alternatives by Solving Minimization and Maximization Problems as Mixed Integer Programs

There is a third method to decrease the possible classes for alternatives. In order to achieve this, maximization and minimization objectives in LP$_{a,1}$ and LP$_{a,2}$ are solved as $IP^1_{(a,j)}$ and $IP^2_{(a,j)}$. In addition to the constraints (13)-(18), we add constraints (19)-(23) with binary variables z_l, p_l, t_l and solve the model as a mixed integer program.

These binary variables force each alternative to be placed in the same class for both minimization and maximization case. By doing so we aim to decrease the possible classes of the alternatives in the first place if the utility values in both models are between the same class thresholds.

$IP^1_{(a,j)}$

[Max ($U\left[g\left(a_j\right)\right] - u_k)\big]$

and

$IP^2_{(a,j)}$

[Min ($U\left[g\left(a_j\right)\right] - u_k)\big]$

$$\left.\begin{array}{l} U[g(a_l)] - u_k + 10z_l \leq 9.99 \\ U[g(a_l)] - u_{k+1} - 10z_l \geq -10 \end{array}\right\} \quad \text{(19) Constraints for placing class 2}$$

$U[g(a_l)] - u_1 - 10p_l \geq -10$ (20) Constraints for placing class 1

$U[g(a_l)] - u_2 + 10t_l \leq 9.99$ (21) Constraints for placing class 3

$z_l + p_l + t_l = 1$ (22)

$z_l, p_l, t_l \in \{0,1\}$ (23)

However, these models do not improve the results very much. Since, it is harder to solve integer problems we continue with the linear programs throughout the study.

3.3.3.9. The Algorithms

In order to place the alternatives into the classes we have developed two algorithms. The first algorithm places an alternative into a class whenever it satisfies the class threshold to be placed in that class. The second algorithm, on the other hand places the alternatives into the classes if all the alternatives can be placed at the same time, otherwise none of the alternatives will be assigned.

Below, we present the steps of both of our algorithms.

Algorithm 1: Placing Alternatives when They Satisfy the Misclassification Threshold

<u>Step 0:</u> Ask the DM to place an alternative selected by him/her from a set of alternatives $a_1, a_2, \dots, a_m$ to place in a class. Let k be the index for class thresholds and $C_1, C_2, \dots, C_q$ be the set of available classes. Put the alternative that is placed by the DM to C_r. Put the alternatives whose classes are unknown to C_0. Let h represent the number of alternatives assigned.

<u>Step 1:</u> Set k=1 and h=1. Go to step 2.

<u>Step 2:</u> Solve $LP_{a,1}\,(j,k)$ for each $a_j \in C_0$ and for each u_k.

<u>Step 3:</u> Solve $LP_{a,2}\,(j,k)$ for each $a_j \in C_0$ and for each u_k.

<u>Step 4:</u> Find $p(j,k)$ by utilizing the optimal objective values $obj *_{a,1}$ and $obj *_{a,2}$ and by utilizing random variables $[\underline{U[g(a_j)] - u_k}, \overline{U[g(a_j)] - u_k}]$. Go to Step 5.

<u>Step 5:</u> If $p(j,1) \geq (1\text{-}\ \delta)$ place a_j into C_1. Set h=h+1

If $p(j,k) \geq (1\text{-}\ \delta)$ and $p(j,k-1) \leq \delta$ assign alternative a_j into $C_{k.}$ Set h=h+1.

If $p(j,q) \leq th$ assign alternative a_j into $C_{q.}$ Set h=h+1.

<u>Step 6:</u> If $|C_o| = 0$, go to Step 7

If $|C_o| > 0$, and h > 0 go to Step 2.

If $|C_o| > 0$, and h = 0 find a_j^s and present the DM the possible classes of a_j^s. Ask the DM to place the alternative in one of the presented classes.

If $|C_o| > 0$ go to Step 2. If $|C_o| = 0$, go to Step 7.

<u>Step 7:</u> Present alternatives in C_k to the DM

Algorithm 2: Placing Alternatives when They All Satisfy the Misclassification Threshold

<u>Step 0:</u> Ask the DM to place an alternative selected by him/her from a set of alternatives $a_1, a_2, \ldots, a_m$ to place in a class. Let k be the index for class thresholds and $C_1, C_2, \ldots, C_q$ be the set of available classes. Put the alternative that is placed by the DM to C_r. Put the alternatives whose classes are unknown to C_0. Let h represent the number of alternatives assigned.

<u>Step 1:</u> Set k=1 and h=1. Go to step 2.

<u>Step 2:</u> Solve $LP_{a,1}(j,k)$ for each $a_j \in C_0$ and for each u_k.

<u>Step 3:</u> Solve $LP_{a,2}(j,k)$ for each $a_j \in C_0$ and for each u_k.

<u>Step 4:</u> Find $p(j,k)$ by utilizing the optimal objective values $obj *_{a,1}$ and $obj *_{a,2}$ and by utilizing random variables $[\underline{U[g(a_j)] - u_k}, \overline{U[g(a_j)] - u_k}]$. Go to Step 5.

<u>Step 5:</u> If all the alternatives satisfy the thresholds defined below, alternatives are placed to the respective classes. If at least one alternative cannot be placed, then none of the alternatives are assigned into the classes.

If $p(j,1) \geq (1-\delta)$ place a_j into C_1. Set h=h+1

If $p(j,k) - p(j,k-1) \geq (1-\delta)$ assign alternative a_j into $C_{k.}$ Set h=h+1.

If $p(j,q-1) \leq \delta$ assign alternative a_j into $C_{q.}$ Set h=h+1.

<u>Step 6:</u> If $|C_o| = 0$, go to Step 7

If $|C_o| > 0$ find a_j^s and ask the DM to place the alternative into one of the presented classes. If $|C_o| > 0$ go to step 2. If $|C_o| = 0$, go to Step 7.

<u>Step 7:</u> Present alternatives in C_k to the DM

3.3.3.9. Comparison of Our Method with Buğdacı et al. [18]

We develop a new interactive probabilistic method because we think that the model can be improved for the following:

1. The linear models
2. Probability calculation and estimated random variables
3. The selection method of the alternative to ask the DM
4. The stopping condition of the algorithm.

The first difference between our method and Buğdacı et al. [18] is the number of linear models solved. In their method they solve six different models that minimize and maximize the range of values for $w_{ip},\ u_k$ and $U[g(a_j)]] - u_k$. Instead of trying to find range of utilities and class threshold separately and narrowing down the possible range of values by finding the range of their difference, we directly find the range for$U[g(a_j)] - u_k$ difference. Therefore we only solve two linear models compared to six models solved in Buğdacı et al.[18]. Solving fewer number of linear models makes our model more effective.

As a second difference, since the models solved are different, the random variables used for probability calculations are also different. They use different random variables for w_{ip} and u_k. Then based on properties of w_{ip} they develop a different random variable for $U[g(a_j)]$. Then, they calculate the probable values for utility and class threshold separately and compare these values to place alternatives into the classes. In our method we directly generate a random variable for the utility value and class threshold differences and fit the suitable probabilistic distribution directly this value. Since, the probabilities are calculated for only one random variable for our method, we can conclude that it is more effective in probability calculations.

Another difference in terms of probabilities is for the placement of alternatives into the middle classes. In Buğdacı et al. [18], the probability of an alternative being in a middle class is compared with upper and lower bound of the class separately. However, this method fails to compare the overall probability. In order to deal with it we define the probability of belonging to a middle class in a different manner as explained in Section 3.3.3.4.

For the alternatives, that are selected to ask DM for placement should be the one that we have least information. As explained in Section 3.3.3.5. in Buğdacı et al.[18] overall probabilities are failed to be compared for middle classes. Therefore, we use the probability calculation method that we previously explain and we compare the class thresholds of only successive classes for all combinations. Since this is the most ambiguous case for placement.

The final difference is in terms of the stopping condition for algorithm. We think that if alternatives are placed into the classes only if all the alternatives satisfy the condition, then misclassification is expected to decrease to some extent. Since alternatives that are placed in a different class than it belongs to can be placed into the correct class at the end of the algorithm. However, this algorithm is expected to continue longer compared to model of Buğdacı et al.[18].

The first experiment that is mentioned in Section 4.1., is also performed by Buğdacı et al. Therefore, we compare the results for this experiment with Buğdacı et al.[18]i.

CHAPTER 4

EXPERIMENTS AND RESULTS

We have developed two probabilistic and interactive multi-criteria sorting algorithms in order to assign alternatives in the preference ordered classes. We made several experiments in order to test the performances of both algorithms. In order to calculate the probability of alternatives to be placed in each of the classes we fit uniform, symmetric triangular and symmetric trapezoidal distributions.

In all our experiments we need to simulate the DM by an underlying utility function and the utility values that separate the preference classes. The additive utility function of the DM is defined by w_{ip} values. The class borders are u_k values. w_{ip} represents the utility values of subintervals p on each criterion i. In order to form the underlying utility function of the DM we randomly generate a set of w_{ip} and u_k values. When DM is asked to place an alternative, its utility is computed according to (1) using these w_{ip} values and it is placed into a class comparing with u_k values.

Firstly, we utilized our algorithm on a data set that is developed by Köksalan and Özpeynirci [8]. They have used the top MBA programs data by Financial Times. Rankings of top 100 global MBA programs are published by Financial Times every year. The ranking is made through three main criteria, namely: “alumni career progress”, “diversity” and “idea generation”. Köksalan et al. [9] use the data published in 2005 and produce numerical scores for 81 of the MBA programs. Köksalan and Özpeynirci utilize these data in order to sort these 81 MBA programs based on three main criteria. They try to sort these MBA programs into 3 classes.

Secondly, we utilized our algorithm on a data set that is published by TÜBİTAK. Rankings of top 50 universities based on their development index are tried to be classified. Since the data for all criteria are in continuous scale, we use this data set directly. There are 5 main criteria which consist of 23 sub-criteria. The five main criteria are namely; "scientific and technological research", "intellectual properties", "collaboration, cooperation and interaction", "entrepreneurship and innovation culture" and "economical contribution and commercialization". We try to sort alternatives into 3 classes.

Thirdly, we have generated 500 random alternatives and try to sort them into five classes. While generating the criteria values we have utilized the range of Financial Times for each criterion.

Lastly, we implemented our algorithm on a data set that is published by US News Report in 2014. They try to rank the hospitals that are best in cancer treatment. We utilize this data in order to sort the hospitals into five preference ordered classes based on three main criteria, namely: "outcome", "process" and "structure"

We evaluate the performance of our algorithms based on three performance measures; namely, number of misclassified alternatives, expected number of misclassified alternatives and number of alternatives placed by the DM. We test the performance of both algorithms for different misclassification thresholds. Misclassification thresholds measure the extent of erroneous classification the DM tolerates. If the misclassification threshold is 0, alternatives are placed into the classes exactly, in other words, no misclassification is allowed. When the misclassification threshold is small, we expect few misclassifications (both expected and realized), on the contrary, we expect to ask DM more alternatives to place into the classes. As the misclassification threshold increases, the information required from the DM decreases, on the other hand the number of misclassified alternatives increases. However, we expect that the number of misclassified alternatives do not exceed the misclassification threshold. Since the algorithms are probabilistic, we may result in unexpected results due to randomness.

In order to compare the results across different data sets we calculate percentage of alternatives that are placed by DM over all alternatives and percentage of misclassified alternatives over the algorithms that are placed by the DM.

In addition to placing alternatives to the classes, the number of possible classes that an alternative can be placed is tried to be decreased in order to ease decision making process for the DM. Even if we cannot place an alternative into a class we may present the DM a narrowed set of possible classes.

For all of the experiments we try to place the alternatives to the classes by utilizing both Algorithm 1 and Algorithm 2. The results, the observations following these results and the performance measures are represented in this section.

4.1.Financial Times Ranking MBA Programs Data Application

There are 81 MBA programs based on 2005 data of Financial Times. The alternatives are sorted into three classes. The rankings of MBA programs are based on 20 criteria. 12 of them are continuous and the remaining 8 are based on ranked data. In order to sort the alternatives, all criteria should be continuous. Köksalan et al. [9] develop a score estimation model and generate continuous data for the entire ranked criteria by preserving the ranking relations both in alternative and in criterion level. We have used this data set generated by Köksalan et al. [9]. There are three main criteria all of which are divided into three subintervals. The values of the parameters that we utilize for the underlying utility function of the decision maker are as follows: $u_k = \{0.65, 0.40\}$ k $= 1, 2$ w_{ip} values for i = 1, 2, 3 and p = 1, 2, and 3 is represented in the Table 4.1 below:

Table 4.1: w_{ip} values Corresponding Underlying Utility of the DM for Financial Times Data

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

The marginal utility function that corresponds to preference structure of DM on each criterion is represented in Figure 4.1.

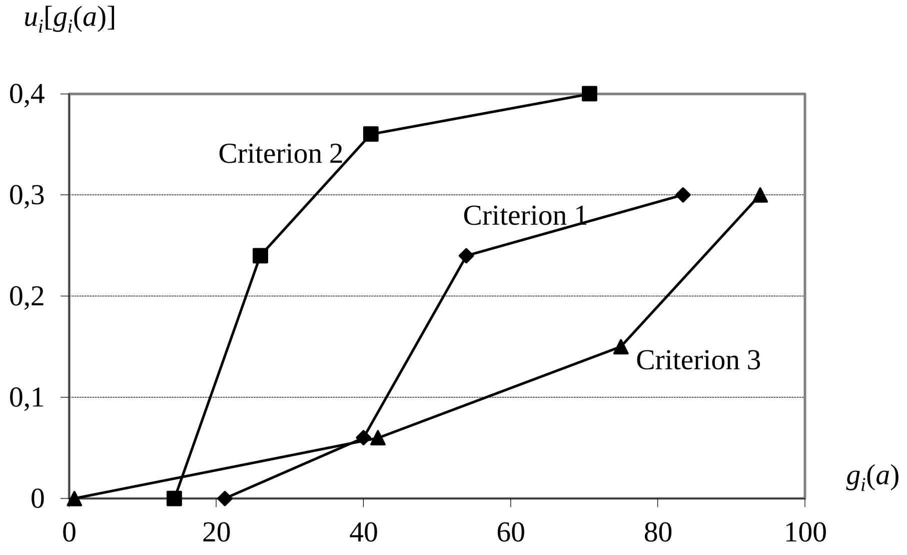

Figure 4.1: The Marginal Utilities on Each Criterion

The global utility of each alternative is calculated based on the marginal utilities as represented in Figure 4.1. Based on the underlying utilities, 15 of the alternatives are in C_1, 47 of the alternatives are in C_2 and the remaining 19 of the alternatives are in C_3.

We run our algorithms for eleven different misclassification thresholds changing from 0 to 0.5. When the misclassification threshold is zero alternatives are placed into the classes exactly with probability 1. Therefore, none of the alternatives are misclassified. As the threshold increases the number of misclassified alternatives is expected to increase as well. However, when the probability threshold is small, fewer number of alternatives are misclassified and DM is expected to place more alternatives into the classes. On the contrary, when the threshold is large DM allow more misclassification. Hence, few classification questions are asked to the DM and more number of misclassification is allowed. Since the process is random we can get unexpected results. In other words high misclassification thresholds may result in few questions asked and few alternatives misclassified and vice versa for the low misclassification thresholds.

For some of the alternatives even it cannot be placed into a class exactly the algorithms can reduce the possible number of classes an alternative can be placed. This process makes the classification easier for the DM since fewer possible classes are presented to the DM.

We used the data set for both of the algorithms and for three different distributions.

4.2. Random Data Generation

We have generated 500 random alternatives. We try to place alternatives into five preference ordered classes. For the underlying utility function of the DM we have utilized the parameter values that we have used for sorting MBA Programs data by Financial Times. Hence, the marginal utilities, the global utility function and as a result the preference structure of the DM are same as Financial Times MBA Programs Sorting Data. The values of class thresholds are as follows: $u_k = \{0.85, 0.70, 0{,}6, 0{,}4\}, \mathrm{k} = 1, 2{,}3{,}4$

With regard to underlying utility function of the DM, 22 of the alternatives are in class 1, 136 of the alternatives are in class 2, 155 of the alternatives are in class 3, 138 of the alternatives are in class 4 and the remaining 49 of the alternatives are in class 5.

We have analyzed our algorithm based on the expected number of misclassified alternatives, number of misclassified alternatives and number of questions asked to the DM in order to place the alternatives into the classes.

The algorithm is run for eleven misclassification thresholds changing from 0 to 0.5. Like Financial Time Data, we expect few misclassification and more questions when the misclassification threshold is small. On the contrary, we expect few questions and more misclassification as the misclassification increases.

With this experiment we want to show that our algorithm works well with more data and more number of classes.

4.3.U.S. News Report Data

We have utilized the data that is published by US News & World on Top Ranked Hospitals in cancer specialty. There is information about 900 hospitals and they rank 50 of the top scoring hospitals. In order for a hospital to be included in the analysis it is required that at least 249 inpatients are treated in 2010, 2011 and 2012. Each hospital in the list evaluated based on 12 criteria. In the end each hospital receives an overall score changing from 0 to 100. The criteria they use in order to rank the hospitals are based on three main dimensions of healthcare, namely; structure, process and outcomes.

Structural dimension includes criteria that are directly related with patient care such as hospital volume, intensity of nurse staffing, technology (e.g. nurse magnet technology) and some measurable features that characterize hospital environment.

Process dimension is based mainly on reputation of hospital for advancing and assisting highly qualified and technological care. The score on reputation relies on the average responses of most recent three surveys in 2011, 2012 and 2013 by board certified physicians whose field is cancer for the Best Hospital Rankings. It also shares partial weight for patient safety criteria with the outcome dimension.

Outcome dimension is represented mostly by survival, namely; risk adjusted mortality rates. In addition, as mentioned above it shares partial weight for patient safety with process dimension.

Table 4.2: Criteria of US News Report Best Hospitals in Cancer

Dimension	Criteria	Weight	Total Weight of Dimensions
Structure	Nurse Magnet Recognition	0.060	0.300
Structure	Cancer Patient Volume	0.120	
Structure	Nurse Staffing	0.120	
Process	Success in Keeping Patients Safe	0.025	0.350
Process	Reputation with Specialists	0.325	
Outcome	Survival	0.325	0.350
Outcome	Success in Keeping Patients Safe	0.025	

Some of the criteria are continuous and some of the criteria are ranked. In order to utilize the data set we use the score estimation model of Köksalan et al. [9] We try to sort 818 hospitals into five preference ordered classes. We have utilized the data set for cancer specialty 2013-2014 data set. In Table 4.2., the weights of all main dimensions and the weight of criteria that constitute these dimensions are represented.

There are three main criteria all of which are divided into three subintervals. The values of the parameters that we utilize for the underlying utility function of the decision maker are as follows: $u_k = \{0.40, 0.25, 0.20, 0.175\}$ k $= 1, 2,3$,4 w_{ip} values for i = 1, 2, 3 and p = 1, 2, and 3 is represented in Table 4.3.

Table 4.3: w_{ip} values Corresponding Underlying Utility of the DM for US News Report Data

p \ i	1	2	3
1	0.04	0.09	0.11
2	0.1	0.15	0.2
3	0.06	0.06	0.18

The marginal utility function that corresponds to preference structure of DM on each criterion is represented in Figure 4.2.

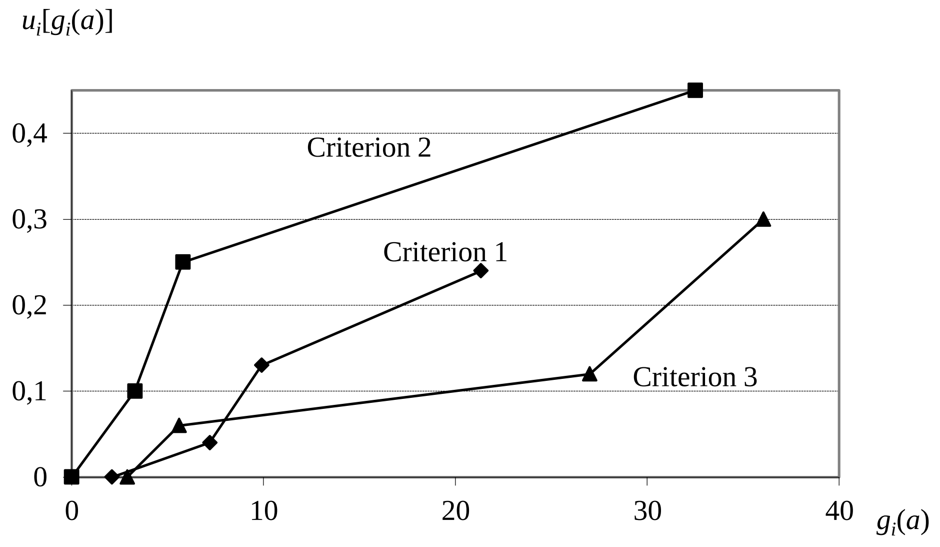

Figure 4.2: The Marginal Utilities on Each Criterion

The global utility of each alternative is calculated based on the marginal utilities as represented in Figure 4.2. Based on the underlying utilities, 39 of the alternatives are in class1, 303 of the alternatives are in class 2, 198 of the alternatives are in class 3, 62 of the alternatives are in class 4 and the remaining 209 of the alternatives are in class 5.

4.4. TÜBİTAK Entrepreneur and Innovator University Index 2014 Ranking

Since 2012, TÜBİTAK decides on 50 most entrepreneur and innovator universities. In 2014, 144 universities are evaluated which have at least 50 academicians. TÜBİTAK ranks only the first 50 and publishes the data set only for these universities. Ranking is based on 5 main criteria which consists of 23 sub criteria. The first main criterion is competence in scientific and technological research. It includes sub criteria such as number of publications, number of citation and number of PhD graduates. The second one is intellectual properties which are mainly based on number of different kind of patents. The third one is collaboration, cooperation and interaction that mainly constitutes of criteria related with university and industry cooperation. The fourth one is entrepreneurship and innovation culture and the last one is economical contribution and commercialization. In Table 4.4. , the weights of each criterion are provided.

Table 4.4: Criteria of TÜBİTAK Entrepreneur and Innovator University Index 2014 Ranking

Criteria	**Weight**
Scientific and technological research	0.20
Intellectual properties	0.15
Collaboration, cooperation and interaction	0.25
Entrepreneurship and innovation culture	0.15
Economical contribution and commercialization	0.25

There are five main criteria all of which are divided into three subintervals. The values of the parameters that we utilize for the underlying utility function of the decision maker are as follows: $u_k = \{0.65, 0.40\}$ k $= 1, 2$ w_{ip} values for i = 1, 2, 3, 4, 5 and p = 1, 2, and 3 is represented in Table 4.5.

Table 4.5: w_{ip} values Corresponding Underlying Utility of the DM for US News Report Data

p \ i	1	2	3
1	0.02	0.03	0.04
2	0.05	0.07	0.07
3	0.03	0.03	0.05
4	0.01	0.02	0.06
5	0.03	0.20	0.30

The marginal utility function that corresponds to preference structure of DM on each criterion is represented in Figure 4.3.

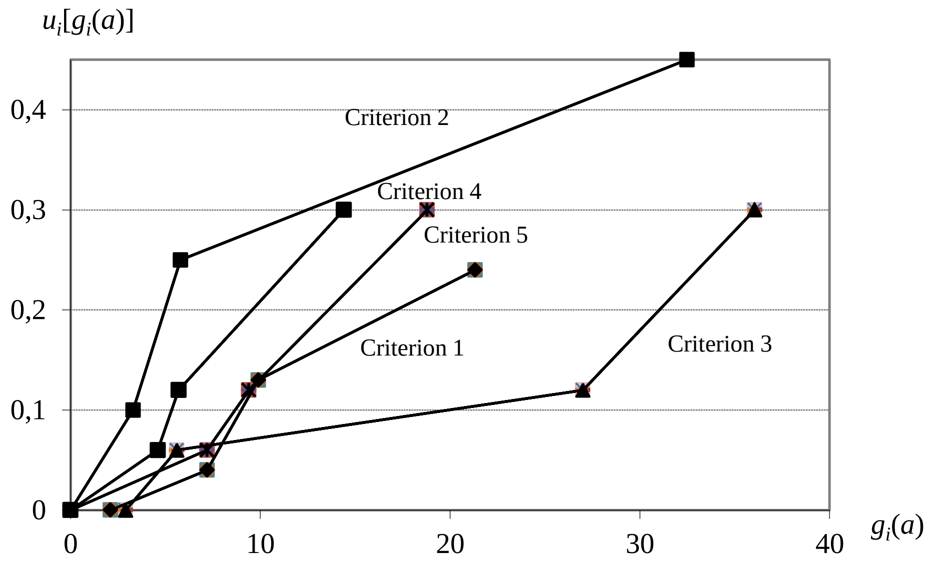

Figure 4.3: The Marginal Utilities on Each Criterion

4.5. Comparison of Approaches in .section 3.3.3.9. Probability Distributions

In order to sort the alternatives from best to the worst we have proposed two algorithms. In the first one, utility of alternatives are compared with class thresholds. If the utility of an alternative is greater than the class threshold, alternative is placed into that class. If none of the alternatives are placed in a run we ask the DM to place a selected alternative into a class

In the second algorithm on the other hand, alternatives are placed into the class if the utility of all alternatives are greater than any of the class thresholds. If this condition is not provided we ask DM to place a selected alternative into a class. Then, we try to place the remaining alternatives into a class all together again. This procedure continues till all the alternatives are placed into the classes.

Although it is hard to place the alternatives into the classes with the second algorithm, we expect fewer misclassifications. Since, we place the alternatives at the same time; the number of misclassified alternatives will decrease throughout the process. For instance, if an alternative is on the boundary of a class threshold but it can verifies the class threshold of one lower class it will be placed by the algorithm that places the alternatives one by one. On the other hand if all the alternatives are placed together the alternatives on the boundary have a higher chance to be placed into the correct classes.

We fit different distributions to difference between utilities and class thresholds for all experiments we have made in order to observe how different distributions affect our performance measures, namely expected number of misclassification, number of misclassification, and number of alternatives asked to the DM to be placed. We fit uniform, triangular and trapezoid distributions for each utility and class threshold difference for both of the algorithms.

In the following part, we will introduce the experiments we have made with the all probability distributions and all possible placement techniques we utilize for selecting the alternative to ask the DM as mentioned in Chapter 3. We present the results for each case from Table 4.6. to Table 4.29.

4.5.1. Financial Times Ranking MBA Programs Data Application

4.5.1.1. Algorithm-1: Placing alternatives one by one

Table 4.6: Results for Financial Times Ranking MBA Programs Data Application by Algorithm 1 with Uniform Distribution

UNIFORM DISTRIBUTION					
Threshold	**Questions asked**	**Number of Misclassification**	**Expected Number of Misclassification**	**Misclassified Alternatives (%)**	**Questions asked (%)**
t=0.00	38	0	0	0	46.91
t=0.05	37	0	0.75	0	45.67
t=0.10	32	0	2.65	0	39.50
t=0.15	28	4	4.39	7.55	34.57
t=0.20	23	7	7.51	12.07	28.40
t=0.25	20	8	9.99	13.11	24.69
t=0.30	17	9	12.74	14.06	20.99
t=0.35	15	17	16.12	25.76	18.52
t=0.40	12	20	19.56	28.99	14.81
t=0.45	10	29	23.62	40.85	12.35
t=0.50	19	19	25.35	30.65	23.46

Table 4.7: Results for Financial Times Ranking MBA Programs Data Application by Algorithm 1 with Triangular Distribution.

TRIANGULAR DISTRIBUTION					
Threshold	**Questions asked**	**Number of Misclassification**	**Expected Number of Misclassification**	**Misclassified Alternatives (%)**	**Questions asked (%)**
t=0.00	41	0	0	0	50.61
t=0.05	30	2	1,08	3.92	37.03
t=0.10	25	6	2,96	10.71	30.86
t=0.15	20	6	5,03	9.83	24.69
t=0.20	16	13	7,19	20.00	19.75
t=0.25	16	15	8,76	23.08	19.75
t=0.30	10	20	12,63	28.17	12.35
t=0.35	10	24	14,92	33.80	12.35
t=0.40	8	29	18,55	39.73	9.88
t=0.45	7	33	19,69	44.59	8.64
t=0.50	5	34	21,28	44.74	6.17

Table 4.8: Results for Financial Times Ranking MBA Programs Data Application by Algorithm 1 with Trapezoidal Distribution

TRAPEZOIDAL DISTRIBUTION					
Threshold	**Questions asked**	**Number of Misclassification**	**Expected Number of Misclassification**	**Misclassified Alternatives (%)**	**Questions asked (%)**
t=0.00	42	0	0	0	51.85
t=0.05	32	3	1.11	6.12	39.51
t=0.10	23	5	2.85	8.62	28.40
t=0.15	21	8	5,21	13.33	25.93
t=0.20	19	10	6.85	16.12	23.46
t=0.25	15	18	9.32	27.27	18.52
t=0.30	12	23	12.28	33.33	14.81
t=0.35	10	25	15.68	35.21	12.34
t=0.40	11	31	17.30	44.29	13.58
t=0.45	7	32	21.12	43.24	8.64
t=0.50	6	34	22.24	45.33	7.41

4.5.1.2. Algorithm-2: Placing all the alternatives together

Table 4.9: Results for Financial Times Ranking MBA Programs Data Application by Algorithm 2 with Uniform Distribution

UNIFORM DISTRIBUTION					
Threshold	**Questions asked**	**Number of Misclassification**	**Expected Number of Misclassification**	**Misclassified Alternatives (%)**	**Questions asked (%)**
t=0.00	38	0	0	0	46.91
t=0.05	37	0	0.05	0	45.67
t=0.10	36	0	0.12	0	44.44
t=0.15	36	0	0.12	0	44.44
t=0.20	36	0	0.12	0	44.44
t=0.25	33	0	0.90	0	40.74
t=0.30	29	1	1.95	1.92	35.80
t=0.35	29	1	2.83	1.92	35.80
t=0.40	29	1	2.83	1.92	35.80
t=0.45	16	10	10.61	15.38	19.75
t=0.50	19	8	7.33	12.90	23.45

Table 4.10: Results for Financial Times Ranking MBA Programs Data Application by Algorithm 2 with Triangular Distribution

TRIANGULAR DISTRIBUTION					
Threshold	**Questions asked**	**Number of Misclassification**	**Expected Number of Misclassification**	**Misclassified Alternatives (%)**	**Questions asked (%)**
t=0.00	41	0	0	0	50.62
t=0.05	38	0	0.08	0	46.91
t=0.10	36	0	0.26	0	44.44
t=0.15	33	0	0.61	0	40.74
t=0.20	27	4	1.32	7.41	33.33
t=0.25	26	4	1.40	7.27	33.00
t=0.30	23	5	2.39	8.62	28.40
t=0.35	25	4	1.69	7.14	30.86
t=0.40	25	4	1.69	7.14	30.86
t=0.45	20	8	4.65	13.11	24.69
t=0.50	10	11	11.07	15.49	12.35

Table 4.11: Results for Financial Times Ranking MBA Programs Data Application by Algorithm 2 with Trapezoidal Distribution

TRAPEZOIDAL DISTRIBUTION					
Threshold	**Questions asked**	**Number of Misclassification**	**Expected Number of Misclassification**	**Misclassified Alternatives (%)**	**Questions asked (%)**
t=0.00	42	0	0	0	51.85
t=0.05	39	0	0.03	0	48.15
t=0.10	37	0	0.17	0	45.68
t=0.15	32	0	0.78	0	39.51
t=0.20	32	0	0.78	0	39.51
t=0.25	27	3	1.58	5.56	33.33
t=0.30	25	4	1.87	7.14	30.86
t=0.35	24	4	2.20	7.02	29.63
t=0.40	15	9	8.68	13.64	18.52
t=0.45	14	10	9.13	14.93	17.28
t=0.50	12	10	11.06	14.49	14.81

4.5.2.Random Data Generation

4.5.2.1. Algorithm-1: Placing alternatives one by one

Table 4.12: Results for Random Data Generation by Algorithm 1 with Uniform Distribution

UNIFORM DISTRIBUTION					
Threshold	**Questions asked**	**Number of Misclassification**	**Expected Number of Misclassification**	**Misclassified Alternatives (%)**	**Questions asked (%)**
t=0.00	82	0	0.00	0	16.40
t=0.05	78	0	5.21	0	15.60
t=0.10	76	0	16.38	0	15.20
t=0.15	74	0	32.50	0	14.80
t=0.20	79	4	51.04	0.001	15.80
t=0.25	65	16	72.36	3.76	13.00
t=0.30	51	43	88.93	10.02	10.20
t=0.35	39	73	113.96	15.84	7.80
t=0.40	38	133	143.45	28.79	7.60
t=0.45	29	187	164.60	39.70	5.80
t=0.50	15	231	196.53	47.63	3.00

Table 4.13: Results for Random Data Generation by Algorithm 1 with Triangular Distribution

TRIANGULAR DISTRIBUTION					
Threshold	**Questions asked**	**Number of Misclassification**	**Expected Number of Misclassification**	**Misclassified Alternatives (%)**	**Questions asked (%)**
t=0.00	84	0	0,00	0	16.80
t=0.05	73	1	9,68	0	14.60
t=0.10	64	14	23,85	3.21	12.80
t=0.15	56	29	41,54	6.53	11.20
t=0.20	51	46	56,54	10.24	10.20
t=0.25	35	76	73,22	16.34	7.00
t=0.30	32	106	92,86	22.65	6.40
t=0.35	25	135	111,50	28.42	5.00
t=0.40	22	150	129,12	31.38	4.40
t=0.45	16	205	154,83	42.36	3.20
t=0.50	14	251	171,42	51.65	2.80

Table 4.14: Results for Random Data Generation by Algorithm 1 with Trapezoidal Distribution

TRAPEZOIDAL DISTRIBUTION					
Threshold	**Questions asked**	**Number of Misclassification**	**Expected Number of Misclassification**	**Misclassified Alternatives (%)**	**Questions asked (%)**
t=0.00	88	0	0.00	0	17.60
t=0.05	84	0	10.24	0	16.80
t=0.10	70	10	25.44	2.33	14.00
t=0.15	60	36	39.51	8.18	12.00
t=0.20	54	70	60.38	15.70	10.80
t=0.25	42	112	80.18	24.45	8.40
t=0.30	34	148	95.85	31.76	6.80
t=0.35	23	189	116.35	39.79	4.60
t=0.40	19	210	134.12	43.66	3.80
t=0.45	19	205	153.98	42.62	3.80
t=0.50	13	214	169.26	43.94	2.60

4.5.2.2. Algorithm-2: Placing all the alternatives together

Table 4.15: Results for Random Data Generation by Algorithm 2 with Uniform Distribution

UNIFORM DISTRIBUTION					
Threshold	**Questions asked**	**Number of Misclassification**	**Expected Number of Misclassification**	**Misclassified Alternatives (%)**	**Questions asked (%)**
t=0.00	82	0	0.00	0	16.40
t=0.05	79	0	0.09	0	15.80
t=0.10	77	0	0.26	0	15.40
t=0.15	76	0	2.94	0	15.20
t=0.20	72	0	1.45	0	14.40
t=0.25	72	0	1.20	0	14.40
t=0.30	72	0	1.20	0	14.40
t=0.35	66	2	5.10	0	13.20
t=0.40	59	7	19.39	1.59	11.80
t=0.45	55	10	21.91	2.25	11.00
t=0.50	33	36	82.07	7.71	6.60

Table 4.16: Results for Random Data Generation by Algorithm 2 with Triangular Distribution

TRIANGULAR DISTRIBUTION					
Threshold	**Questions asked**	**Number of Misclassification**	**Expected Number of Misclassification**	**Misclassified Alternatives (%)**	**Questions asked (%)**
t=0.00	84	0	0.00	0	16.80
t=0.05	77	0	0.10	0	15.40
t=0.10	74	0	0.43	0	14.80
t=0.15	72	0	0.09	0	14.40
t=0.20	71	0	0.74	0	14.20
t=0.25	70	1	1.28	0	14.00
t=0.30	59	9	2.30	2.04	11.80
t=0.35	49	17	13.18	3.77	9.80
t=0.40	49	17	21.12	3.77	9.80
t=0.45	44	29	30.17	6.36	8.80
t=0.50	50	11	15.14	2.44	10.00

Table 4.17: Results for Random Data Generation by Algorithm 2 with Trapezoidal Distribution

TRAPEZOIDAL DISTRIBUTION					
Threshold	**Questions asked**	**Number of Misclassification**	**Expected Number of Misclassification**	**Misclassified Alternatives (%)**	**Questions asked (%)**
t=0.00	88	0	0.00	0	17.60
t=0.05	84	0	0.05	0	16.80
t=0.10	82	0	0.22	0	16.40
t=0.15	82	0	0.22	0	16.40
t=0.20	79	0	0.96	0	15.80
t=0.25	75	1	2.34	0	15.00
t=0.30	73	1	2.86	0	14.60
t=0.35	61	11	17.77	2.51	12.20
t=0.40	57	10	17.47	2.26	11.40
t=0.45	57	11	18.21	2.48	11.40
t=0.50	62	10	10.35	2.28	12.40

4.5.3.U.S. News Best Hospital Data

4.5.3.1. Algorithm-1: Placing alternatives one by one

Table 4.18: Results for U.S. News Best Hospital Data by Algorithm 1 with Uniform Distribution

UNIFORM DISTRIBUTION					
Threshold	**Questions asked**	**Number of Misclassification**	**Expected Number of Misclassification**	**Misclassified Alternatives (%)**	**Questions asked (%)**
t=0.00	83	0	0.00	0	10.23
t=0.05	78	27	5.35	3.68	9.62
t=0.10	73	48	17.36	6.50	9.00
t=0.15	62	79	34.10	10.55	7.64
t=0.20	60	116	64.86	15.45	7.40
t=0.25	51	77	70.49	10.13	6.29
t=0.30	42	164	111.49	21.33	5.18
t=0.35	28	220	155.67	29.00	3.45
t=0.40	22	404	192.57	51.20	2.71
t=0.45	22	352	233.01	44.61	2.71
t=0.50	17	363	262.50	45.72	2.10

Table 4.19: Results for U.S. News Best Hospital Data by Algorithm 1 with Triangular Distribution

TRIANGULAR DISTRIBUTION					
Threshold	**Questions asked**	**Number of Misclassification**	**Expected Number of Misclassification**	**Misclassified Alternatives (%)**	**Questions asked (%)**
t=0.00	76	0	0.00	0	9.37
t=0.05	53	50	8.97	6.60	6.53
t=0.10	45	86	24.10	11.23	5.55
t=0.15	41	137	41.29	17.79	5.06
t=0.20	30	214	62.36	27.40	3.70
t=0.25	25	284	81.23	36.13	3.08
t=0.30	21	336	111.22	42.53	2.59
t=0.35	20	363	124.69	45.89	2.47
t=0.40	10	415	155.05	51.81	1.23
t=0.45	9	397	163.38	49.50	1.11
t=0.50	7	411	214.19	51.12	0.01

Table 4.20: Results for U.S. News Best Hospital Data by Algorithm 1 with Trapezoidal Distribution

TRAPEZOIDAL DISTRIBUTION					
Threshold	**Questions asked**	**Number of Misclassification**	**Expected Number of Misclassification**	**Misclassified Alternatives (%)**	**Questions asked (%)**
t=0.00	74	0	0.00	0	9.12
t=0.05	57	53	9.79	7.03	7.03
t=0.10	47	115	23.69	15.05	14.18
t=0.15	41	162	38.41	21.04	5.55
t=0.20	31	313	53.25	40.13	3.82
t=0.25	26	365	72.37	46.50	3.21
t=0.30	21	392	91.40	49.62	2.59
t=0.35	19	383	121.41	48.36	2.34
t=0.40	18	288	107.82	36.32	2.22
t=0.45	14	267	220.14	33.50	1.73
t=0.50	10	293	266.42	36.58	1.23

4.5.3.2. Algorithm-2: Placing all the alternatives together

Table 4.21: Results for U.S. News Best Hospital Data by Algorithm 2 with Uniform Distribution

UNIFORM DISTRIBUTION					
Threshold	**Questions asked**	**Number of Misclassification**	**Expected Number of Misclassification**	**Misclassified Alternatives (%)**	**Questions asked (%)**
t=0.00	83	0	0.00	0	10.23
t=0.05	83	0	0.00	0	10.23
t=0.10	81	0	0.20	0	9.99
t=0.15	80	0	0.34	0	9.86
t=0.20	81	0	0.20	0	9.99
t=0.25	72	5	2.79	0.68	8.88
t=0.30	71	6	3.18	0.81	8.75
t=0.35	68	12	9.43	1.62	8.38
t=0.40	70	7	3.48	9.45	8.63
t=0.45	50	56	37.14	7.36	6.17
t=0.50	53	34	26.54	4.49	6.53

Table 4.22: Results for U.S. News Best Hospital Data by Algorithm 2 with Triangular Distribution

TRIANGULAR DISTRIBUTION					
Threshold	**Questions asked**	**Number of Misclassification**	**Expected Number of Misclassification**	**Misclassified Alternatives (%)**	**Questions asked (%)**
t=0.00	76	0	0.00	0	9.37
t=0.05	75	0	0.00	0	9.25
t=0.10	67	5	0.87	0.67	8.26
t=0.15	65	6	1.85	0.80	8.01
t=0.20	64	7	2.05	0.94	7.89
t=0.25	61	8	3.61	1.07	9.86
t=0.30	57	13	6.05	1.72	7.03
t=0.35	61	9	3.50	1.20	7.52
t=0.40	43	53	18.23	6.90	5.30
t=0.45	42	48	17.71	6.24	5.18
t=0.50	47	31	13.89	4.06	5.80

Table 4.23: Results for U.S. News Best Hospital Data by Algorithm 2 with Trapezoidal Distribution

TRAPEZOIDAL DISTRIBUTION					
Threshold	**Questions asked**	**Number of Misclassification**	**Expected Number of Misclassification**	**Misclassified Alternatives (%)**	**Questions asked (%)**
t=0.00	74	0	0.00	0	9.12
t=0.05	71	0	0.04	0	8.75
t=0.10	72	0	0.01	0	8.88
t=0.15	72	0	0.01	0	8.88
t=0.20	58	8	3.39	1.06	7.15
t=0.25	63	6	1.54	0.80	7.77
t=0.30	63	6	1.54	0.80	7.77
t=0.35	56	8	4.47	1.06	6.91
t=0.40	50	18	10.68	2.37	6.17
t=0.45	46	30	15.25	3.70	3.70
t=0.50	33	49	42.03	6.04	4.32

4.5.4.TÜBİTAK Entrepreneur and Innovator Universities Ranking

4.5.4.1. Algorithm-1: Placing alternatives one by one

Table 4.24: Results for TÜBİTAK Entrepreneur and Innovator Universities Ranking Data by Algorithm 1 with Uniform Distribution

UNIFORM DISTRIBUTION					
Threshold	**Questions asked**	**Number of Misclassification**	**Expected Number of Misclassification**	**Misclassified Alternatives (%)**	**Questions asked (%)**
t=0.00	30	0	0.00	0	60
t=0.05	26	0	0.35	0	52
t=0.10	26	0	0.88	0	52
t=0.15	24	0	1.61	0	48
t=0.20	22	1	2.57	3.57	44
t=0.25	20	2	3.32	6.67	40
t=0.30	18	2	4.63	6.25	.36
t=0.35	15	2	6.72	5.71	30
t=0.40	14	6	8.17	16.67	.28
t=0.45	13	7	9.54	18.92	26
t=0.50	9	10	14.86	24.39	18

Table 4.25: Results for TÜBİTAK Entrepreneur and Innovator Universities Ranking Data by Algorithm 1 with Triangular Distribution

TRIANGULAR DISTRIBUTION					
Threshold	**Questions asked**	**Number of Misclassification**	**Expected Number of Misclassification**	**Misclassified Alternatives (%)**	**Questions asked (%)**
t=0.00	31	0	0.00	0	62
t=0.05	24	1	0.49	3.85	48
t=0.10	22	1	1.34	3.57	44
t=0.15	17	2	2.15	6.06	34
t=0.20	16	2	2.97	5.88	32
t=0.25	16	2	4.48	5.88	32
t=0.30	14	3	5.76	8.33	.28
t=0.35	9	4	7.74	9.75	.18
t=0.40	9	5	8.98	12.20	18
t=0.45	9	5	9.44	12.20	18
t=0.50	8	10	12.09	23.81	.16

Table 4.26: Results for TÜBİTAK Entrepreneur and Innovator Universities Ranking Data by Algorithm 1 with Trapezoidal Distribution

TRAPEZOIDAL DISTRIBUTION					
Threshold	**Questions asked**	**Number of Misclassification**	**Expected Number of Misclassification**	**Misclassified Alternatives (%)**	**Questions asked (%)**
t=0.00	32	0	0.00	0	64
t=0.05	23	0	0.38	0	46
t=0.10	19	1	1.52	3.22	38
t=0.15	18	1	2.40	3.13	36
t=0.20	15	2	3.39	5.71	30
t=0.25	14	3	4.44	8.33	28
t=0.30	13	3	5.69	8.11	26
t=0.35	10	4	7.89	10.00	20
t=0.40	9	5	9.83	12.00	18
t=0.45	8	5	10.73	11.90	16
t=0.50	7	10	13.14	23.26	14

4.5.4.2. Algorithm-2: Placing all the alternatives together

Table 4.27: Results for TÜBİTAK Entrepreneur and Innovator Universities Ranking Data by Algorithm 2 with Uniform Distribution

UNIFORM DISTRIBUTION					
Threshold	**Questions asked**	**Number of Misclassification**	**Expected Number of Misclassification**	**Misclassified Alternatives (%)**	**Questions asked (%)**
t=0.00	30	0	0.00	0	60
t=0.05	26	0	0.09	0	52
t=0.10	26	0	0.41	0	52
t=0.15	24	0	0.41	0	48
t=0.20	24	0	1.20	0	48
t=0.25	22	0	2.54	0	44
t=0.30	18	2	2.54	6.25	36
t=0.35	18	2	2.54	6.25	36
t=0.40	18	2	2.54	6.25	36
t=0.45	18	2	2.54	6.25	36
t=0.50	15	4	4.40	11.43	30

Table 4.28: Results for TÜBİTAK Entrepreneur and Innovator Universities Ranking Data by Algorithm 1 with Triangular Distribution

TRIANGULAR DISTRIBUTION					
Threshold	**Questions asked**	**Number of Misclassification**	**Expected Number of Misclassification**	**Misclassified Alternatives (%)**	**Questions asked (%)**
t=0.00	31	0	0.00	0	62
t=0.05	26	0	0.03	0	52
t=0.10	23	0	0.24	0	46
t=0.15	20	2	0.83	6.67	40
t=0.20	19	2	1.09	6.45	38
t=0.25	18	2	1.37	6.25	36
t=0.30	16	2	1.93	5.88	32
t=0.35	14	2	2.99	5.56	28
t=0.40	14	2	2.99	5.56	28
t=0.45	14	2	3.07	5.56	28
t=0.50	11	3	5.37	7.69	22

Table 4.29: Results for TÜBİTAK Entrepreneur and Innovator Universities Ranking Data by Algorithm 1 with Trapezoidal Distribution

TRAPEZOIDAL DISTRIBUTION					
Threshold	**Questions asked**	**Number of Misclassification**	**Expected Number of Misclassification**	**Misclassified Alternatives (%)**	**Questions asked (%)**
t=0.00	32	0	0.00	0	64
t=0.05	24	0	0.10	0	48
t=0.10	22	0	0.24	0	44
t=0.15	21	0	0.36	0	.42
t=0.20	21	0	0.36	0	42
t=0.25	16	2	2.30	8.33	.32
t=0.30	16	2	2.30	8.33	.32
t=0.35	14	2	2.94	7.69	28
t=0.40	14	2	2.94	7.69	28
t=0.45	13	2	4.35	7.41	.26
t=0.50	13	2	4.35	7.41	.26

4.6. Observations on the Results

4.6.1. Comparison of Two Algorithms

Both of the algorithms performed nearly the same when the misclassification threshold is zero for all of the experiments. As the misclassification threshold increases, number of questions asked to the DM for placement decreases, on the other hand the number of misclassified alternatives increases. (There can be some exceptions due to randomness) When we place the alternatives one by one, the number of misclassification increases more compared with the algorithm that places all the alternatives together. On the contrary, the number of questions asked to the DM will be less in Algorithm 1 compared to Algorithm 2.

In both of the algorithms and all the experiments, expected number of misclassification and number of misclassification is 0 when the misclassification threshold is zero. As the misclassification threshold increases, both expected number of misclassified alternatives and number of misclassified alternatives increase.

In order to assign alternatives into the classes we compare the utility of alternatives with class thresholds. The difference between the algorithms is as follows: In the first algorithm we assign the alterative whenever its utility value is greater than a class threshold, on the other hand, for the second algorithm we wait until all the alternatives verifies class threshold of at least one class. When the probability is calculated as 0, it implies that there could not be found any feasible solution that makes utility of that alternative greater than that class threshold. Hence, alternative cannot be placed into that class. On the other hand, if the probability threshold is 1, the utility of alternative is greater than the class threshold for certain and there is no feasible solution that makes the utility of this alternative worse than the class threshold it exceeds.

4.6.2. Effect of w_{ip} Values Generated from Different Probability Distributions

In all our experiments we need to simulate the DM by an underlying utility function and the utility values that separate the preference classes. The additive utility function of the DM is defined by w_{ip} values. The class borders are u_k values. w_{ip} represents the utility values of subintervals p on each criterion i. In order to form the underlying utility function of the DM we randomly generate a set of w_{ip} and u_k values. In order to eliminate the effect of w_{ip} values on the performance of the algorithm, we have randomly generated w_{ip} values by using uniform, normal and exponential distributions. We have tested the performance of both algorithms by constructing the underlying utility of the DM from these by results on Financial Times MBA Ranking Data. The results for each distribution and algorithm are represented from Table 4.30 to Table 4.32. As it can be observed from the tables none of the performance measures change dramatically for the different distribution of w_{ip} values. Therefore, we conclude that there is no bias to select random w_{ip} values in order to form the underlying utility function of DM.

Table 4.30: Results for Financial Times Ranking Data Application by Algorithm 2 with Trapezoidal Distribution and w_{ip} Values Generated by Uniform Distribution

UNIFORM w_{ip} VALUES					
Threshold	**Questions asked**	**Number of Misclassification**	**Expected Number of Misclassification**	**Misclassified Alternatives (%)**	**Questions asked (%)**
t=0.00	37	0	0.00	0	45.68
t=0.05	35	0	0.02	0	43.21
t=0.10	35	0	0.02	0	43.21
t=0.15	33	0	0.37	0	40.74
t=0.20	31	0	1.78	0	38.27
t=0.25	28	0	1.62	0	34.57
t=0.30	27	0	2.43	0	33.33
t=0.35	23	1	0.78	1.72	28.40
t=0.40	20	2	0.78	3.28	24.69
t=0.45	17	3	6.32	4.69	20.99
t=0.50	10	7	9.56	9.86	12.35

Table 4.31: Results for Financial Times Ranking Data Application by Algorithm 2 with Trapezoidal Distribution and w_{ip} Values Generated by Exponential Distribution

EXPONENTIAL w_{ip} VALUES					
Threshold	**Questions asked**	**Number of Misclassification**	**Expected Number of Misclassification**	**Misclassified Alternatives (%)**	**Questions asked (%)**
t=0.00	41	0	0.00	0	50.62
t=0.05	35	0	0.12	0	43.21
t=0.10	35	0	0.12	0	43.21
t=0.15	33	0	0.40	0	40.74
t=0.20	33	0	0.37	0	40.74
t=0.25	33	0	0.37	0	40.74
t=0.30	27	1	1.87	2.08	33.33
t=0.35	27	1	1.87	1.85	33.33
t=0.40	21	4	3.90	7.41	25.93
t=0.45	18	7	6.01	11.11	22.22
t=0.50	14	9	8.29	20.90	17.28

Table 4.32: Results for Financial Times Ranking Data Application by Algorithm 2 with Trapezoidal Distribution and w_{ip} Values Generated by Normal Distribution

NORMAL w_{ip} VALUES					
Threshold	**Questions asked**	**Number of Misclassification**	**Expected Number of Misclassification**	**Misclassified Alternatives (%)**	**Questions asked (%)**
t=0.00	34	0	0.00	0	41.98
t=0.05	27	0	0.02	0	33.33
t=0.10	27	0	0.02	0	33.33
t=0.15	24	0	0.37	0	29.63
t=0.20	21	0	1.78	0	25.93
t=0.25	20	1	1.62	1.64	24.69
t=0.30	19	1	2.43	1.61	23.46
t=0.35	22	2	2.47	3.39	27.16
t=0.40	22	2	2.47	3.39	27.16
t=0.45	13	4	6.32	19.12	16.05
t=0.50	10	9	9.56	12.68	12.35

4.6.3. The Relationship between $U_i - u_k$ Differences and Probabilities throughout the Algorithm

As already mentioned in the Algorithms in section 3, we try to fit a distribution to maximum and minimum $U[g(a_j)] - u_k$ differences and we try to assign alternatives into the classes according to the positivity and negativity condition of these differences.

As more and more alternatives are placed into the classes we expect that the minimum difference between $U[g(a_j)] - u_k$ increases and the maximum difference between $U[g(a_j)] - u_k$ decreases. Since the maximum value decreases and minimum value increases, the possible range of values narrow down. Hence, we can place all the alternatives. We have tested this lemma on Financial Times MBA Ranking Data with Algorithm 2. The results for the data set with trapezoidal distribution are represented in Figure 4.4. As it is seen in Figure 4.4, as the number of iterartions increases the minimum and maximum values approach to actual difference. Hence, it is guaranteed that all the alternatives are placed into the classes.

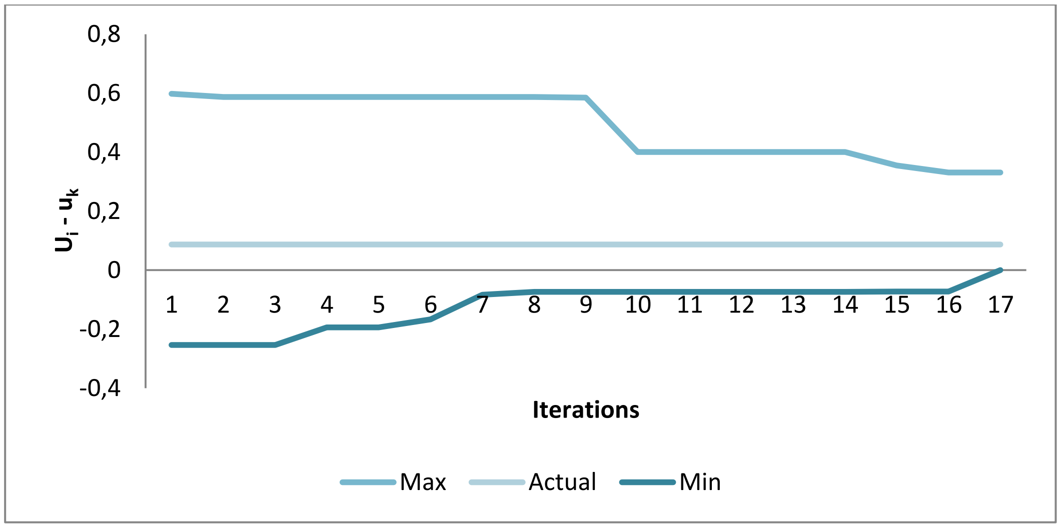

Figure 4.4: The Comparison of Actual Utility-Utility Threshold Difference and Min-Max Differences

As it can be seen from Figure 4.5., probability of being assigned to Class 1 approaches to 1 and probability to be placed in Class 2 approaches to 0. For this example, the actual class of alternative is class 1 based on the underlying utility function of DM. Hence throughout the iterations probability of being placed into the actual class increases whereas probability of being placed to other classes decreases.

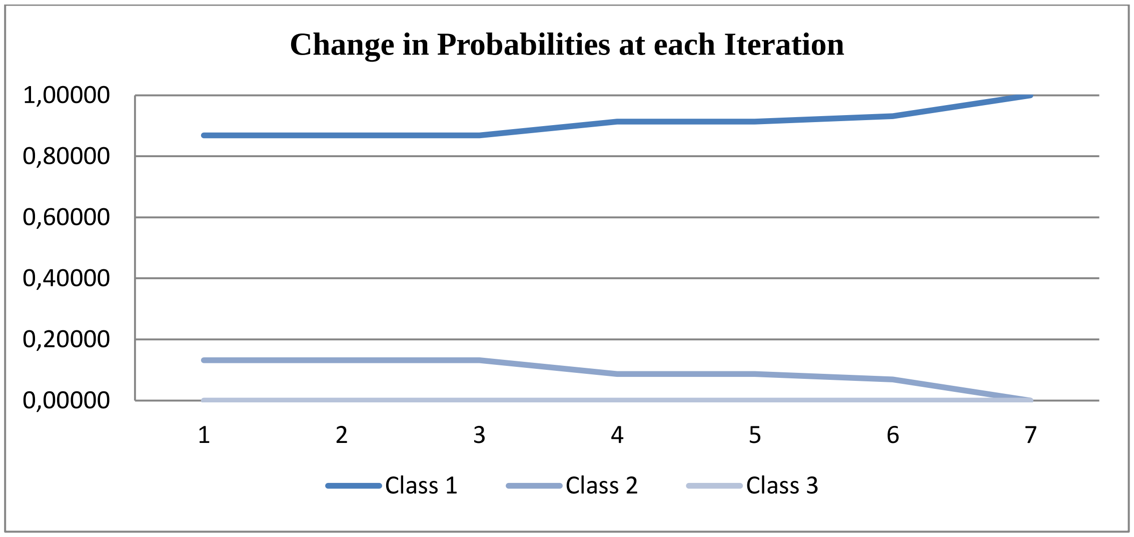

Figure 4.5: Change in Probabilities at each iteration

4.6.4. Mixed Integer Models

We normally do not limit the possible classes an alternative can be placed when we solve linear models. However as we have mentioned in Section 3.3.3.8, we can restrict the classes alternatives can be placed while solving the minimization and maximization models. By solving the models with the additional binary variables we force each alternative to be placed only one class. Hence, at each iteration the corresponding probability values are calculated for each utility and class threshold difference. Then, a class is selected for each alternative if it satisfies the conditions. We have tested MIP model by using Financial times MBA Ranking Data.

The integer model does not improve the results very much. Since solving integer models are harder than solving linear models we continue with linear models. However, the integer models can be enhanced to obtain better solutions as a further research area.

4.6.5. Number of Questions Placed by the DM at the Beginning of the Algorithm

We normally start the algorithm by asking the DM to place an alternative into each class. In order to start the algorithm we need to ask at least one alternative to the DM. If none of the alternatives are placed at the beginning, then we have no information about the preference structure of the DM. Hence, there will be no limit on minimum and maximum difference between utility and class threshold and the solution of these models are going to be unbounded. In order to enhance the solution process, we ask DM to place an alternative into each of the classes. However, it is possible to start just asking the class of one alternative. We have also solved the models by starting one placed alternative and utilizing Financial times MBA Ranking Data. If the algorithm starts with one alternative placed by the DM, then the number of misclassification is greater than the case where the DM place an alternative into each of the classes.

4.6.6. Comparison of Two Algorithms in terms of Misclassified Alternatives

We have compared both algorithms in terms of misclassified alternatives. We have already mentioned that we make less number of misclassifications with the second algorithm, since it places all alternatives at once. In addition, we expect that some of the alternatives will change classes while more and more alternatives can be placed to classes since the information the algorithm get increases. In order to observe these results we utilized the experiments represented in Part 4. We observed that when the second algorithm is used, alternatives that are placed in an incorrect class at the first iteration could be assigned to the correct class at the last iteration. Therefore, the number of misclassified alternatives is smaller when the second algorithm is used.

4.6.7. Number of Misclassifications in Different Classes

We expect that it is easier to place alternatives into the best and worst classes. In order to place the alternatives to the best or worst class we need to compare only one class threshold. On the other hand, while placing alternatives into the middle classes we need to compare utility of each alternative with two class thresholds. In addition, most of the alternatives fall into middle classes with respect to the underlying utility function of the DM. Some are close to upper bound of the class and some are close to lower bound. Hence, range of alternatives that are going to be placed into a class is larger.

4.6.8. Comparison of Number of Questions for Different Alternative Selection Methods to Ask the DM

We have mentioned three different alternative selection methods in order to select the most ambiguous alternative and to ask its class to the DM. In order to show how effective this method is we compared the number of alternatives asked to the DM for placement when we select the alternative the alternative to ask the class to the DM.

4.6.9. Comparison with Buğdacı et al. [18] for Financial Times Data Set.

The results for Buğdacı et al. data set is provided in Table 4.33. As we compare the results with Table 4.9, 4.10 and 4.11, it can be observed that the number of misclassified alternatives are fewer in our method as expected. Due to the structure of the algorithm, number of questions increases, but it is still a reasonable number compared to decrease in the number of misclassified alternatives.

Table 4.33: Results for Buğdacı et al. [18] for Financial Times Ranking MBA Programs Data Application with Normal Distribution

NORMAL DISTRIBUTION				
Threshold	**Questions asked**	**Number of Misclassification**	**Misclassified Alternatives (%)**	**Questions asked (%)**
t=0.00	37	0	0	45.68
t=0.05	36	0	0	44.44
t=0.10	33	0	0	40.74
t=0.15	32	0	0	40.74
t=0.20	31	0	0	38.27
t=0.25	26	1	1.82	34.57
t=0.30	20	4	0	33.33
t=0.35	15	11	16.67	28.40
t=0.40	11	16	22.86	24.69
t=0.45	8	22	30.14	20.99
t=0.50	3	39	0.50	12.35

4.6.10. Comparison of Performance Measures for Different Experiments.

As we compare the performance measures for from Table 4.6 to 4.29, we observe that the higher the number of alternatives, the easier it is to place the alternatives into the classes. The percentage of information required from the DM is the least when there are many alternatives. For larger data sets, more information is collected about the preferences of the DM. For the experiments that have similar number of alternatives, the percentage of misclassified alternative and percentage of information obtained from the DM are similar.

CHAPTER 5

CONCLUSION

We developed two interactive methods for multi-criteria sorting problems. The methods try to place alternatives into the preference ordered classes probabilistically. Alternatives are placed into the classes by comparing the probabilities with misclassification thresholds which represent the maximum acceptable error level. The assignment is realized if the probability of making an incorrect assignment is acceptably small. If misclassification threshold is given as zero, alternatives are placed into the classes exactly.

We assumed that the unknown underlying utility function of the DM is additive. In addition, marginal utility of each alternative on each criterion is piecewise linear. We try to place alternatives into the classes based on different probability distributions. We generate unbiased estimators by using trapezoidal, uniform and triangular distributions.

In order to evaluate the algorithms different data sets are utilized. The methods are firstly implemented on a data set that is published by Financial Times in order to sort the best MBA Programs (2009). This data set is utilized for sorting purposes by Köksalan and Özpeynirci (2009). Each alternative is composed of three criteria and they are tries to be placed into three preference algorithm. To test the performance of alternatives with different number of classes, 500 random data are generated and they are tried to be placed into five preference ordered classes. In addition, the data set that is published by US News Report that ranks Best Hospital in Cancer is arranged for sorting purposes. Hospitals are tried to be placed into five preference ordered classes. Finally, the data set that is published by TÜBİTAK in order to rank the 50 innovative and entrepreneur universities in Turkey. Each alternative consist of five criteria and tried to be placed into three preference ordered classes.

We tested the performance of the methods by number of alternatives that are misclassified, expected number of misclassification, and amount of information obtained from the DM on different data sets.

In the first algorithm, linear models that maximize and minimize the differences between utility and class thresholds are solved. Then, we fit a probability distribution between the minimum and maximum difference. If the calculated misclassification probability is acceptably small alternative is placed into the class. If none of the alternatives can be placed DM is asked to place an alternative. The algorithm terminates when all the alternatives are placed.

In the second algorithm, there is no difference in terms of probability calculation. However, if alternatives cannot be placed into the classes at the same time, none of the alternatives are assigned into the classes.

We suggest asking the DM the class of alternative about which we have least information. In order to do that, we ask the alternative whose probability of being placed into the successive classes is closest 0.5.

We suggest that in the second algorithm both the realized and expected number of misclassification is smaller. Since alternatives are placed after more number of iterations, their probability will approach the actual value.

An interesting research could be trying to decrease misclassification by defining misclassification degree of alternatives. Misclassification degree is defined by the measure for how many classes further or before an alternative is placed rather than the class it belongs to. It is a worse case for example for an alternative whose actual class is C_1 is placed into C_5 compared with being place to C_2. Therefore, the algorithm can be re-evaluated by such a performance measure.

REFERENCES

[1] E. Jacquet-Lagrèze, and Y. Siskos. Assessing a set of additive utility functions for multicriteria decision making: The UTA method, European Journal of Operational Research, 10 (2), 151–164, 1982.

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

[3] M. Koksalan, V. Mousseau, Ö. Özpeynirci, and S. Özpeynirci. A New Outranking-Based Approach for Assigning Alternatives to Ordered Classes. Naval Research Logistics, Vol. 56 (1), 74-85, 2009.

[4] K. Kosmido et al. Country Risk Evaluation: Methods and Applications, DOI: 10.1007/978-0-387-76680-5 2, Springer Science+ Business Media, LLC, 19-33,2008.

[5] M. Köksalan, and C. Ulu. An Interactive Procedure for Placing Alternatives in Preference Classes, uropean Journal of Operational Research, Vol 144, 429-439, 2003.

[6] C. Ulu, C. and M. Köksalan. An Interactive Procedure for Selecting Acceptable Alternatives in the Presence of Multiple Criteria, Naval Research Logistics, Vol. 48, 592-606, 2001.

[7] P. Korhonen, J. Wallenius, S. Zionts. Solving the discrete multicriteria problem using convex cones Management Science, 10, 1336-1345, 1984.

[8] M. Köksalan, and S. B. Özpeynirci. An Interactive Sorting Method for Additive Utility Functions, Computers and Operations Research, 36, 2565-2572, 2009.

[9] Köksalan, M., Büyükbaşaran, T., Özpeynirci, Ö., & Wallenius, J. (2010). A flexible approach to ranking with an application to MBA programs. European Journal of Operational Research, 201(2), 470-476.

[10] Doumpos, M. and Zopounidis, C. (2004b) Developing sorting models usingpreference disaggregation analysis: An experimental investigation. EuropeanJournal of Operational Research, Vol. 154, Issue 3, 585-598.

[11] B. Soylu. A Multi-Criteria Sorting Procedure with Tchebycheff Utility Function, Computers and Operations Research, Vol.38, pp.1091-1102, 2011.

[12] Ishizaka, Alessio, Pearman, C. and P. Nemery. AHPSort: an AHP-based method for sorting problems. International Journal of Production Research, 50 (17). pp. 4767-4784, 2012.

[13] C. Zopounidis, and M. Doumpos. Developing a multicriteria decision support system for financial classification problems: The FINCLAS system. Optimization Methods and Software, 8(3-4):277-304, 1998.

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

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

[16] Dehnokhalaji, A., Korhonen, P. J., Köksalan, M., Nasrabadi, N., & Wallenius, J. (2011). Convex cone-based partial order for multiple criteria alternatives. *Decision Support Systems*, *51*(2), 256-261.

[17] Brans, J. P., Vincke, P., & Mareschal, B. (1986). How to select and how to rank projects: The PROMETHEE method. European journal of operational research, 24(2), 228-238.

[18] Buğdaci, A. G., Köksalan, M., Özpeynirci, S., & Serin, Y. (2013). An interactive probabilistic approach to multi-criteria sorting. *IIE Transactions*,*45*(10), 1048-1058.

[19] R. Lahdelma, J. Hokkanen, and P. Salminen. SMAA - Stochastic multiobjective acceptability analysis. European Journal of Operational Research, 106(1):137-143, 1998.

[20] Tervonen, T., Figueira, J. R., Lahdelma, R., Dias, J. A., & Salminen, P. (2009). A stochastic method for robustness analysis in sorting problems. European Journal of Operational Research, 192(1), 236-242.

[21] M. Kadzinski, T. Tervonen. Stochastic ordinal regression for multiple criteria sorting problems. Decision Support Systems (55), 55-66, 2013.

[22] Prasad, R. V., Aneja, Y. P., & Nair, K. P. K. (1990). Optimization of bicriterion quasi-concave function subject to linear constraints.

[23] Kadziński, M., & Słowiński, R. (2013). DIS-CARD: a new method of multiple criteria sorting to classes with desired cardinality. Journal of Global Optimization, 56(3), 1143-1166.

[24] Greco, S., Matarazzo, B., & Słowiński, R. (2011). Dominance-based rough set approach on pairwise comparison tables to decision involving multiple decision makers. In Rough Sets and Knowledge Technology (pp. 126-135). Springer Berlin Heidelberg.

[25] Dias, L. C., & Mousseau, V. (2003). IRIS: A DSS for multiple criteria sorting problems. Journal of Multi-Criteria Decision Analysis, 12(4-5), 285-298.

[26] Larichev, O. I., & Moshkovich, H. M. (1994). An approach to ordinal classification problems. International Transactions in Operational Research, 1(3), 375-385.

[27] Zopounidis, C., & Doumpos, M. (2000). PREFDIS: A multicriteria decision support system for sorting decision problems. Computers & Operations Research, 27(7), 779-797.