FROM QUALITATIVE TO QUANTITATIVE EVALUATION METHODS IN MULTI-CRITERIA DECISION MODELS

Mag. Biljana Mileva Boshkoska

Doctoral Dissertation
Jožef Stefan International Postgraduate School
Ljubljana, Slovenia, November 2013

Evaluation Board:
Prof. Dr. Sašo Džeroski, Chairman, Jožef Stefan Institute, Jamova 39, Ljubljana, Slovenia
Asst. Prof. Dr. Martin Žnidaršič, Member, Jožef Stefan Institute, Jamova 39, Ljubljana, Slovenia
Prof. Dr. Vladislav Rajkovič, Member, University of Maribor, Kidričeva cesta 55a, Kranj, Slovenia

MEDNARODNA PODIPLOMSKA ŠOLA JOŽEFA STEFANA
JOŽEF STEFAN INTERNATIONAL POSTGRADUATE SCHOOL
Ljubljana, Slovenia

Mag. Biljana Mileva Boshkoska

FROM QUALITATIVE TO QUANTITATIVE EVALUATION METHODS IN MULTI-CRITERIA DECISION MODELS

Doctoral Dissertation

OD KVALITATIVNIH DO KVANTITATIVNIH METOD VREDNOTENJA V VEČPARAMETRSKIH ODLOČITVENIH MODELIH

Doktorska disertacija

Supervisor: Prof. Dr. Marko Bohanec

Ljubljana, Slovenia, November 2013

Index

Abstract

The thesis addresses the decision making problem of ranking a finite set of qualitative options that are sorted into a set of classes.

The problem is directly motivated by DEX methodology, where options that belong to the same class are indistinguishable. A starting method for solving the problem is the linear-based QQ method. QQ is based on assumptions that options are monotone or nearly linear, hence it does not work as desired for non-linear non-monotone options. To solve this issue, we propose and evaluate four different QQ-based methods for estimating a regression function: impurity functions for weights estimation in the linear regression function; polynomial functions for regression; linear programming for search of the optimal parameters of the regression function; and copula functions for aggregation and regression.

The main focus is on the last method which proposes a replacement of the linear functions in QQ with copula-based functions. This approach leads to fully and partially nested Archimedean constructions (FNACs and PNACs). Three families of Archimedean copulas are considered: Frank, Clayton and Gumbel. Regression functions are derived for the FNACs and PNACs in order to obtain the option ranking with the method. Apart from modeling the non-linearities in the data, the copula-based approach allows to define different dependences among the considered attributes, and based on the different FNACs and PNACs it provides different possible rankings for a given problem. To find the best ranking function, a measure which maximizes the distances among the options in a given class is proposed.

Extensive numerical experiments were performed to evaluate the performance and applicability of the four proposed methods and to give insights into their applicability in practice. The experiments confirmed the usefulness of the proposed copula-based method for ranking non-linear decision tables. Finally, the copula-based methods were successfully applied to two real-world cases: ranking of EC motors and ranking of workflows.

Povzetek

Disertacija obravnava problem razvrščanja (rangiranja) končne množice kvalitativnih alternativ, ki so razvrščene v posamezne razrede.

Problem razvrščanja je neposredno spodbudila DEX metodologija, kjer so alternative, ki pripadajo istemu razredu, med seboj nerazpoznavne. Postopek za rešitev problema se prične z linearno kvalitativno-kvantitativno (QQ) metodo. QQ temelji na predpostavkah, da so alternative monotone ali približno linearne, zato v primeru nelinearnih in/ali nemonotonih alternativ ne daje željenih rezultatov. Za rešitev težave disertacija predlaga in evaluira štiri različice QQ metode za oceno regresijske funkcije: nečiste (impurity) funkcije za ocenjevanje uteži v linearnih regresijskih funkcijah, polinomske funkcije za regresijo, linearno programiranje za iskanje optimalnih parametrov regresijske funkcije, in kopule (copula functions) za agregacijo in regresijo.

Glavni poudarek je na zadnje omenjeni metodi, ki predlaga zamenjavo linearnih funkcij v QQ metodi kopulami. Uporaba kopul vodi k popolnim in delno vgrajenim Arhimedovim konstrukcijam (FNACs in PNACs). Obravnavane so tri družine Arhimedovih kopul: Frank, Clayton in Gumbel. Za razvrstitev alternativ so razvite regresijske funkcije z uporabo FNACs in PNACs. Uporaba kopul poleg modeliranja nelinearnosti v podatkih omogoča opredelitev različnih odvisnosti med atributi in na podlagi različnih FNACs in PNACs zagotavlja različne možne razvrstitve določenega problema. Pri iskanju najboljše funkcije za razvrstitev predlagamo mero, ki povečuje razdalje med alternativami v danem razredu.

Za oceno učinkovitosti in uporabnosti štirih predlaganih metod so bili izvedeni obsežni numerični eksperimenti. Le-ti so za razvrstitev nelinearnih tabel odločanja potrdili uporabnost predlagane metode, ki temelji na kopulah. Nenazadnje se metoda kopul uspešno uporablja v dveh resničnih primerih: za razvrščanje EC motorjev in pri razvrščanju delovnih tokov (workflow).

List of Abbreviations

AUF	=	additive utility function
CBR	=	Case-Based Reasoning
CBRR	=	Case-Based Rule Reasoning
CDF	=	cumulative distribution function
CIPER	=	Constrained Induction of Polynomial Equations for Regression
DEX	=	decision expert
DM	=	Decision maker
DSS	=	decision support system
DT	=	Decision table
DW	=	data-mining workflows
EC	=	electronically commutated
FNAC	=	fully nested Archimedean construction
gB	=	Gini Breiman
gC	=	Gini covariance
gP	=	Gini population
idf	=	inverse distribution function
IG	=	Information gain
LP	=	linear programming
MACBETH	=	Measuring Attractiveness by a Categorical Based Evaluation Technique

MAUT	=	Multiple Attribute Utility Theory
MCDA	=	multi-criteria decision analysis
MCDM	=	multi-criteria decision making
PCT	=	pairwise comparison table
PDA	=	preference disaggregation principal
PDF	=	probability density function
PNAC	=	partially nested Archimedean construction
QDT	=	qualitative decision table
QQ	=	qualitative-quantitative method
RBR	=	Rule-Based Reasoning
ROR	=	robust ordinal regression
ZAPROS	=	an abbreviation of Russian words for Closed Procedures near Reference Situations

1 Introduction

Multi-criteria decision analysis (MCDA) is a sub-discipline of operations research, concerned with structuring and solving decision problems that involve multiple criteria (Zopounidis and Pardalos, 2010). For a given set of decision options, MCDA considers three types of problematics: choosing, sorting and ranking (Roy, 2005). Choosing means a selection of one option (or a sub-set of options) from the set of decision options as the best ones. Sorting aims at assigning a class to each of the available options from a set of predefined classes. Ranking aims at defining a complete or partial order on given set of options. This thesis addresses the problematic of ranking.

The starting point for the research are decision problems where options are represented with qualitative attributes that form a decision table. The decision maker's preferences split the decision table into subsets of equally preferred options, called classes, so that options belonging to the same class are considered indistinguishable. In other words, the sorting problem is solved this way. In practice, this is often inadequate and hence one wants to further distinguish between options belonging to the same class. This means that, in addition to sorting the options into discrete classes, one also wants to rank them within classes. Furthermore, the wish is to obtain such rankings with least effort, i.e., using only the information already available in the decision table.

This dissertation presents a modeling approach that combines qualitative and quantitative models. In particular, it addresses the following problem: Given some qualitative multi-criteria model, is it possible to construct a corresponding quantitative multi-criteria model for the evaluation, and consequently for ranking, of options? The resulting model should be in some way consistent with the original one and should be preferably constructed in an automatic or semi-automatic way from the information contained in the qualitative model. These are very important questions, both theoretically and practically. Theoretically, it is important for bridging the gap between both types of models and involves a number of theoretically interesting sub-problems, such as finding a suitable representation of a decision problem in different forms for different computational process, within the same decision-making process (Doyle-and and Thomason, 1999). Practically, bridging this gap is important to overcome some limitations of qualitative models, such as low sensitivity and limited applicability for the ranking of options.

Therefore, the goal of this thesis is to develop and study quantitative methods that rank the options belonging to the same class solely by using the information contained in the qualitative decision table. The obtained ranking must have three main properties. Firstly, it has to distinguish among all options, if it is possible, or allow equal ranking in cases when is desired (for example, symmetric options in symmetric decision tables). Secondly, the ranking should be monotone. Finally, it has to provide consistency with the qualitative model, so that the evaluation of each option must belong to the interval $[c \pm 0.5]$, where $c \in C$ is the quantitative value of the class. The last property additionally provides direct information to the decision maker about the class c in which the evaluated option belongs to, hence it ensures readability of the rankings.

The problem addressed here is directly motivated by decision expert (DEX) methodology (Bohanec and Rajkovič, 1990; Bohanec et al., 2012). DEX is a qualitative modeling methodology that, in the process of developing a decision model, produces decision tables which can be interpreted either as a set of options or a set of decision rules governing the preference evaluation. To solve the ranking problem, the qualitative-quantitative method (QQ) (Bohanec, 2006; Bohanec et al., 1992) has been developed. QQ

is based on assumptions that options are monotone or nearly linear, hence it does not work as desired for non-linear non-monotone options. There are other qualitatiative MCDM methods that also deal with this issue, as described in section 3.1. However, none of them solves the problem stated above.

1.1 Aims and Hypothesis

The aim of the dissertation is to *develop, implement and evaluate a method for monotonic, consistent and full ranking of a set of qualitative multi-attribute decision options derived from DEX methodology*.

The overall aim is addressed through the following main objectives:

Objective 1 Definition of theoretical framework for automatic ranking of qualitative options derived with DEX methodology.

Objective 2 Evaluation, modification and definition of shortcomings of three proposed state-of-the-art research directions for ranking of qualitative options: usage of impurity functions for weights estimation in the linear regression function; usage of polynomial functions for regression; and usage of optimization for regression.

Objective 3 Development and implementation of methodology based on copula functions for option ranking.

Objective 4 Demonstration and evaluation of the benefits and generality of the copula-based methodology.

The thesis addresses the following hypotheses that are developed and experimentally tested:

Hypothesis 1 The integration of qualitative decision problems and statistical copula-based functions enables full option ranking based solely on the information provided in decision tables derived from DEX methodology.

Hypothesis 2 Copula-based regression equations improve the number of solvable qualitative decision tables compared to the state-of-the-art methods.

1.2 Research Methodology and Specific Contributions

The thesis starts with the existing QQ method, designed for ranking of monotone qualitative options. The modifications of QQ lead to four research directions:

Research direction 1 Usage of impurity functions for weights estimation in the linear regression function in QQ.

Research direction 2 Usage of polynomial functions for regression.

Research direction 3 Usage of optimization technique for providing a regression function.

Research direction 4 Usage of copula-based functions for performing the regression task.

In each of the four research directions, the regression function is used for ordering the options and hence for ranking.

The first research direction includes investigation of different impurity functions for estimation of coefficients in the linear regression equation used by QQ. The main contribution arising from this research

direction is the usage of different non-linear functions instead of the standard least squares algorithm, that lead to full rankings of many non-monotone decision tables, for which QQ provides equal rankings (ties) of options or fails to fulfill the monotonicity of the rankings. The main disadvantage of these functions is their limitation to provide full option ranking when option attributes have different probability distributions but receive equal weights in the linear regression equation in QQ.

The second research direction introduces polynomial functions instead of the linear one in QQ. For that purpose the methods Constrained Induction of Polynomial Equations for Regression (CIPER) and New CIPER are employed for heuristic search of the best polynomial for a given decision table. Results show that polynomial functions outperform QQ. They are suitable for cases when the requited solution should be monotone, but usually fail to provide full ranking of options.

The third research direction redefines the option ranking problem as constraint optimization problem, and as such, investigates the usage of linear programming for defining its solution. This intuitive approach mainly leads to overly stringent constraints that rarely form a feasible region for solutions. The main contribution here is the investigation of optimization technique for option ranking and answering the question why the optimization could not be used in most cases of the examined decision tables.

The fourth research approach, which is the main focus of this thesis, changes the view of the decision tables from deterministic to stochastic. In this approach, the attributes are considered as random variables. Copulas are functions which connect marginal distributions of random variables and their joint distribution. The copula function is highly sensitive to small variations of input variables, thus providing distinct results for cases where linear regression used in QQ fails. The combination of the sensitivity of copulas and their monotonicity property leads to correct option rankings unlike QQ, which may lead to inverse option rankings. This thesis uses one-parametric multivariate copulas for evaluation of symmetric decision tables, and bi-variate copulas which are extended to multi-variate ones, for non-symmetric decision tables. To form a multi-variate copula, the bi-variate ones are merged forming a hierarchical copula construction. In the thesis, two types of hierarchical copulas are examined: the fully nested Archimedean construction (FNAC) and partially nested Archimedean construction (PNAC). For the obtained hierarchical copula constructions, the thesis presents new quantile regression equations for different position of the dependent variable in the FNAC and PNAC. FNAC and PNAC are built from bi-variate Clayton, Frank and Gumbel copulas, which belong to the family of Archimedean copulas.

Another contribution of this thesis is the evaluation of the proposed methods in the four research directions. For evaluation, different decision tables (artificial and real) are used to demonstrate the generality and applicability of the methods. Firstly, evaluation on three artificially constructed sets of decision tables with different number of attributes and different cardinality is performed. Secondly, the applicability of the copula-based method on two real-case examples is demonstrated: ranking of EC motors, and ranking of data mining workflows.

The proposed methods lead to new decision support methods for qualitative option evaluation and ranking within classes, and are competitive with current state-of-the-art methods. The new methods lead to several contributions. Firstly, the used approaches extend the space of solvable monotone and linear decision tables to the space of general discrete decision tables. Secondly, methods bridge the gap between qualitative and quantitative models in terms of improving qualitative methods' low sensitivity and limited applicability for the ranking of options within classes. Finally, the methods are applicable for ranking of qualitative options specified with non-linear and or non-monotone decision tables.

1.3 Organization of the Thesis

The thesis is based on two main theoretical fields: qualitative decision making methods, and regression methods for option ranking. Due to the diversity of the methods in each of the two research fields,

Chapters 3, 5 and 6 in the thesis contain the background and the references to the related state-of-the-art literature. For better presentation of the discussed topics, several running examples are presented throughout the dissertation, which are chosen to provide best description of the used techniques and methods.

The organization of the thesis is shown in Figure 1.1. The center of the Figure 1.1 is the problem that is considered in the thesis denoted as 'Ranking of options'. The chapters that are presented as circles are enumerated in clockwise direction, starting with Chapter 2, and ending with Chapter 10.

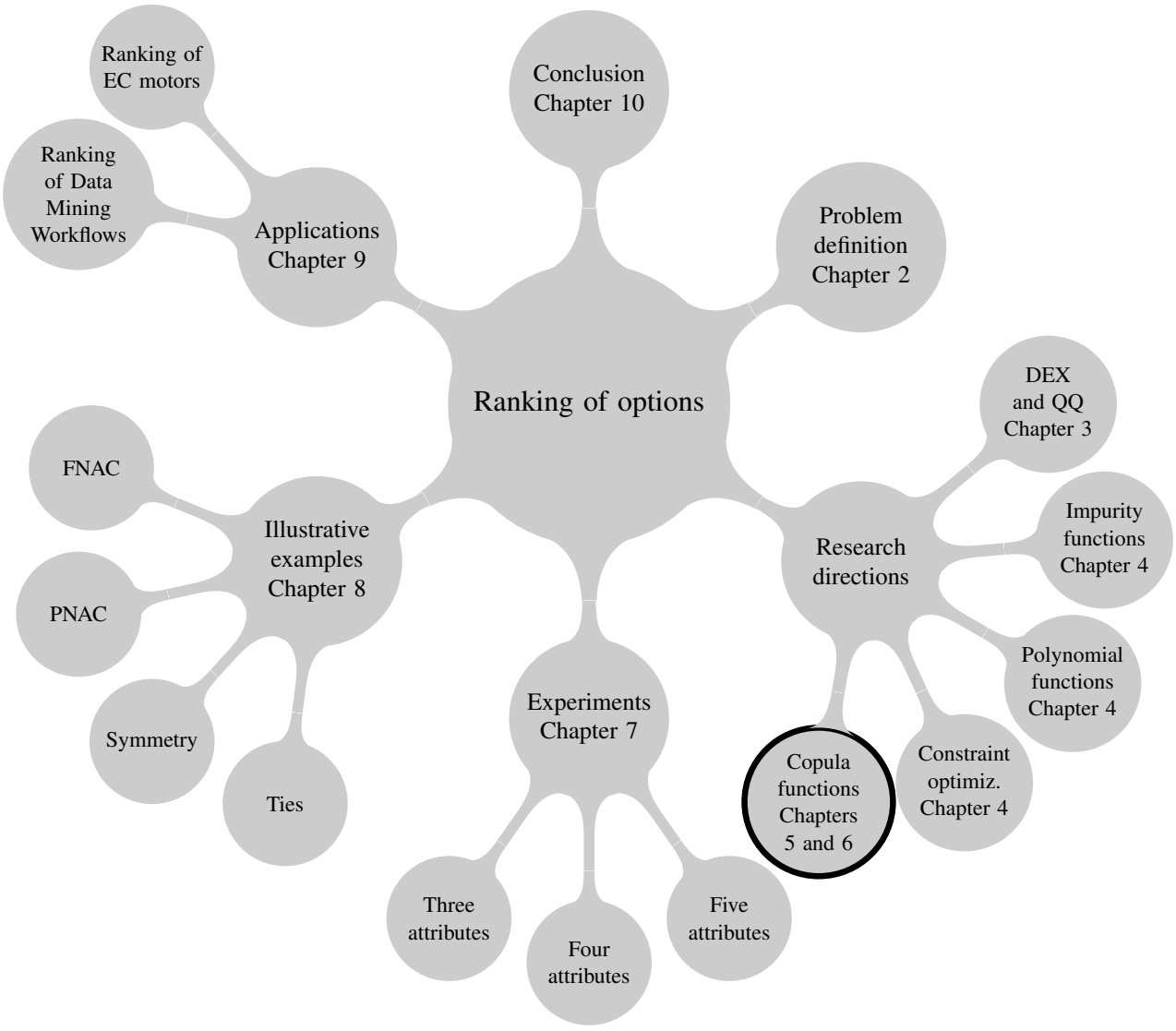


Figure 1.1: Organization of the thesis

The thesis starts with the mathematical formulation of the problem that is given in Chapter 2. The proposed solutions in the thesis are based on the QQ method. The structure of QQ is described in Chapter 3. Chapter 4 describes three modifications of QQ: usage of different non-linear estimators for weights calculation in the linear regression function; usage of polynomial functions instead of the linear one; and redefinition of the problem in terms of constraint optimization that is solved using linear programming. The main accent in the thesis is given on the fourth modification of QQ that applies copulas as functions for statistical regression. These are introduced in Chapter 5, along with the theoretical framework for

constructing Archimedean multi-variate copulas. Chapter 6 defines the quantile regression using copulas. The chapter presents the developed regression equations for the multi-variate Archimedean copulas, which is one of the main contributions of this dissertation. Chapter 7 provides results of three groups of experiments based on randomly generated artificial decision tables. The experiments are performed for uniform distribution of attributes as the least informative setting of decision tables. Chapter 8 provides different examples on which the modeling process are presented starting from qualitative model, copula regression and ranking. Different typical examples are provided based on artificial decision tables in order to describe the behavior of the method for symmetric decision tables, functions where linear regression methods breach the monotonicity and resolving decision tables with ties. Chapter 9 presents the applicability of the copula-based method on two real case examples in two different domains: ranking of electrically commutated motors and ranking of data-mining workflows. Chapter 10 gives conclusion remarks and directions for future research.

2 Formal Description of the Problem

The thesis addresses the problem of qualitative option ranking, when the set of options is accompanied with the decision makers' preferences. There are three prevailing approaches designed to support the preference modeling in MCDA:

1. Multiple Attribute Utility Theory (MAUT),
2. outranking methods,
3. logical "if ..., then ..." decision rules.

MAUT exploits the idea of assigning a score to each alternative. In outranking methods the preferences of the Decision maker (DM) are given as pairwise comparison of options. The third approach presents the preferential information in terms of exemplary decisions by building preferential model that consists of "if ..., then ..." decision rules. The thesis uses the later approach for defining the preference model which has several properties (Greco and Matarazzo, 2005):

1. it is expressed in natural language,
2. its interpretation is immediate, and
3. it can represent situations of hesitation.

Based on the given preference model, a function for option ranking is estimated.

This chapter provides the basic definitions that will be used throughout the thesis.

2.1 Problem Formulation

Decision problems that are of interest in the thesis are represented in the form of a Decision table (DT), whose separate rows refer to distinct options $u_i, i = 1,2,\ldots,r$, and columns refer to different attributes $A_j, j = 1,2,\ldots,n$. Based on the attribute's values there are two types of decision tables: a qualitative decision table, and a quantitative decision table.

Definition 2.1. *A qualitative decision table (QDT) is a 5–tuple*

$$QDT =< U, QA, QC, QV, Qf >$$

where

U is a finite set of r options (index of options),

$QA = \{QA_1, QA_2, \ldots, QA_n\}$ is a finite set of n qualitative condition attributes,

QC is a qualitative decision or class attribute

QV_q is the domain of qualitative attribute q, $q = \{1,2,\ldots,n_c\}$, $QV = QV_1 \times \cdots \times QV_n \times QV_c$ and

$Qf : U \times QA \to QV_c$ *is a qualitative mapping function.*

Definition 2.2. *A quantitative DT is a 5–tuple*

$$DT = < U, A, C, V, qf >$$

where

U *is a finite set of r options (index of options),*

$A = \{A_1, A_2, \ldots, A_n\}$ *is a finite set of n quantitative condition attributes,*

C *is a quantitative decision or class attribute*

V_q *is the domain of attribute q,* $q = \{1, 2, \ldots, n_c\}$, $V = V_1 \times \cdots \times V_n \times V_c$ *and*

$qf : U \times A \to V_c$ *is a quantitative mapping function.*

In this thesis, the DTs are obtained from QDTs using the mapping function F that is defined as follows:

Definition 2.3. *A mapping function* $F : QV_i \to V_i$*, is a function that maps the qualitative attributes' domains into quantitative ones, so that:*

$$F(QV_i) = V_i, where\ min(V_i) = 1, max(V_i) = p,\ where\ |QV_i| = p, p \in \mathbb{Z}, \forall i \in \{1, 2, \ldots, n_c\}.$$

The values of U are the same in both decision tables. U represents the index of the options. The i^{th} option in QDT and DT will be denoted as $a^{(i)}$, where $i \in U$. The domain of QV_q is represented with discrete, ordered qualitative values.

The labels of the attributes in both tables are changed from QA and QC to A and C, respectively, expressing the distinction that QA and QC refer to a decision table with qualitative domain of attributes while A and C refer to a decision table with quantitative domain of attributes.

In this thesis we will use QDTs with attributes whose values are preferentially ordered. We will distinguish the following preference relations between qualitative attribute values:

- $a^{(i)} \succ_q a^{(j)}$ denotes that $a^{(i)}$ is strictly preferred to $a^{(j)}$ with respect to the q^{th} attribute, $q \in \{QA \cup QC\}$,
- $a^{(i)} \prec_q a^{(j)}$ denotes that $a^{(j)}$ is strictly preferred to $a^{(i)}$ with respect to the q^{th} attribute, $q \in \{QA \cup QC\}$,
- $a^{(i)} \sim_q a^{(j)}$ denotes that $a^{(i)}$ is indistinguishable to $a^{(j)}$ with respect to the q^{th} attribute, $q \in \{QA \cup QC\}$.

In Definition 2.1, the mapping function Qf which maps the different combinations of attributes into a class attribute is called a *utility function*. It reflects the degree of preference of the decision maker for each option.

The mapping function F must preserve the preferences of the decision maker given in the qualitative decision table, i.e., for all preferential values of the input attributes $(x, y) \in QA_i$ or $(x, y) \in QC$, $x \neq y$, $i \in \{1, \ldots, n\}$ the following must hold:

$$\begin{aligned} (x \succ y) &\Rightarrow F(x) > F(y) \\ (x \sim y) &\Rightarrow F(x) = F(y) \\ (x \prec y) &\Rightarrow F(x) < F(y). \end{aligned} \tag{2.1}$$

These notations are mapped in the quantitative space, that is obtained by applying of the function F, in the following manner:

- $\succ$ is mapped to >, which denotes "is greater than"
- $\prec$ is mapped to <, which denotes "is less than"
- $\sim$ is mapped to =, which denotes "is equal to".

Definition 2.4. *A QDT (DT) is called symmetric if all attributes share the same domain and the evaluations of the class attribute are invariant to any permutation of attributes* $QA = \{QA_1, \ldots, QA_n\}$ *(*$A = \{A_1, \ldots, A_n\}$*).*

Definition 2.5. *A QDT (DT) is called partially symmetric if a subset of attributes share the same domain and the evaluations of the class attribute are invariant to any permutation of that subset of attributes* $QA = \{QA_1, \ldots, QA_r\}$ *(*$QA = \{A_1, \ldots, A_r\}$*),* $r \geq 2, r < n$*.*

If a QDT (DT) is neither symmetric nor partially symmetric, it is called non-symmetric.

Definition 2.6. *For* $\mathcal{A} = (A, C)$*, an aggregation function* f *that maps n real arguments into a real value is defined as:*

$$f : A \in \mathbb{R}^n \to \mathbb{R}. \tag{2.2}$$

In the QQ context, an aggregation function f should satisfy the following properties:

Property 1: Monotonicity (increasing) For $a, b \in A$,

$$(\forall i \in \{1, \ldots, n\} : a_i \geq b_i) \Rightarrow f(a) \geq f(b). \tag{2.3}$$

Property 2: Full ranking within classes For $(a, c), (b, c) \in \mathcal{A}$ where $a, b \in A$ and $c \in C$

$$(\exists i \in \{1, \ldots, n\} : a_i \neq b_i) \Rightarrow f(a) \neq f(b). \tag{2.4}$$

Property 3: Consistency preservation For $(a, c) \in \mathcal{A}$,

$$f(a_1, \ldots, a_n) \in [c - 0.5, c + 0.5]. \tag{2.5}$$

In addition to these properties, the aggregation function f should be symmetric for a symmetric DT.

Definition 2.7. *(Symmetry) (Beliakov et al., 2007) An aggregation function* f *is called symmetric, if its value does not depend on the permutation of the arguments, i.e.,*

$$f(a_1, a_2, \ldots, a_n) = f(a_{P(1)}, a_{P(2)}, \ldots, a_{P(n)})$$

for every a and every permutation $P = (P(1), P(s), \ldots, P(n)) of (1, 2, \ldots, n)$

Given Definitions 2.1–2.7 the problem addressed in this thesis reads:
Given the information in the QDT and the mapping function F***, find an aggregation function*** f ***that provides full option ranking for DT. The property of full option ranking should be relaxed, when that is desired, such as in cases of symmetric DT.***

2.2 Structure of the Decision Model

Decision models, which are used for representing a decision problem, may be given with

1. a linear structure or
2. a hierarchical structure of attributes.

In the linear structure, the attributes are given as a set, or are sorted according to some criteria, such as by descending importance of the attribute. This setting does not specify dependencies among the attributes because they are all given at the same level. It is the main limitation of these methods, as humans have the upper limit capacity to cope approximately with up to seven attributes at the same time. Hence, the linearly structured problems are limited to a small number of attributes. This problem may be solved by using a hierarchical structure of attributes, in which attributes are presented at several levels. A higher-level attribute, which is obtained by aggregation of lower level attributes, represents a class attribute as given with Definition 2.1.

This thesis considers models built with the DEX method, which have hierarchical structures. DEX is presented in more detail in section 3.2.

2.3 Variability as a Measure for Ranking

To consider a solution provided by the function f as a good one, it must fulfill the three properties (2.3)–(2.5). When two or more functions f fulfill (2.3)–(2.5), the one that provides the highest differentiability of options is preferred. Therefore, the most preferred ranking is the one with the highest spread of values in the intervals $[c \pm 0.5], c \in C$. For that purpose, the sum of mean absolute deviation D for each class k, $k = 1, \ldots, m$, where m is the number of classes, is used as a measure of variability:

$$D = \sum_{k=1}^{m} \frac{1}{n_k} \sum_{i=1}^{n_k} |x_{ik} - m_k(X)| \tag{2.6}$$

where $x_{ik} = f(A_i)$, $m_k(X)$ is the mean value of evaluated options that belong to class c_k, and n_k is the number of elements in the class c_k.

2.4 Running Example

In this section, the running example given in Table 2.1 is used to demonstrate the problem formulation provided in section 2.1. The example is used throughout the thesis to illustrate and compare the different algorithms and approaches for option ranking.

Table 2.1 is an example of a QDT. In Table 2.1, U is the universe of all nine options, therefore $r = 9$. The set of qualitative condition attributes is $QA = \{QA_1, QA_2\}$ thus $n = 2$. The qualitative class attribute is QC. The domain of the attributes is QV_1, QV_2, $QV_c \in \{good, better, the\ best\}$. The data in Table 2.1 specify the function Qf.

The preferential order of the attribute's and the class's values is: $the\ best \succ better \succ good$. This gives a partial ranking of the options, for instance, all 'better' options are preferred to all 'good' options. However, this gives no indication of option ranking within each class, even though it is clear that, for example, option 2 is better than option 1: both are classified as 'good', but option 2 is better with respect to the value of attribute QA_1. Therefore, the goal is to fully rank the options that belong to the same class solely by using the information contained in Table 2.1.

Table 2.1: Qualitative decision table

No.	QA_1	QA_2	QC
1	good	good	good
2	better	good	good
3	good	better	good
4	good	the best	good
5	the best	good	better
6	better	better	better
7	the best	better	the best
8	better	the best	the best
9	the best	the best	the best

Table 2.2: Quantitative decision table

No.	A_1	A_2	C	Ranking
1	1	1	1	0.7857
2	2	1	1	1.1429
3	1	2	1	1.0000
4	1	3	1	1.2143
5	3	1	2	2.1000
6	2	2	2	1.9000
7	3	2	3	2.9615
8	2	3	3	2.8077
9	3	3	3	3.1923

In order to perform ranking, a quantitative representation of the decision Table 2.1 is defined using the function $F : QV_i \to V_i$. Here one may demonstrate the properties of the function F, which are given with (2.1). For example, Table 2.2 is obtained from Table 2.1 using the mapping function F:

$$F(good) = 1,\ F(better) = 2 \ \text{ and } \ F(the\ best) = 3.$$

The function F defines the values of all rows in Table 2.2 with respect to columns 2–4. Following (2.1) we may write:

$$(better \succ good) \Rightarrow F(better) > F(good)$$
$$(the\ best \succ better) \Rightarrow F(the\ best) > F(better).$$

The next step is to find an aggregation function f and demonstrate that it fulfills (2.3)–(2.5). For Table 2.2, the function f is obtained by using the QQ method, which is presented in Chapter 3. The values of f for the given options are provided in the last column in Table 2.2. Based on these values, one may check that f satisfies the properties (2.3)–(2.5).

Checking of (2.3) is a two-step approach. The first step is to find a minimal set of all comparable groups of options in the given decision tables. For the given Table 2.2 the set is: {(1,2,5,7,9), (1,2,6,8), (1,3,4,8,9), (3,6,7), (6,7,9)}. To find this set, we first define all groups of comparable options. Then we select those which can not be represented as a subset of some of the other groups. The second step is checking (2.3) for all options that belong to the same group. For example, taking the group (6,7,9) and applying (2.3) leads to:

$$a^{(7)} \geq a^{(6)} \Rightarrow f(a^{(7)}) = 2.9615 \geq f(a^{(6)}) = 1.9000$$
$$a^{(9)} \geq a^{(7)} \Rightarrow f(a^{(9)}) = 3.1923 \geq f(a^{(7)}) = 2.9615$$

The variability is calculated according to (2.6) as:

$$\begin{aligned} D = &\frac{1}{4}\{|0.7963 - 1.0357| + |1.1429 - 1.0357| + |1.000 - 1.0357| + |1.2143 - 1.0357|\} + \\ &+ \frac{1}{2}\{|2.1000 - 2.0000| + |1.9000 - 2.0000|\} + \\ &+ \frac{1}{3}\{|2.9615 - 2.9872| + |2.8077 - 2.9872| + |3.1923 - 2.9872|\} = 0.3796. \end{aligned}$$

The following chapters describe different approaches to obtaining the aggregation function f in an analytical format using only the information provided in a DT.

3 From Qualitative to Quantitative Multi-Criteria Models

DEX is a qualitative modeling methodology which provides a decision model that governs the preferences of the decision maker represented as qualitative decision tables. In order to solve the task of ranking of options with equal preference, the QQ method is used (Bohanec, 2006; Bohanec et al., 1992). QQ is a three-stage method based on linear regression for evaluation of options. The usage of linear functions in QQ for ranking is appropriate for linear, or nearly linear decision tables. A decision table is considered nearly linear if it can be 'sufficiently well' (by some distance measure) approximated by some linear function.

In this chapter we first provide an overview of the related state-of-the-art qualitative decision making methods. Then we present the details of the DEX methodology, followed by the QQ method.

3.1 Qualitative Decision Making Methods

In qualitative decision making one may distinguish two major groups of methods. The first one is based on interactive questioning procedure used for obtaining the DM's preferences and final evaluation of options. These methods do not allow inconsistent judgements, which are solved by asking the DM to decide upon them. Two methods belong in this group. The first one is Measuring Attractiveness by a Categorical Based Evaluation Technique (MACBETH) that uses the semantic judgements about differences in attractiveness of several attributes to quantify the relative preferability of individual options (Bana e Costa et al., 1999).

The name of the second method is ZAPROS (an abbreviation of Russian words for Closed Procedures near Reference Situations). It is based on the verbal decision analysis approach that provides outranking relationships among options (Larichev, 2001a,b; Moshkovich and Larichev, 1995). It is designed to deal with a large number of options, however the number of criteria should be relatively small.

The second group of methods avoids the long interactive questioning procedures by employing the preference disaggregation principal (PDA) (Jacquet-Lagreze and Siskos, 2001). PDA requires a set of reference options for which the DM knows his/her preferences. Based on the preferential structures in the reference set, the preference models for evaluation of options are obtained. Four methods are considered in this group: UTA (UTilité Additive stands for additive utility), DRSA (Dominance-based Rough Set Approach), Doctus and DEX.

UTA method (Jacquet-Lagreze and Siskos, 1982) is considered as the best representative of the PDA. The DMs' preferences are given as a weak order of a reference subset of alternatives. UTA uses the DM's preferences as constraints in the linear programming (LP) and it assesses an additive utility function (AUF) used for option ranking. The AUF is piece-wise linear on arbitrary chosen intervals. UTA is based on two assumptions. Firstly, it assumes a preferential independence of the criteria for the DM. Secondly, the method assumes existence of an AUF. Consequently, when the AUF does not exist, it is assumed that the given DM's preferences are 'irrational' in the sense of exhibiting intransitive preferences, that the preferences are not independent, or that preferences are not monotonically increasing (Beuthe and Scannella, 2001). To deal with these issues, a plethora of UTA-based methods have been developed (Figueira et al., 2005). The most recent ones, UTA^{GMS} (generalizes UTA), UTADIS^{GMS} (a variant of UTA for sorting and classification) and GRIP (generalization of UTA by considering both pairwise

comparison and intensities of preferences), are developed by applying the concept of a robust ordinal regression (ROR). ROR takes into account all AUFs compatible with the preferential information (Greco et al., 2010). The result of the method considers two preference relations: the necessary and the possible one. The necessary preference relation is the one where option *a* is necessarily preferred to option *b*, if *a* is at least as good as *b* for all compatible value functions, while *a* is possibly preferred to option *b*, if *a* is at least as good as *b* for at least one compatible value function. To support the DM in situations when the preference statements cannot be represented in terms of AUF, methods introduce interactions with the decision maker in the process of defining the pairwise comparisons. Two solutions are proposed: the DM can work with AUF which is not fully compatible with preferences, or to remove some of the preference information causing the incompatibility (Greco et al., 2008). UTA originally does not deal with hierarchical structure of attributes. A recent extension of UTA introduces this concept under the name of Multiple Criteria Hierarchy Process (Corrente et al., 2012). The main drawback of the methods is their inability to represent interactions among criteria due to the limitation of the AUF. Hence two aggregation models are proposed defining the interaction of criteria. The first one builds non-additive utility function by engaging a specific fuzzy integral, the Choquet integral (Angilella et al., 2004). This results in obtaining weights that are interpreted as the "importance" of coalitions of criteria. The second one redefines the additive value function in UTA by adding terms such as "bonuses" and "penalty" in order to define the interaction among the criteria (Greco et al., 2012).

The preferential independence of the criteria is avoided by the last three methods in this group, which represents the DMs' preferences in terms of "if ... ,then ..." decision (or production) rules. The decision rules are given in a data table. The methods differ from other approaches by the possibility to handle inconsistencies in the DMs preferences, which may result from several reasons: hesitation of the DM, indiscernibility of some attributes or non-linearities imposed by some attributes.

DRSA uses the basis of rough sets theory with primary goal of solving classification and sorting problems in MCDA (Greco et al., 2001). However, DRSA can be used also for ranking and choosing options, by converting the data table into pairwise comparison table (PCT).

Doctus (Baracskai and Dörfler, 2003) is a Knowledge-Based Expert System Shell used for evaluation of decision options that are called cases. There are three types of evaluation of cases, called reasoning: Rule-Based Reasoning (RBR), Case-Based Reasoning (CBR), and Case-Based Rule Reasoning (CBRR). In RBR, the method uses "if ..., then ..." production rules provided by the decision maker based on which the evaluation of cases is performed. If the decision maker cannot articulate the rules, but he can provide important cases, then the CBR is used. In CBR a decision tree is built in order to define the evaluation rules. The CBRR is used to decrease the number of attributes given by the cases in CBR and RBR. The reasoning in Doctus leads to partial ordering of cases.

DEX was developed independently of DRSA and Doctus, and implemented in the DEXi software package (Bohanec, 2013). It decomposes a multi-attribute decision problem into smaller parts leading to a hierarchical evaluation model which contains the dependencies among attributes. In order to provide full ranking of options, the QQ method is used. The following two sections provide details about DEX and QQ.

3.2 Qualitative Modeling with DEX

DEX belongs to the group of qualitative multi-criteria decision making (MCDM) methods. In DEX, the qualitative attributes build a hierarchical structure which represents a decomposition of the decision problem into smaller, less complex and possibly easier to solve sub-problems. There are two types of attributes in DEX: basic attributes and aggregated ones. The former are the directly measurable attributes, also called input attributes, that are used for describing the options. The latter are obtained by

aggregating the basic and/or other aggregated attributes. They represent the evaluations of the options. The hierarchical structure in DEX represents a tree. In the tree, attributes are structured so that there is only one path from each aggregate attribute to the root of the tree. The path contains the dependencies among attributes such that the higher-level attributes depend on their immediate descendants in the tree. This dependency is defined by a utility function. The higher-level attribute, its immediate descendants and the utility function form a qualitative decision table as defined by Definition 2.1.

In DEX, the aggregation of the qualitative attributes into a qualitative class in each row in the decision table is interpreted as *if-then* rule. Specifically, the decision maker's preferences over the available options are given with the attribute that is called a qualitative class, or only class as given with Definition 2.1. Options that are almost equally preferred belong to the same qualitative class.

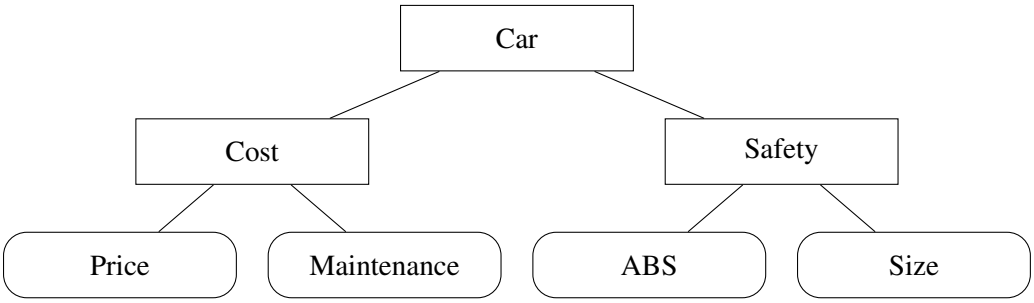


Figure 3.1: Hierarchy of attributes for evaluation of cars

An example of a DEX model tree that is used for the evaluation of cars is presented in Figure 3.1. The basic attributes in Figure 3.1 are given with rectangles with curved edges, such as *Price, Maintenance, ABS* and *Size*. The aggregated ones are given with rectangles with sharp edges, such as *Costs, Safety* and *Car*. The value scales of each attribute are given in Table 3.1, which is obtained from the implementation of the DEX model for car assessment in the computer program DEXi. The aggregation process in DEX results in a partial ranking of options, meaning that several options may be evaluated to belong in the same qualitative class, thus making them indistinguishable.

Table 3.1: DEXi model tree and attribute scales for assessment of cars

Attribute	Scale
CAR	low, acceptable, medium, good, *excellent*
├─ COSTS	high, medium, *low*
│ ├─ Price	high, medium, *low*
│ └─ Maintenance	expensive, medium, *cheap*
└─ SAFETY	low, acceptable, good, *excellent*
├─ ABS	no, *yes*
└─ Size	small, medium, *big*

For example, the aggregation of the qualitative basic and aggregated attributes in a hierarchical set up of tables is given with Tables 3.2, 3.3 and 3.4. The given tables show that several options may be evaluated as equal. For example, several *Car*s are evaluated as *medium*, but one may not distinguish among them.

Starting from an existing DEX model, and Definitions 2.1–2.7, the goal in the thesis is to find an aggregation function f which is able to differentiate among the options in the same class, possibly by providing full ranking of the options.

Table 3.2: Car aggregation

Costs	Safety	Car
low	excl.	excl.
low	good	good
low	accept.	medium
low	low	low
medium	excl.	medium
medium	good	accept.
medium	accept.	medium
medium	low	low
high	excl.	good
high	good	medium
high	accept.	low
high	low	low

Table 3.3: Costs aggregation

Price	Maint.	Costs
low	cheap	low
low	medium	low
low	exp.	medium
medium	cheap	low
medium	medium	medium
medium	exp.	high
high	cheap	high
high	medium	high
high	exp.	high

Table 3.4: Safety aggregation

ABS	Size	Safety
no	small	low
no	medium	accep.
no	big	good
yes	small	low
yes	medium	good
yes	big	excl.

3.3 The Qualitative-Quantitative Method

The QQ method (Bohanec, 2006; Bohanec et al., 1992) was developed as an extension to the DEX method (Bohanec and Rajkovič, 1990; Bohanec et al., 2012) with the aim of option ranking within classes. The goal of QQ is finding a function f as defined in (2.2) that would provide full ranking of options that belong to the same class.

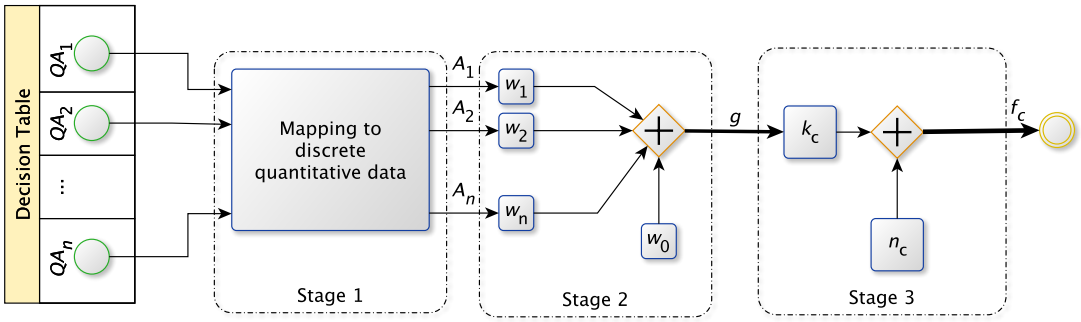


Figure 3.2: Three stages of the QQ method

QQ consists of three stages, as schematically presented in Figure 3.2. In the first stage, the values of the qualitative attributes $QA_1,\ldots,QA_n$ and the qualitative class QC are mapped into discrete quantitative values $A_1,\ldots,A_n,C \in \mathbb{Z}$, using the mapping $F : QV_i, \rightarrow V_i, i = \{1,\ldots,n+1\}$ given with Definition 2.3 (step 1 in Figure 3.2). As a result, a numerical table is obtained such as the one given in Table 2.2, that represents an output of the first stage of QQ, and an input to the second stage of the method. For example, let us use Tables 3.2, 3.3 and 3.4 as inputs to QQ. In the first stage, QQ maps them in Tables 3.5, 3.6 and 3.7, respectively. [1]

In the second stage, QQ estimates a regression function[2] $g : \mathbb{R}^n \rightarrow \mathbb{R}$ that

$$A_{agg} = g(A_1,\ldots,A_n). \tag{3.1}$$

[1] Note that Table 3.6 is equivalent to the running example given in Table 2.2

[2] Despite $A_i \in \mathbb{N}$, the function $g(A_1,\ldots,A_n)$ is defined in $\mathbb{R}^n$

Table 3.5: Mapping to quantitive aggregation table of **Car**

Costs	Safety	Car
3	4	5
3	3	4
3	2	3
3	1	1
2	4	3
2	3	3
2	2	2
2	1	1
1	4	4
1	3	3
1	2	1
1	1	1

Table 3.6: Mapping to quantitive aggregation table of **Costs**

Price	Maint.	Costs
3	3	3
3	2	3
3	1	2
2	3	3
2	2	2
2	1	1
1	3	1
1	2	1
1	1	1

Table 3.7: Mapping to quantitive aggregation table of **Safety**

ABS	Size	Safety
1	1	1
1	2	2
1	3	3
2	1	1
2	2	3
2	3	4

The most frequent approach of defining g in (3.1) is the usage of additive functions as a result of their simplicity (Malakooti, 2011).

QQ uses the following linear regression function for option evaluation:

$$A_{agg} = \sum_i w_i A_i + w_0 \tag{3.2}$$

and defines the relation between the aggregated (dependent) attribute A_{agg} and input attributes A_i. A_{agg} is an estimation of the class attribute C. In (3.2), A_i are attributes and w_i are weights obtained by the method of least squares. For example, for options given in Table 3.6, the equation (3.2) has the form:

$$g = 0.833A_1 + 0.500A_2 - 0.778.$$

The third stage of QQ ensures the consistency between the qualitative and quantitative models. It means that whenever the former yields the qualitative class, the latter should yield a numerical value in the interval $[c_i - 0.5, c_i + 0.5], c_i \in C$.

This is of interest in hierarchical evaluation models in which the values of the aggregation of basic attributes into a class attribute are propagated in the next higher level as an input attribute. These are further on aggregated, and the procedure is repeated to the top most aggregation (class) attribute. This means that the arguments of (3.1) are not integers, but real numbers spread in the interval [a - 0.5, a + 0.5], where a is some ordinal value of the attribute A. Consequently, the range of A_i is [0.5, m + 0.5] where m is the number of values that receives the respective qualitative attribute QA_i.

When using QQ for aggregation, the evaluation result represents a continuous value, which may not capture the information about the class into which a certain option belongs to. Therefore QQ introduces the third step, of ensuring that the evaluation result belongs into the interval $[c_i - 0.5, c_i + 0.5], c_i \in C$. To achieve this, for the regression function (3.2), a set of functions f_c is defined which ensures compliance with the original class c_i. We call this process normalization, which in addition should achieve the maximal spread of the rankings in the class. The set of ranking functions that ensures compliance with the qualitative model is:

$$f_c(A_1, \ldots, A_n) = k_c g(A_1, \ldots, A_n) + n_c, \tag{3.3}$$

where k_c and n_c are given with

$$k_c = \frac{1}{max_c - min_c} \tag{3.4}$$

$$n_c = c + 0.5 - k_c max_c. \tag{3.5}$$

In (3.4) and (3.5), k_c and n_c are parameters for the normalization of the function g. Here c is the value of the class. In order to obtain the values of k_c and n_c, QQ uses Algorithm 1 for each of the classes.

Algorithm 1 Calculations of weights k_c and n_c in QQ method

1: **for** $\forall el_i \in c_i$ where $el_i = \{A_{1i}, A_{2i}, \cdots A_{ni}\}$ **do** ▷ for each option el_i that belongs to class c_i
2: $G_{c_i} \leftarrow \{g(\forall A_{ji} \in el_i \pm 0.5)\}$ ▷ find all values of g for all combinations of $A_{ji} \pm 0.5$
3: $min_c = \min(G_{c_i})$ ▷ find the minimum of the function g in class c_i
4: $max_c = \max(G_{c_i})$ ▷ find the maximum of the function g in class c_i
5: $k_c = \frac{1}{max_c - min_c}$ ▷ calculate the coefficient k_c
6: $n_c = c_i + 0.5 - k_c max_c$ ▷ calculate the coefficient n_c
7: **end for**

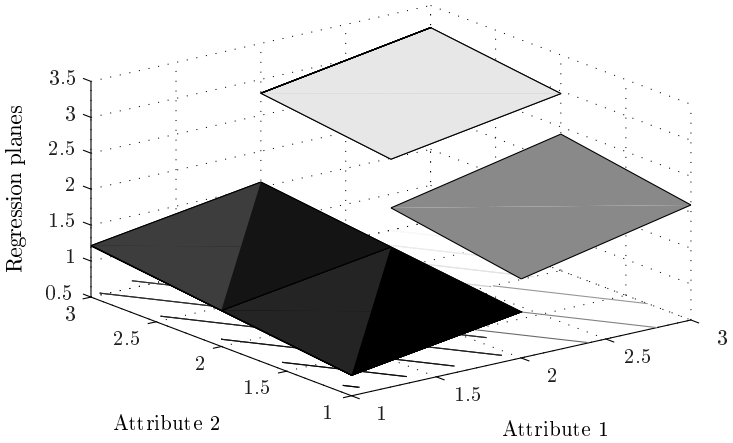


Figure 3.3: Estimated regression curves obtained with QQ for options given in Table 2.2

Each linear function in equation (3.3) represents a model for the corresponding class in the originally defined qualitative decision table. These functions are used to rank the options in the classes. For example, the options given in Table 3.6 are ranked with the following functions f_c:

$$f_c = \begin{cases} 0.4286g + 0.5476, & \text{if } c = 1; \\ 0.6000g + 0.7667, & \text{if } c = 2; \\ 0.4615g + 1.7051, & \text{if } c = 3. \end{cases}$$

These are shown in Figure 3.3. The final evaluation of the options is presented in column *Evaluation* in Table 3.8. These values are used for ranking of the options. The higher evaluation value leads to higher rank of the option.

This method encounters the following main difficulties:

1. It is restricted to using linear functions in the second stage of the method given with (3.2), which leads to satisfactory performances only for linear or nearly linear decision tables. The goal in the thesis is to improve QQ so that it would lead to satisfactory rankings in case of non-linear and/or non-monotone decision tables.

Table 3.8: Quantitative ranking of options

No.	Price	Maint.	Costs	Evaluation
1	3	3	3	3.192
2	3	2	3	2.962
3	3	1	2	2.100
4	2	3	3	2.808
5	2	2	2	1.900
6	2	1	1	1.143
7	1	3	1	1.214
8	1	2	1	1.000
9	1	1	1	0.785

2. QQ is limited to discrete attributes due to the mappings in the first stage. Another goal of the thesis is to find a way to include other types of attributes (including the class attribute), such as probabilistic values, sets, interval or fuzzy values.

To overcome the limitations, and to achieve the above goals, the thesis investigates the following modifications of QQ:

1. To tackle the first problem, different impurity functions are examined to determine the weights in the linear regression estimator.

2. Introduction of polynomial functions instead of the linear function in the second stage of QQ.

3. The problem is redefined as a constraint optimization problem and a solution using linear programming is sought.

4. In order to tackle the second difficulty when using QQ, a solution in the probabilistic space is proposed, leading to probabilistic regression.

The first three modifications of QQ are explained in Chapter 4. The last proposal is explained in Chapters 5 and 6.

4 Modifications of QQ with Impurity Functions, Polynomials and Optimization Functions

To provide a way of ranking non-linear non-monotone decision tables we studied four methods in the thesis:

1. Usage of impurity functions for weights estimation in the linear regression function;
2. Replacement of the linear function with a polynomial regression equation;
3. Reformulation of the quantitative problem of option ranking as an optimization problem, and providing insight into its limitations.
4. Usage of copula-based functions for regression and consequently for option ranking.

The first three methods are described in this chapter. The last one differs from the first three in two main aspects: it does not use weights in the regression equation for option ranking, and it redefines the attributes as random variables. To provide an extensive background for such an approach, the method is separately described in Chapters 5 and 6.

4.1 Impurity Functions for Weights Estimation in QQ

In order to improve the performance of QQ, different weights estimators are examined for which it was expected to provide better results than QQ. QQ estimates weights in (3.2) by least square regression, which is based on an often too strong assumption that the underlying quantitative mapping is linear or nearly linear. Alternatively, one can use alternative methods for estimating the weights, hence circumventing this assumption. In particular, we use the impurity functions. The impurity functions are defined to measure the goodness of a split at a node for a given variable in a decision tree which is used in machine learning and data mining (Izenman, 2008). Here, the impurity functions are used to determine the similarity between an input attribute and the output attribute (the class attribute). Following the definition of node impurity function (Izenman, 2008), we define the impurity function between the input and class attribute as follows.

Definition 4.1. *Let $c_1,\dots,c_k$, $k \geq 2$, are the classes of the output attribute C. For the input attribute A, an impurity function $i(A)$ between the input and the class attribute is:*

$$i(A) = \sum_{i=1}^{k} \mu(p(a_1|c_i),\dots,p(a_n|c_i))$$

where $a_1,\dots,a_n$ are all possible values that the attribute A may receive, and $p(a_j|c_i)$ is an estimate of the conditional probability that an observation a_j is in c_i.

The function μ:

1. is symmetric (its value does not depend on the permutation of the arguments),

2. defined on the set $(p_{a_1},\ldots,p_{a_n})$ that sums to a unit,
3. have minimum at points $(1,0,\ldots,0),(0,1,\ldots,0),\ldots,(0,0,\ldots,1)$, and have maximum at the point $(\frac{1}{K},\ldots,\frac{1}{K})$.

The following impurity functions are examined here: Gini index, information gain and χ^2. These impurity functions are used in the following subsections for weights estimation in (3.2) with $w_0 = 0$. For each of the input attributes, the value of the impurity function is calculated between the input attribute and the class attribute, followed by normalization of the obtained values in the unit interval. The normalized values are regarded as weights in (3.2).

4.1.1 Gini Index

The Gini index was firstly proposed by Italian statistician Corrado Gini (Xu, 2004) as a measure of income inequality. It is mathematically defined as a ratio between the Lorenz curve that plots the income of population versus population and perfect equality of income, as shown in Figure 4.1. It is also defined

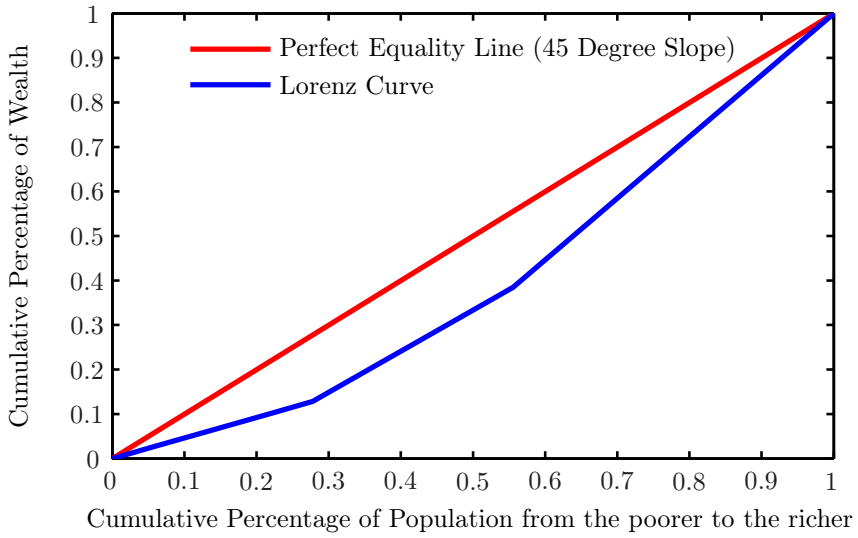


Figure 4.1: Gini index is calculated as a ratio between the Lorenz curve and the perfect equality line

as second order of the generalized information function by Louis (1996). In his work, Louis starts from defining the entropy of type β, where β is a constant such as $\beta > 0, \beta \neq 1$. For a discrete probability distribution $(p_1,\cdots,p_m)$, the generalized information function reads:

$$H^{\beta}(p_1,\cdots,p_m) = \sum_{i=1}^{m} p_i u^{\beta}(p_i) \tag{4.1}$$

where $u^{\beta}(p_i)$ is uncertainty defined by

$$u^{\beta}(p_i) = \frac{2^{\beta-1}}{2^{\beta-1}-1}(1-p_i^{\beta-1}). \tag{4.2}$$

The measure u^{β} is strictly decreasing function of p_i. When $\beta = 2$ in (4.1) and (4.2) it follows:

$$H^2 = 2[1-\sum_{i=1}^{m} p_i^2] = 4\sum_{i\neq j} p_i p_j = 2\sum_{i=1}^{m} p_i(1-p_i). \tag{4.3}$$

Equation (4.3) is known as Gini index and it was firstly used in machine learning by Breiman et al. (1984). Since its proposal, the Gini index has been used in many different areas to measure different kinds of distributions. In machine learning it is used for making splits in decision trees (Breiman et al., 1984) and for representation of the performances of different classifiers (Hand and Till, 2001). In this thesis, three estimates of the Gini index are used to define the weights in (3.2): Gini Breiman (gB), Gini covariance (gC), and Gini population (gP).

The gB approach The estimation of the Gini index according to Breiman et al. (1984) is calculated for each of the attributes A_j as follows:

$$gB = 1 - \sum_{k=1}^{m} p^2(c_k). \tag{4.4}$$

In (4.4), $p(c_k)$ is the estimated probability that the class attribute C obtains the value k, $k = 1,\ldots,m$. The weights in (3.2) are obtained through the importance of each attribute A_i, that is calculated as (Kononenko, 1997):

$$gB(A_i) = \sum_{j=1}^{p} p(A_{ij}) \sum_{k=1}^{m} p^2(c_k|A_{ij}) - \sum_{k=1}^{m} p^2(c_k).$$

Here A_i is the i^{th} attribute, $p(A_{ij})$ is the probability that the attribute A_i obtains the value j, $p(c_k|A_{ij})$ is the conditional probability that the attribute A_i receives the value j and is classified in c_k.

For the running example given with Table 3.6, the importance of the attributes is $gB(A_1) = 0.2716$ and $gB(A_2) = 0.1235$, which, when normalized, lead to the following weights: $\omega_1 = 0.6875$ and $\omega_2 = 0.3125$.

The gC approach Calculating the Gini index by using a covariance matrix was introduced by Xu (2004), and it has the following form

$$gC = \frac{2\text{cov}(A_i, C)}{s}$$

where A_i and C are the input and the class attribute C and s is the mean of A_i. The notation $cov(A_i, C)$ denotes the covariance between the two attributes A_i and C. It is a measure of linear dependency between random variables and is calculated as:

$$\text{cov}(A_i, C) = E(A_i C) - (EA_i)(EC)$$

where $E(X) = \sum_{i=1}^{\infty} x_i p_i$ is the expected value of the discrete random variable X that receives the values $x_1, x_2, \ldots, x_i$, with probabilities $p_1, p_2, \ldots, p_i$. It is a weighted average of all possible values that the random variable may take.

Using gC, the importance of the attributes in Table 3.6 is $gC(A_1) = 0.0649$ and $gC(A_2) = 0.0147$, which, when normalized, give the following weights: $\omega_1 = 0.6250$ and $\omega_2 = 0.3750$.

The gP approach Noorbakhsh (2007) used the source form of the Gini index defined by Gini, for calculating wealth distribution among population. It has the following form:

$$gP = \frac{1}{\mu} \sum_{i=1}^{r} \sum_{j=1}^{r} p_i p_j |c_i - c_j| \tag{4.5}$$

where r is the number of options, and

$$\mu = \sum_{i=1}^{r} c_i p_i$$

and

$$p_i = \frac{A_i}{\sum_{i=1}^{r} A_i}.$$

This approach leads to the following calculations for the attributes in Table 3.6; $gP(A_1) = 0.2066$ and $gP(A_2) = 0.2357$. These values are normalized to the following weights values: $\omega_1 = 0.4670$ and $\omega_2 = 0.5329$.

4.1.2 Chi Square χ^2

The distribution of χ^2 has its origin in statistics and was devised as a test of goodness of fit of an observed distribution to a theoretical one (Fisher, 1924). As an impurity measure it was firstly used in the algorithm CHAID (Kaas, 1980). In this thesis, χ^2 is used for measuring the association between each of the input attributes and the class attribute under the hypothesis of independence.

Determination of the value of χ^2 is a two-step process for a given contingency table. A contingency table is a two-entry frequency table that reports the joint frequencies of two variables. Here the variables are an input attribute and the class attribute. The first step for obtaining χ^2 is calculation of the expected value for each cell in the contingency table. The second step is comparison of the expected values with the observed values using:

$$\chi^2(A_j, C) = \sum_{i=1}^{|A_j|} \sum_{j=1}^{|C|} \frac{(x(i,j) - E_{i,j})^2}{E_{i,j}} \tag{4.6}$$

where A_j is the j^{th} attribute, C is the class attribute, $|A_j|$ and $|C|$ are the number of different values that the input and class attribute may receive, respectivly, $x(i,j)$ are observed frequencies in cell (i,j) in the contingency table, and $E_{i,j}$ is the corresponding expected value under the assumption of independence (Härdle and Simar, 2007):

$$E_{i,j} = \frac{x_{i0} x_{j0}}{x_{00}}.$$

Here x_{i0} and x_{j0} are observed frequencies in each row and column in the contingency table respectively, and

$$x_{00} = \sum_{i=1}^{|C|} x_{i0}.$$

The values for χ^2, obtained between each input attribute and the class attribute, are finally normalized in order to obtain the weight of each of the attributes in (3.2).

To demonstrate the usage of χ^2 for weights calculation, consider the running example given in Table 2.2. The contingency tables and the values of x_{i0}, x_{j0} and x_{00} between A_1 and C, and between A_2 and C are given with Tables 4.1 and 4.2, respectivlly.

Table 4.1: Contingency table, x_{i0}, x_{j0} and x_{00} between $a_1 \in A_1$ and $c_i \in C$ for the running example in Table 2.2

x(i,j)	$c_1 = 1$	$c_2 = 2$	$c_3 = 3$	x_{j0}
$a_1 = 1$	3	0	0	3
$a_1 = 2$	1	1	1	3
$a_1 = 3$	0	1	2	3
x_{i0}	4	2	3	$x_{00} = 9$

Table 4.2: Contingency table, x_{i0}, x_{j0} and x_{00} between $a_2 \in A_2$ and $c_i \in C$ for the running example in Table 2.2

x(i,j)	$c_1 = 1$	$c_2 = 2$	$c_3 = 3$	x_{j0}
$a_2 = 1$	2	1	0	3
$a_2 = 2$	1	1	1	3
$a_2 = 3$	1	0	2	3
x_{i0}	4	2	3	$x_{00} = 9.$

The values of χ^2 are $\chi^2_1(A_1,C) = 6.5$ and $\chi^2_2(A_2,C) = 3.5$, which, when normalized, lead to the following weights values in (3.2): $\omega_1 = 0.65$ and $\omega_1 = 0.35$.

4.1.3 Information Gain

Information gain (IG) has its origin in information theory and it is frequently used in decision tree learning for determining the attribute that gives most information regarding some splitting criteria. It is defined as:

$$IG(C,A_j) = H(C) - \sum_{q=1}^{p} \frac{|AS_q|}{r} H(c_q) \tag{4.7}$$

where C is the class attribute, A_j is the j^{th} input attribute, r is the number of options, p is the number of values that the attribute A_j may receive, $|AS_q|$ is the number of options that receive the same value AS_q, c_q is the subset of the class attribute for which the attribute A_j receives the q^{th} value. The equation (4.7) may be expanded with the equation for entropy H, leading to (Raileanu and Stoffel, 2000):

$$IG(C,A_j) = -\sum_{k=1}^{m} p(c_k)\log(p(c_k)) + \sum_{q=1}^{p} p(a_q) \sum_{k=1}^{m} p(c_k|a_q)\log(p(c_k|a_q)) \tag{4.8}$$

where $a_q = \frac{|AS_q|}{r}$, $p(c_k)$ is the probability that a randomly selected example belongs to the class $c_k \in C$, $-\log(p(c_k))$ is the information that it conveys with, $p(a_q)$ is the probability that the attribute A_j will receive the value AS_q, and $p(c_k|a_q)$ is the conditional probability.

4.1.4 Weights Calculation with Impurity Functions

Each of the explained impurity functions (4.4)–(4.8) are used to calculate the weights in (3.2) (and setting $w_0 = 0$) by applying Algorithm 2.

Algorithm 2 Calculation of weights w_i with impurity function

1: **for** $i = 1 \rightarrow n$ **do** ▷ for each input attribute A_i, and class attribute C
2: $W_i = f(A_i,C)$ ▷ calculate the value of the impurity function $f(A_i,C)$
3: **end for**
4: $w_i \leftarrow norm(W_i)$ in the interval [0,1] ▷ obtain the weights w_i with normalization

In the next step, the set of regression functions (3.3) is calculated and finally the third stage of QQ is applied. This procedure has been used on the running example given in Table 2.2, and the final numerical results are presented in Table 4.3. These calculations lead to different regression curves given in Figures 4.2a–4.2e. The Figures show that regression curves are very similar to each other, and they

differ in the inclination angle. The most dissimilar are the curves obtained using gP approach, given in Figure 4.2c. Here the regression estimation curve for the second class given in Figure 4.2c provides an inverse rank estimation for the two options belonging in this class in comparison with the rankings obtained by the rest of the methods. This can be better noticed from the numerical calculations in Table 4.3, where option 6 is better ranked than option 5 only when gP method is used. Both rankings are considered as correct in this case, as all conditions for the required method given in Chapter 2 are fulfilled. The example shows that different methods lead to different rankings that can satisfy the requirements.

Table 4.3: Quantitative ranking of options with different impurity functions

No.	$\mathbf{A_1}$	$\mathbf{A_2}$	**C**	**gB**	**gC**	**gP**	**IG**	χ^2
1	1	1	1	0.7963	0.7857	0.7420	0.7910	0.7941
2	2	1	1	1.2037	1.1429	0.9681	1.1640	1.1765
3	1	2	1	0.9815	1.0000	1.0000	1.0000	1.0000
4	1	3	1	1.1667	1.2143	1.2580	1.2090	1.2059
5	3	1	2	2.1364	2.1000	1.9691	2.1099	2.1154
6	2	2	2	1.8636	1.9000	2.0309	1.8901	1.8846
7	3	2	3	3.0185	2.9615	2.8262	2.9765	2.9848
8	2	3	3	2.7963	2.8077	2.8692	2.8047	2.8030
9	3	3	3	3.2037	3.1923	3.1738	3.1953	3.1970

4.2 Polynomial Models for Regression

The linear regression, introduced in the second stage of QQ, may be replaced with a different form of regression, the polynomial regression. The polynomial regression introduces a regression model in a form of polynomial equation which predicts the value of the dependent variable y. A polynomial regression equation over the attributes $A_1, A_2, \ldots, A_n$ can be written as:

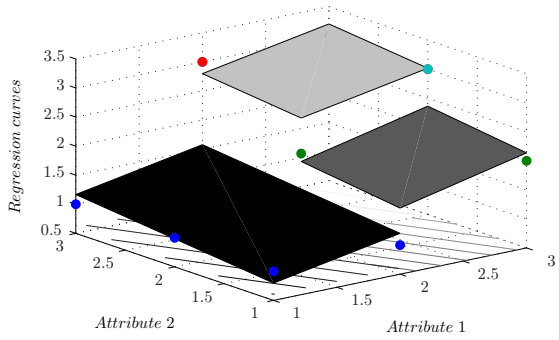
$$y = w_0 + \sum_{i}^{m} w_i T_i \tag{4.9}$$

where $T_i = \prod_{j=1}^{n} A_j^{u_{i,j}}$, and $u_{i,j} \in \mathbb{N}$. Here y is estimation of the class attribute C. To explore the usage of polynomial functions for regression in the second stage of QQ, the CIPER machine learning algorithm is used. There are two versions of the CIPER algorithm: CIPER (Todorovski et al., 2004) and New CIPER (Pečkov et al., 2008). Both algorithms are explained next.

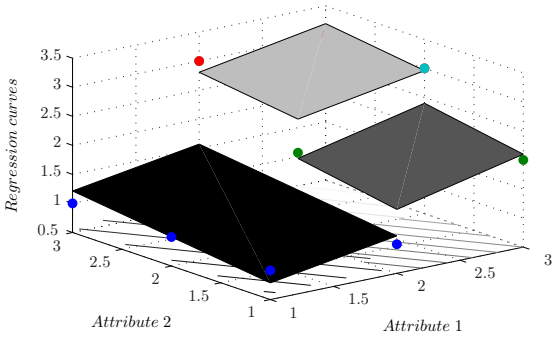
4.2.1 CIPER

CIPER is an algorithm that uses a specific heuristics to define and search the space of possible polynomial functions. As a result it finds one (or several) polynomial function that satisfies the heuristics and that provides best fit for the data. The heuristics is given with a set of constraints. In order to define the constrains, CIPER introduces the following notation for (4.9):

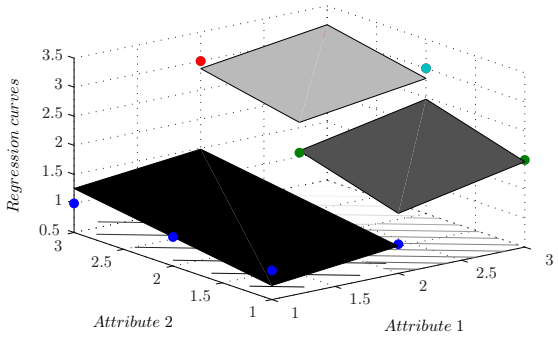
1. length of y is $Len(y) = \sum_{i=1}^{m} \sum_{j=1}^{n} u_{i,j}$,
2. the size of y (number of terms) is $size(y) = m$,
3. a degree of a term T_i is $Deg(T_i) = \sum_{j=1}^{n} u_{i,j}$ and



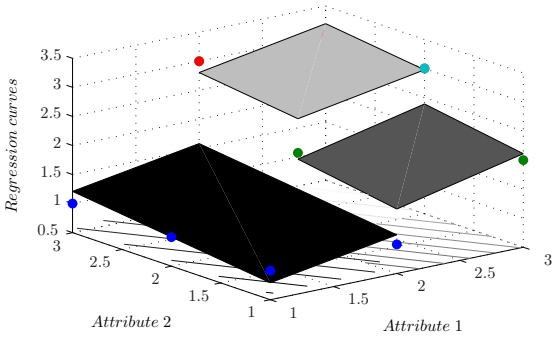
(a) Gini Breiman



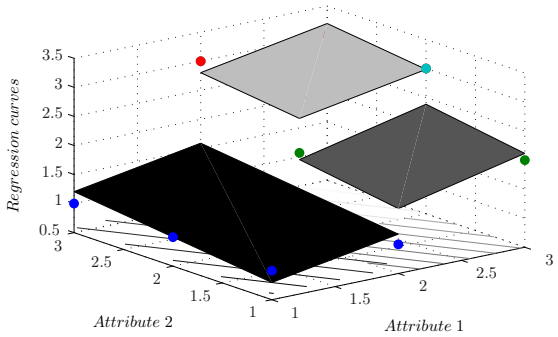
(b) Gini covariance



(c) Gini population



(d) Information gain



(e) χ^2

Figure 4.2: Regression curves obtained with different weights estimation in QQ

4. degree of y is $Deg(y) = max_{i=1}^{m} Deg(T_i)$.

In order to search the space of possible polynomial equations using the predefined heuristics, CIPER employs iterative beam search. The beam may be initialized in two ways: with the simplest polynomial equation or with some user-defined polynomial, that is considered as the minimal one in the beam search. In each iteration a new set of polynomials is generated from the existing polynomials in the beam by using some refinement operator.

A refinement operator is a function which takes as input some polynomial structure y and generates a new one by modifying y. The refinement operator in CIPER modifies y by increasing $Len(y)$ for one unit. It may be performed in two ways: by adding a new first degree term T_i or by increasing an existing term with a variable. The coefficients w_i in the newly created polynomial equations are calculated using the method of least squares.

Each of the generated polynomial equations are evaluated so that the degree of fit of the equation to the given data set is estimated. For evaluation of equations a minimum descriptor length (MDL) heuristics is used. CIPER uses an ad-hoc MDL heuristic given as (Todorovski et al., 2004):

$$MDL(y) = Len(y)\log(k) + k\log(MSE(y)) \tag{4.10}$$

where $MSE(y)$ is the mean squared error and k is the number of training examples. The first term in (4.10) represents a penalty for the equation complexity, while the second term measures the degree of fit of the equation. More preferred equations are those with smaller MDL.

During the search, CIPER maintains a set of b best possible equations in the beam that satisfy the imposed constraints. The search finishes when the refinement operator cannot generate any new equations whose evaluation outperforms the evaluation of the equations that are already kept in the beam.

4.2.2 New CIPER

The New CIPER improves and extends CIPER in following maners:

1. it provides an improved refinement operator,
2. it provides a new MDL Scheme for polynomial regression,
3. it employes error on unseen data as search heuristics.

The New CIPER extends CIPER so that it:

1. learns piecewise polynomial models,
2. is capable for multi-target polynomial regression,
3. performs classification via multi-target regression.

Each of the improvements and extensions are explained.

Improved refinement operator The need for improving the refinement operator is motivated from the fact that when a term is added to a polynomial, it decreases its error. On the other hand, when a term is replaced with a more complex version, such as a multiplication of the term with a variable, it does not decrease the error in general. For example, adding A_1 to A_2 yielding to $A_1 + A_2$ would reduce the error of the equation. However, the replacement of A_1 with A_1A_2 may increase the error of the equation. New CIPER introduces a third refinement operator, which takes a term in the equation, copy it and multiply it with the new variable. The obtained term is added to the existing equation. For example, the equation $A_1 + A_2$ is refined by coping the term A_1 and multiplying it with A_2. The new term is added to the equation leading to $A_1 + A_2 + A_1A_2$. This refinement operator increases the complexity of the newly generated polynomial equation. Therefore, in each step of the algorithm, for the newly generated equation, each term is removed and an evaluation of the equation is performed. If the evaluation outperforms the existing equation in the beam, the new equation is added to the beam.

New MDL Scheme for polynomial regression For evaluation of polynomial equations, the New CIPER uses the following MDL:

$$MDL(y) = L(y) + 2W(y, D)$$

where $L(y)$ is the number of bits for encoding the polynomial structure, and W is a stochastic complexity of a linear regression model that uses least squares. To calculate $L(y)$, the following equation is used (Pečkov et al., 2008):

$$L(y) = 2\log(l) + 2\log(\log(l)) + \log|G(n,l)| + \log|G'(u_1,u_2,\ldots,u_n)|.$$

Here $l = Len(y)$, $|G(n,l)|$ is the number of polynomial structures with n terms and length l. The class of polynomials that belong to $G(n,l)$ is partitioned into subclasses with fixed term degrees $G'(u_1,u_2,\ldots,u_n)$. All polynomials that belong to this subclass have n terms with degrees $u_1 \geq u_2 \geq \cdots \geq u_m$. $|G'(u_1,u_2,\ldots,u_n)|$ is the number of polynomial structures in $G'(u_1,u_2,\ldots,u_n)$.

W is calculated with (Rissanen, 2000):

$$W = \min_{\gamma}(N-p)\log(\hat{\tau}) + p\log(N,\hat{R}) + (N-p-1)\log(\frac{N}{N-p}) - (p+1)log(p)$$

where γ is an index that goes though all subsets of variables in the linear regression, N is the size of the data table, p is the number of elements in γ, $\hat{\tau}$ is the maximum likelihood estimation of the model error, and $\hat{R} = \frac{1}{N}\hat{w}^T(A^TA)\hat{w}$ where $\hat{w} = (A^TA)^{-1}A^Ty$.

Employing error on unseen data as search heuristics New CIPER introduces additional heuristics for evaluation of polynomial models as an alternative to the MDL. It is called CV (cross validation) heuristics. It considers the data set as a training set that is used to build the polynomial model. It splits the train set into 10 parts (options), and builds the model on 9 parts. It calculates the square relative error on 9 parts obtaining the value of the variable $reTrain^2$, and on the 10th part leading to the value of $reTest^2$. The procedure of splitting the training set and building a polynomial model is repeated according to a predefined number of times. For each of the modes, the tuple $(reTrain^2, reTest^2)$ is kept. The beam search is modified so that a derived model with the refinement operator may enter the beam only if its heuristics value is smaller than $min(reTrain^2, reTest^2)$.

In the process of generating new polynomial models, an over-fitting may occur. To avoid it, New CIPER poses a requirement that the $reTrain^2$ should be smaller than $reTest^2$. Otherwise the model is rejected from further refinement. This procedure decreases the probability of over-fitting. As soon as the models are built, a final model is produced by averaging all built models.

Learning piecewise polynomial models New CIPER introduces the possibility of including piecewise polynomial models in the beam search. It is designed so that it partitions the space along each attribute before the the search begins. The partitioning of the attribute's space is not dependent on the output attribute (class attribute).

The procedure for partitioning the array of real values A is the following. First, k arrays are obtained from A: $A_1, A_2, \ldots, A_k$. For the particular partitioning, a heuristics H is calculated according to:

$$H(A_1,A_2,\ldots,A_k) = k(\sum_{j=2}^{k}(\frac{m_j - m_{j-1}}{2})^2 + \sum_{x\in A}\min_{j=1}^{k}(x-m_j)^2) \tag{4.11}$$

where m_j is the mean of the values in A_j. The value of (4.11) is compared to the sum of variances:

$$\sum_{x\in A}(x - mean(A))^2. \tag{4.12}$$

If 4.11 is smaller than (4.12), then the splitting is called *possible split* with the number k for values in A. For a given k, the best *possible split* is given with $A_1^s, A_2^s, \ldots, A_k^s$ and it holds:

$$H(A_1^s, A_2^s, \cdots, A_k^s) \leq H(A_1, A_2, \cdots, A_k)$$

for any other partitioning $A_1, A_2, \cdots, A_k$ of the array A. Next, a function $P_k(A)$ is defined so that it finds the best partitioning of an array A into k arrays, where $k \in [2,9]$. Hence, $P_k(A)$ chooses an interval (a,b), where $b \geq a$ and $a,b \in [2,9]$ so that:

$$H(P_k(A)) \leq H(P_j(A)), \text{ for } \forall j = a, \ldots, b.$$

$P_k(A)$ creates k binary attributes, one for each A_j^s, $j = 2, \ldots, k$. This procedure is extended to all attributes. The obtained binary attributes slice the space. The binary attributes are used for obtaining a piecewise polynomial model.

The last two extensions of New CIPER include inducing a polynomial equation for prediction of several variables, and introducing of classification by using the induced polynomial structure. These were not of interest here.

CIPER induces the following polynomial on the running example:

$$\hat{C} = 0.2903A_1 + 0.0461A_2 + 0.1515A_1A_2 + 0.6216. \tag{4.13}$$

The results from the polynomial (4.13) are given in the last column in Table 4.4 and the regression curve is presented in Figure 4.3.

Table 4.4: Ranking obtained with CIPER for the running example

No.	$\mathbf{A_1}$	$\mathbf{A_2}$	C	CIPER
1	1	1	1	1.1095
2	2	1	1	1.5514
3	1	2	1	1.3072
4	1	3	1	1.5048
5	3	1	2	1.9932
6	2	2	2	1.9006
7	3	2	3	2.4940
8	2	3	3	2.2498
9	3	3	3	2.9948

Table 4.5: Ranking obtained with New CIPER for the running example

No.	$\mathbf{A_1}$	$\mathbf{A_2}$	C	New CIPER
1	1	1	1	0.9734
2	2	1	1	1.6106
3	1	2	1	1.0630
4	1	3	1	1.1527
5	3	1	2	2.2106
6	2	2	2	1.8961
7	3	2	3	2.6549
8	2	3	3	2.1816
9	3	3	3	3.0992

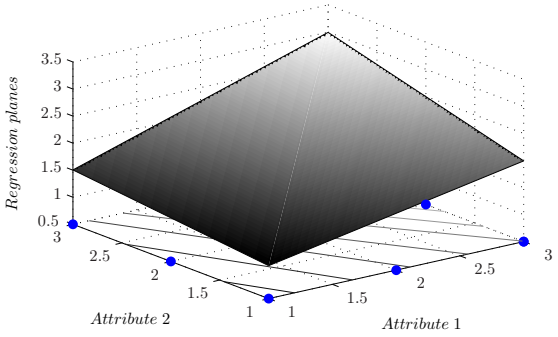


Figure 4.3: Regression curve obtained with CIPER for the running example

By applying New CIPER on the running example, the following polynomial is obtained.

$$\hat{C} = 0.4412A_1 - 0.1434A_2 + 0.2516A_1A_2 - 0.0186A_1^2A_2 + 0.4426. \tag{4.14}$$

The results from the polynomial (4.14) are given in the last column in Table 4.5, and the regression function is presented in Figure 4.4.

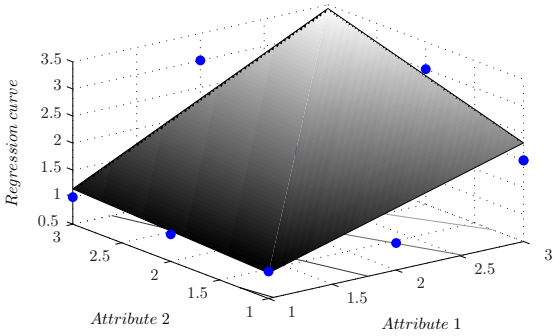


Figure 4.4: Regression curve obtained with New CIPER for the running example

4.3 Reformulation of the Problem as Optimization Problem with Constraints

In this section, the ranking problem is reformulated as optimization problem with constraints. The objective is to find a weight vector x, which would minimize the distances between the ranking values in-between two consecutive classes. The constrains are given so that ranking values are always in the intervals $c \pm 0.5$. These constrains eliminate the need of the third stage of QQ. The objective function is linear, hence a linear programming is used to find the solution of the problem.

The general form of the linear programming model is given with:

$$\begin{aligned} &\max f(\omega) \\ \text{subject to } &g_i(\omega) = l_i, \text{ for } i = 1, \cdots, n \text{ Equality constraints} \\ &h_j(\omega) \leq d_j, \text{ for } j = 1, \cdots, m \text{ Inequality constraints} \\ &\omega_i \geq 0 \end{aligned}$$

where ω is a vector of decision variables residing in a n-dimensional space, or in this case weights of the linear objective function, $f(\omega)$ is the objective function, and $g_i(\omega)$ and $h_j(\omega)$ are constraint functions that need to be satisfied, and l_i and d_j are constants (Hillier and Lieberman, 2001). The model may also get one of the following forms:

1. The objective function may solve the dual problem of minimization, $\min f(\omega)$, instead of the primal problem of maximization;
2. Additional inequality constraints with 'greater-than-or-equal' sign: $h_k(\omega) \geq p_k$ for $k = 1, \cdots, r$;
3. Deleting the non-negativity constraint for some of the decision variables ω_i.

The reformulation of the problem of option ranking to an optimization problem is demonstrated on the running example given in Table 3.6. The Table 3.6 consists of two quantitative attributes and a class. The constraints are defined directly from Table 3.6 as follows. Each row, which may be written in the form of $\omega_1 A_1 + \omega_2 A_2$, has to receive an evaluation in the interval $c_i \pm 0.5$. This is written with two constraints:

$$\omega_1 A_1 + \omega_2 A_2 \geq c_i - 0.5 \text{ and } \omega_1 A_1 + \omega_2 A_2 \leq c_i + 0.5$$

The constraints using this formulation are given with (4.15). Next, we have to define the optimization function.

$$\begin{aligned}
\omega_1+\omega_2 &\geq 0.5,\\
\omega_1+\omega_2 &\leq 1.5\\
2\omega_1+\omega_2 &\geq 0.5,\\
2\omega_1+\omega_2 &\leq 1.5\\
\omega_1+2\omega_2 &\geq 0.5,\\
\omega_1+\omega_2 &\leq 1.5\\
\omega_1+3\omega_2 &\geq 0.5,\\
\omega_1+3\omega_2 &\leq 1.5\\
3\omega_1+\omega_2 &\geq 1.5,\\
3\omega_1+\omega_2 &\leq 2.5\\
2\omega_1+2\omega_2 &\geq 1.5,\\
2\omega_1+2\omega_2 &\leq 2.5\\
3\omega_1+2\omega_2 &\geq 2.5,\\
3\omega_1+2\omega_2 &\leq 3.5\\
2\omega_1+3\omega_2 &\geq 2.5,\\
2\omega_1+3\omega_2 &\leq 3.5\\
3\omega_1+3\omega_2 &\geq 2.5,\\
3\omega_1+3\omega_2 &\leq 3.5.
\end{aligned} \tag{4.15}$$

There are two ways to define the optimization function for the running example. The first one is to minimize the distances between border classes and the second one is to maximize the distances between border classes. The maximization of the distances of two neighbouring classes would lead to small differences among options in the class, which is in contrast to the goal of this thesis. Hence the minimization approach is used, in order to obtain rankings of options with maximum spread into the intervals $c_i \pm 0.5$ where $c_i \in C$.

The goal is to find the values of ω_1 and ω_2 that would minimize the distances between the consecutive classes. For the running example, we would like to find a function that would rank all options so that it minimizes the sum of the distances between the highest option in one class c_i and the lowest in the next higher class c_{i+1}.

When defining the objective function, the rankings of the options are unknown, and consequently the highest and lowest evaluations of options in classes are also unknown. To solve this problem, we define all possible objective functions over the given decision table, and then we search among them for solution of the defined problem. For example, let us choose that the maximal ranking in the first class of the running example is the option number 4, and the minimal ranking in the next higher class, the second class, receives option 6. Next, let option 5 receives the maximal ranking in the second class and let us choose that option 7 receives the minimal ranking in the third class. Then the objective function that is:

$$min\ (2\omega_1+2\omega_2)-(\omega_1+3\omega_2)+(3\omega_1+2\omega_2)-(3\omega_1+\omega_2)=min\ (\omega_1+2\omega_2).$$

Writing down all possible combinations would lead to the following objective functions:

$$\begin{aligned}
&min\ 3\omega_1 + \omega_2,\\
&min\ 3\omega_1\\
&min\ 3\omega_1 - \omega_2,\\
&min\ 3\omega_1 - 2\omega_2\\
&min\ 2\omega_1 + 2\omega_2,\\
&min\ 2\omega_1 + \omega_2\\
&min\ 2\omega_1 - \omega_2,\\
&min\ 2\omega_1\\
&min\ \omega_1 + 3\omega_2,\\
&min\ \omega_1 + 2\omega_2\\
&min\ \omega_1 + \omega_2,\\
&min\ \omega_1\\
&min\ 3\omega_2,\\
&min\ 2\omega_2\\
&min\ \omega_2,\\
&min\ -\omega_1 + 3\omega_2
\end{aligned} \tag{4.16}$$

The set of all possible objective functions may be decreased by deleting the functions which are linear combination of some other function in the set. For example, instead of having two functions $min\ 3\omega_1$ and $min\ \omega_1$, we may delete the first one and use only the second one in the set of possible objective functions.

Table 4.6: Ranking of options with constraint optimization

No.	A_1	A_2	C	Evaluation
1	3	3	3	3.0000
2	3	2	3	2.5001
3	3	1	2	2.0002
4	2	3	3	2.4999
5	2	2	2	2.0000
6	2	1	1	1.5001
7	1	3	1	1.9998
8	1	2	1	1.4999
9	1	1	1	1.0000

The next step is to find a solution of the problem defined with any of the objective functions. For that reason, the implementation of the Simplex algorithm in MATLAB was used. The Simplex algorithm did not find a solution for any of the defined objective functions that satisfy the constraints 4.15.

For example, the obtained solution from applying the first objective function given in (4.16), $min\ 3\omega_1 + \omega_2$, is provided in Table 4.6.

This solution is proposed by the Simplex solver algorithm in MATLAB as the closest one to the constraint region of solutions defined by the problem. It does not satisfy the constrains (4.15). For instance, the evaluation of option 7 belongs to the interval 2 ± 0.5, instead of the required interval of 1 ± 0.5. The example shows that the main difficulty of this approach are the stringent constraints that rarely lead to a solution in the constraint space.

4.4 Implementation of the QQ-Based Algorithms

QQ and its modifications with impurity functions are implemented in MATLAB for a single or hierarchical setting of decision tables. The optimization approach uses the implemented function for linear programming in MATLAB, however a set of functions were developed in order to: automatically define all constrains and all possible objective functions for a given decision table, and to search among the objective functions for a solution.

Calculations for CIPER and New CIPER were performed by using the developed tools provided by their authors [1]. Shell commands were used for iterative calculations of all decision tables covered in the three sets in Chapter 7. Analysis was performed in MATLAB afterwards for checking the monotonicity, full ranking and consistency of the results.

[1] CIPER is also available at http://kt.ijs.si/software/ciper/.

5 Copula Theory

The main focus of this thesis is the usage of copula-based functions as estimators of the regression function for option ranking. To be able to use copulas, attributes in the decision tables are considered as random variables described with their marginal distributions. Copulas are functions which connect the marginal distributions of the random variables and their joint distribution, which is used for regression. In particular, the thesis exploits bi-variate copulas and extend them to multi-variate ones. This is of essential interest when working with multi-attribute qualitative models. Bi-variate copulas provide the dependencies between two random variables, while multi-variate ones provide the dependancies among all random variables at hand. To form a multi-variate copula, the bi-variate ones are merged forming a hierarchical copula construction. This chapter presents the basic concepts of probability theory and describes all steps required for building a hierarchical copula.

For the obtained hierarchical copula constructions, the thesis presents new quantile regression equations which are given in Chapter 6.

5.1 Basic Concepts of the Probability Theory

Random variables may be discrete or continuous depending on the values they receive. A random variable X is discrete, if its set of possible values $\boldsymbol{\mathcal{X}}$ is finite, or at mostly, countably infinite. It is uniquely defined by:

- the set of possible values $\boldsymbol{\mathcal{X}}$,
- its cumulative distribution function (CDF) or probability density function (PDF).

In continuation, several definitions and theorems are given in order to provide step-by-step formalization of the theory behind copulas.

Definition 5.1 (Leemis and Park (2005))**.** *A probability density function for a discrete random variable X is a real-valued function $f(\cdot)$ defined for each possible $x \in X$ as the probability that X has the value x*

$$f(x) = Pr(X = x).$$

In addition, $f(\cdot)$ is defined so that

$$\sum_{x \in \boldsymbol{\mathcal{X}}} f(x) = 1.$$

Definition 5.2 (Leemis and Park (2005))**.** *A probability density function for the continuos random variable X is a real-valued function $f(\cdot)$ defined for each $x \in X$ as*

$$\int_a^b f(x)dx = Pr(a \leq X \leq b),$$

for any interval $(a,b) \in \boldsymbol{\mathcal{X}}$. In addition, $f(\cdot)$ is defined so that

$$\int_{x \in \boldsymbol{\mathcal{X}}} f(x)dx = 1.$$

Definition 5.3. *(Leemis and Park, 2005) A cumulative distribution function of the discrete random variable X is the real-valued function $F(\cdot)$ defined for each $x \in X$ as*

$$F(x) = Pr(X \leq x) = \sum_{t \leq x} f(t)$$

where the sum is over all $t \in \mathcal{X}$, for which $t \leq x$.

Definition 5.4. *(Leemis and Park, 2005) A cumulative distribution function of the continuos random variable X is the continuos real-valued function $F(\cdot)$ defined for each $x \in \mathcal{R}$ as*

$$F(x) = Pr(X \leq x) = \int_{t \leq x} f(t)dt.$$

The CDF has the following properties:

- it is strictly increasing: if $x_1 < x_2$, then $F(x_1) < F(x_2)$,
- CDF values belong to the interval $[0,1]$.

Definition 5.5. *(Leemis and Park, 2005) Let X be a discrete random variable with CDF $F(\cdot)$. The inverse distribution function (idf) of X is the function $F^{-1} : (0,1) \to \mathcal{X}$ defined for all $u \in (0,1)$ as*

$$F^{-1}(u) = min_x\{x : u < F(x)\}$$

where the minimum is over all posible values $x \in X$.

An example of idf $F^{-1}(u)$ of the discrete random variable x is given in Figure 5.1.

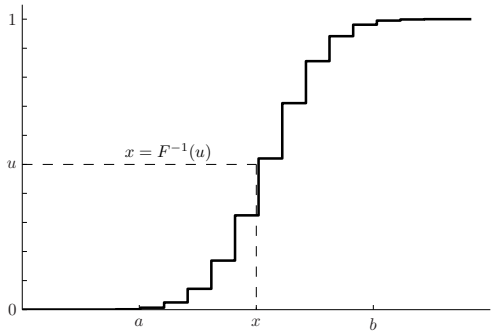


Figure 5.1: Inverse distribution function $F^{-1}(u)$

Theorem 5.1. *(Leemis and Park, 2005) Let X be an integer-valued random variable with $\mathcal{X} = \{a, a+1, ..., b\}$ where b may be ∞ and let $F(\cdot)$ be the CDF of X. For any $u \in (0,1)$, if $u < F(a)$ then $F^{-1}(u) = a$, else $F^{-1}(u) = x$ where $x \in \mathcal{X}$ is the (unique) possible value of X for which $F(x-1) \leq u < F(x)$.*

Theorem 5.2. *(Probability integral transformation) If X is a discrete (continuous) random variable with idf $F^{-1}(\cdot)$ and the continuos random variable U is $\mathcal{U}(0,1)$ and Z is the discrete (continuous) random variable defined by $Z = F^{-1}(U)$ then Z and X are identically distributed.*

Here $\mathcal{U}(0,1)$ stands for uniform distribution on the interval $[0,1]$. An example of the theorem 5.2 is given in Figure 5.2. In Figure 5.2, the random variable X and the random variable Z obtained with the probability integral transformation of a random variable U with uniform distribution, share the same normal distribution $\mathcal{N}(\mu, \sigma)$.

Multi-criteria decision models deal with more than one attribute, therefore, we need to define the relations that may exist between different attributes. In probability theory, the relation between random variables is determined with their joint distribution function.

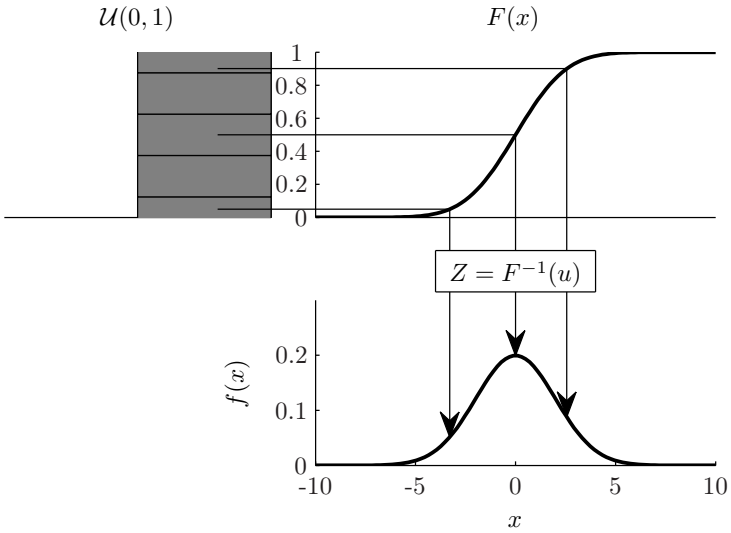


Figure 5.2: Probability integral transform of a random variable X with PDF $f(x)$ and CDF $F(x)$

Definition 5.6. *A joint (two-dimensional) distribution function $H(x,y)$ of two random variables X and Y is given with $H(x,y) = Pr(X \leq x, Y \leq y)$.*

The function $H(x,y)$ maps $H : [0,1] \times [0,1] \to [0,1]$, and it has the following properties:

- $H(\infty,\infty) = 1$ and $H(-\infty,y) = H(x,-\infty) = H(-\infty,-\infty) = 0$
- $H(x_1,y_1) - H(x_1,y_2) - H(x_2,y_1) + H(x_2,y_2) \geq 0, \; \forall \, x_1 \leq x_2, y_1 \leq y_2.$

5.2 Quantile Regression

Quantile regression is a tool for performing statistical regression when there is vague or no knowledge of the relationship among the random variables. It models the relation between specific quantiles of the regression variable and a set of independent variables. The difference between the linear and quantile regression is that the linear regression coefficients represent the increase (decrease) of the regression variable produced by increase (decrease) in the independent variables associated with their coefficients. The quantile regression estimates the change in a specified quantile of the regression variable produced by a change in the independent variables (Despa, 2007).

Definition 5.7. *A quantile $q \in [0,1]$ is a value which divides the distribution $F(x)$ so that there is a given proportion of observations below the quantile:*

$$F^{-1}(q) = inf\{x : F(x) \geq q\}.$$

In case when the quantile in the regression equation is $q = 0.5$, a median is obtained. The median represents a central value of the distribution, so that half of the data points are less than or equal to it and half are greater than or equal to it. The advantage of using median regression as a measure of centre, compared to the mean regression that is obtained when using the least-square algorithm, is due to its ability to resist the strong effect of outliers (Walters et al., 2006).

The need of joint distribution functions for performing regression To define the regression function (3.2), a copula-based approach is used. When using copulas, attributes A_i are regarded as random variables. When attributes and options are given in a decision table, we have at hand a sample of joint behavior of the attributes for which we can also estimate their marginal distributions $F_i(A_i)$. For determining the dependences among the random variables, we need to find their joint distribution $F(A_1,\ldots,A_n,A_{agg})$ in the unseen space of options. In order to estimate the multi-variate joint densities with a constant estimation accuracy, the required sample size rapidly increases with the number of dimensions (Silverman, 1986). As we deal with small sample sizes, we adopt the copula approach for estimation of the joint density and distribution. Having defined the joint distribution function and the marginals of each random variable, we may proceed with the regression task in the probability space.

5.3 Copula Functions

By definition, for two random variables X and Y, which are defined on the same probability space, the joint distribution function $H(x,y)$ defines the probability of events defined in terms of both X and Y. Finding the joint distribution function in an explicit form is a difficult task. Sklar (1973, 1996) proved that the joint distribution function $H(x,y)$ of two random variables is equal to the copula $C(u,v)$.

Theorem 5.3. *(Sklar's theorem) Let H be a two-dimensional distribution function with marginal distribution functions F_{X_1} and F_{X_2}. Then a copula $C(u,v)$ exists that for all $x_1,x_2 \in \mathcal{R}^2$,*

$$H(x_1,x_2) = C(u,v) \tag{5.1}$$

where $u = F_{X_1}(x_1)$ and $v = F_{X_2}(x_2)$. Moreover, if F_{X_1} and F_{X_2} are continuos, then $C(u,v)$ is unique. Otherwise $C(u,v)$ is uniquely determined on the Cartesian product Range(F_{X_1}) $\times$ Range(F_{X_2}). Conversely, if $C(u,v)$ is a copula and F_{X_1} and F_{X_2} are distribution functions then H is a two–dimensional distribution function with marginals F_{X_1} and F_{X_2}.

From Theorem 5.3 it follows that, in order to use (5.1), the theorem 5.2 is fundamental for usage of copulas. For example, let consider a vector of two random variables (X_1,X_2), each one with marginal cdfs $F_1(X_1) = Pr(X_1 \leq x_1)$ and $F_2(X_2) = Pr(X_2 \leq x_2)$. Applying theorem 5.2 to each of the components of the vector, leads to $(U_1,U_2) = (F_1(X_1),F_2(X_2))$. Finally, the copula of (X_1,X_2) is defined as the joint cumulative distribution function of (U_1,U_2):

$$C(u_1,u_2) = Pr(U_1 \leq u_1, U_2 \leq u_2).$$

In the thesis, we use copulas as aggregation functions, which map from the unit m-interval $[0,1]^m$ to the unit interval [0,1].

Copulas are bound with the Fréchet-Hoeffding bounds (Durante and Sempi, 2010):

$$\max(u+v-1,0) \leqslant C(u,v) \leqslant \min(u,v), \forall u,v \in [0,1] \tag{5.2}$$

where

$$u = F_1(X_1),\ u \sim \mathcal{U}(0,1),$$
$$v = F_2(X_2),\ v \sim \mathcal{U}(0,1),$$

and

$$M(u,v) \equiv \max(u,v),$$
$$W(u,v) \equiv \min(u,v)$$

where $\mathcal{U}(0,1)$ is uniform distribution on the interval $[0,1]$, and $M(u,v)$ and $W(u,v)$ are called the maximum and the minimum copula, respectively. Thus copulas lie between these two extremes. They have three specific properties:

Property 1 $C(1,v) = C(v,1) = v,\ \forall v \in [0,1]$

Property 2 $C(u,v) = 0,$ if $u = 0$ and/or $v = 0$

Property 3 $C(u_1,v_1) - C(u_1,v_2) - C(u_2,v_1) + C(u_2,v_2) \geq 0$ holds whenever $u_1 \geq u_2$ and $v_1 \geq v_2$.

The first property says that if we know that the marginal probability of one of the variables is one, then the joint probability is the same as the probability of the other variable. The second property says that the joint probability is zero if the marginal probability of any variable is zero. The third property, known as 2-increasing property, says that the value of copula function is always non-negative. The last property is of special interest. Therefore, its derivation is given below, by using the following cases:

1. Let us fix the first argument to u_1, and let $v_1 \geq v_2$. In this case one obtains

$$C(u_1,v_1) - C(u_1,v_2) \geq 0. \tag{5.3}$$

   Decreasing the fixed argument to a smaller value $u_2 \leq u_1$, while still holds that $v_1 \geq v_2$, results in:

$$C(u_2,v_1) - C(u_2,v_2) \geq 0 \tag{5.4}$$

   Finally, the difference between (5.3) and (5.4) leads to:

$$C(u_1,v_1) - C(u_1,v_2) - C(u_2,v_1) + C(u_2,v_2) \geq 0$$

   whenever $u_1 \geq u_2$ and $v_1 \geq v_2$.

2. For the opposite case let us fix the first argument to u_1, and $v_1 \leq v_2$. In this case one obtains

$$C(u_1,v_1) - C(u_1,v_2) \leq 0 \tag{5.5}$$

   Decreasing the first argument to a smaller value $u_2 \leq u_1$, while still holds that $v_1 \leq v_2$, results in:

$$C(u_2,v_1) - C(u_2,v_2) \leq 0 \tag{5.6}$$

   Finally, the difference between (5.5) and (5.6) leads to:

$$C(u_1,v_1) - C(u_1,v_2) - C(u_2,v_1) + C(u_2,v_2) \geq 0$$

   whenever $u_1 \leq u_2$ and $v_1 \leq v_2$.

The 2–increasing property does not require $C(u,v)$ to be actually increasing in either argument as the "increasingness" can be negative as well. For example, if $C(u,v) = au + bv$ where a and b are any constants, then $C(u,v)$ is 2-increasing, since the inequality (5.3) holds. So we may fix u, increase v and $C(u,v)$ may decrease, but the inequality (5.3) will still hold. If $C(u,v)$ is increasing in each argument then $C(u,v)$ is 2-increasing. The opposite does not hold.

The three properties of copulas ensure that they can be used as functions that link a multidimensional distribution to its one-dimensional margins. Therefore they are building blocks for models for construction of multi-variate dependence (Berg and Aas, 2009).

5.3.1 Archimedean Copulas and Connection to T-norms

If two random variables x and y are independent, then their joint distribution function is $H(x,y) = F(x)G(y)$. In this case, $H(x,y) = \Pi(u,v) = uv$, where $\Pi(u,v)$ is called the product copula. $M(u,v)$, $W(u,v)$ and $\Pi(u,v)$ are the three most important copulas, as well as the three most important t-norms. If two random variables are dependent, one may write their joint distribution function as a sum of functions of its marginal distributions:

$$\varphi(H(x,y)) = \varphi(F(x)) + \varphi(G(y)), \tag{5.7}$$

where $\varphi(\cdot)$ is called a constructor function (Nelsen, 2006). Depending on the form of $\varphi(\cdot)$, one may elicit different kinds of copulas. In order for a function $\varphi : [0,1] \to [0,\infty]$ to be a constructor function, it must be:

- continuous,
- strictly decreasing from $I = [0,1]$ to $[0,\infty]$ and
- $\varphi(1) = 0$.

The inverse function of φ is φ^{-1}, defined in the interval $0 \leqslant t \leqslant \varphi(0)$. If $\varphi(0) < \infty$, then the inverse function is called pseudo-inverse $\varphi^{[-1]}$. In terms of copulas (5.7) reads:

$$\varphi(C(u,v)) = \varphi(u) + \varphi(v),$$

which leads to:

$$C(u,v) = \varphi^{[-1]}(\varphi(u) + \varphi(v)). \tag{5.8}$$

Copulas obtained using (5.8) are called Archimedean copulas (Joe, 1997). The usage of Archimedean copulas is mainly motivated by the following three properties:

- symmetry: $C(u_1,u_2) = C(u_2,u_1)$ for all $u_1,u_2 \in [0,1]$;
- associativity: $C(C(u_1,u_2),u_3) = C(u_1,C(u_2,u_3))$, for all $u_1,u_2,u_3 \in [0,1]$;
- if φ is the generator, then $c\varphi$ is also generator for $c > 0$.

This dissertation focuses on Frank, Clayton and Gumbel copulas which belong to the family of Archimedean copulas. These three copulas and their generator functions are given in Table 5.1. The median regression curve v based on the different Archimedean copulas is given in the last column of Table 5.1. Details of estimation of the parameter θ are given in Section 5.3.2.

Table 5.1: Different Archimedean copulas, their generator functions φ, borders of θ parameter and value of regression variable v

	$C_\theta(u,v)$	$\varphi_\theta(t)$	θ	$Solve(\frac{\partial C_\theta(u,v)}{\partial u} = q, v)$
Clayton	$\left[\max\left(u^{-\theta}+v^{-\theta}-1,0\right)\right]^{-1/\theta}$	$\frac{1}{\theta}\left(t^{-\theta}-1\right)$	$[-1,\infty)\setminus\{0\}$	$(1-u^{-\theta}+(qu^{1+\theta})^{-\frac{\theta}{1+\theta}})^{-\frac{1}{\theta}}$
Frank	$-\frac{1}{\theta}\ln\left(1+\frac{(e^{-\theta u}-1)(e^{-\theta v}-1)}{e^{-\theta}-1}\right)$	$-\ln\frac{e^{-\theta t}-1}{e^{-\theta}-1}$	$(-\infty,\infty)\setminus\{0\}$	$\frac{1}{\theta}\log\frac{-e^{\theta}(1-q+qe^{\theta u})}{-e^{\theta}+qe^{\theta}-qe^{\theta u}}$
Gumbel-Hougaard	$\exp\left(-\left[(\ln u)^{\theta}+(-\ln v)^{\theta}\right]^{1/\theta}\right)$	$\ln\frac{1-\theta(1-t)}{t}$	$[1,\infty)$	*only numerical solution*

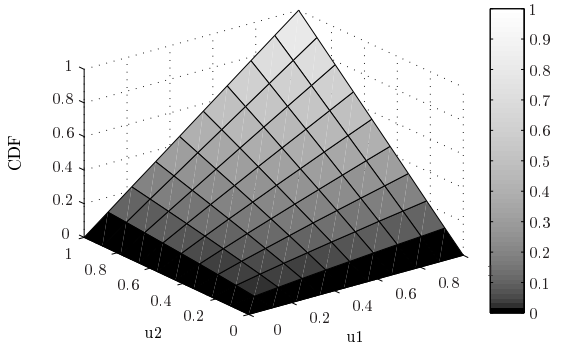


Figure 5.3: Clayton copula distribution function $C(u_1, u_2)$

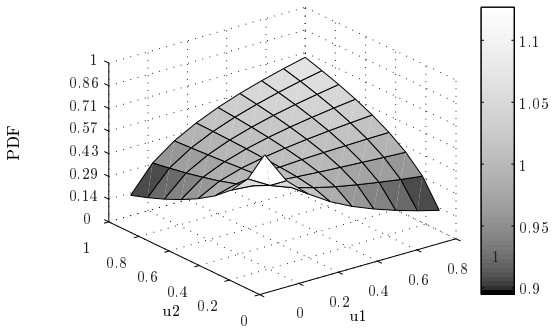


Figure 5.4: Clayton copula density function $c(u_1, u_2)$

5.3.2 Estimation of the Parameter $\hat{\theta}$

The PDF and CDF are connected with the derivative (integral) relation. In terms of copulas, the joint CDF is $H(x,y) = C(u,v)$, hence to obtain the joint PDF one uses the relation:

$$c(u,v) = \frac{\partial^2 C(u,v)}{\partial u \partial v}.$$

As an example, one may consider the Clayton copula. Starting from its CDF, one gets the density function as:

$$c(u,v) = (1+\theta)\,(uv)^{-(\theta+1)}(u^{-\theta}+v^{-\theta}-1)^{-\frac{1+2\theta}{\theta}}$$

The rearrangment of the last equation leads to:

$$c(u,v) = (1+\theta)(uv)^{-(\theta+1)}(u^{-\theta}+v^{-\theta}-1)^{-\frac{1+2\theta}{\theta}} \tag{5.9}$$

The only free parameter when using the Archimedean copulas is θ. One way to find θ in (5.9) is to use the maximum likelihood estimation. By definition, the likelihood function of a random sample of size n is the joint probability density (mass) function denoted by (Martinez and Martinez, 2002):

$$L(\theta) = L(\theta; x_1, x_2, ..., x_n) = f(x_1, x_2, ..., x_n; \theta) \tag{5.10}$$

where θ is a vector of parameters. In order to estimate $\hat{\theta}$ that maximizes the likelihood function (5.10), we take the derivative of $L(\theta)$ with respect to θ and set it equal to zero:

$$\frac{\partial L(\theta)}{\partial \theta} = 0$$

The function $log(L(\theta))$ varies monotonically with its argument, or in other words, $log(L(\theta))$ increases and decreases when $L(\theta)$ increases and decreases respectively. This is because both functions are monotonically related to each other, leading to the same maximum estimate for both functions (Myung, 2003). In other words, the maxima of the likelihood function and the maxima of the logarithm of the likelihood function are the same. We use this fact in cases when it is easier to find the maxima of a logarithm likelihood function, such as when exponents are involved in the density function, as it is in our case. The logarithm of the density function $c(u,v)$ is:

$$\log c(u,v) = \log(1+\theta) - (\theta+1)\log(uv) - \frac{1+2\theta}{\theta}\log(u^{-\theta}+v^{-\theta}-1)$$

Finally, $\hat{\theta}$ is determined as the minimum negative log likelihood (Brent, 1993; Forsythe et al., 1976). The same procedure may be applied to all copulas that belong to the Archimedean family.

5.3.3 Interpretation of the Parameter θ

Copulas can help us in studying the dependence association between random variables. For example, one may use the well-known Kendall's τ measure of association, which measures the concordance between random variables. The population version for Kendall's τ is

$$\tau_\theta = 1 + 4\int_0^1 \frac{\varphi(t)}{\varphi(t)'}.$$

An example, of Kendall's tau τ_θ, for different Archimedean copulas is given in Table 5.2. From Table 5.2

Table 5.2: Kendall's tau (Nelsen, 2006)

Copula	Kendall's tau
Gumbel	$\frac{\theta-1}{\theta}$
Clayton	$\frac{\theta}{\theta+2}$
Frank	$\frac{1-\theta}{4}[1-D_1(\theta)], D_1(\theta) = \frac{1}{\theta}\int_0^\theta \frac{t}{e^t-1}dt$

it is clear that the higher values of θ mean higher dependence association between the two random variables.

Another measure of dependence is the tail dependence. It is a concept which relates the amount of dependence in the upper-right quadrant tail or lower-left quadrant tail of a bi-variate distribution. It is a useful concept for studying dependence between extreme values. The Gumbel copula has upper tail dependence λ_u given with Rachev (2003):

$$\lambda_u = 2 - 2^{\frac{1}{\theta}}.$$

Clayton copula does not have upper tail dependence, however it has a lower tail dependence λ_L given with Rachev (2003):

$$\lambda_L = 2^{-\frac{1}{\theta}}.$$

The Frank family of copulas does not have upper nor lower tail dependence.

5.4 Higher-Dimensional Copulas

Quantitative models generally consist of more than two attributes. In order to use the described bi-variate copulas in multi attribute models, one needs to extend them to higher dimensions. There are two approaches of extending to higher dimensional copulas (Fischer et al., 2009; Savu and Trade, 2006):

1. The first approach extends the bi-variate copula to multi-variate copula using only one dependence parameter θ to specify the dependences among n random variables. Such copulas are known as exchangeable Archimedean copulas (Berg and Aas, 2009; Hofert, 2008). In this case the copula functions and the joint distribution function are related with the relation:

   $$H(x_1,\ldots,x_n) = C(u_1,\ldots,u_n;\theta).$$

   Their main drawback is that copula densities for dimensions higher than two are tedious to derive (Trivedi and David, 2006);

2. The second approach uses bi-variate copulas to form a hierarchical structure with at most $n-1$ dependence parameters θ_i for n random variables. There are two possibilities to build such hierarchical structures (Berg and Aas, 2009): the FNAC, as shown in Figure 5.5, and the PNAC, as shown in Figure 5.7.

For a given decision table, the described two approaches can be used as follows:

1. Symmetric decision tables are aggregated with the first approach, which uses only one dependence parameter θ. This approach ensures the symmetry in the evaluations of options.

2. Partially symmetric decision tables are aggregated in two-step procedure. In the first step, each group of symmetric attributes is aggregated with one parametric copula. In the second step, the non-symmetric attributes and the formed copulas are aggregated either with one parametric or with multi parametric copula.

3. Non-symmetric decision tables are aggregated either with one parametric or with multi parametric copula (FNACs or PNACs).

5.4.1 One-Parametric Archimedean Multi-Variate Copulas

Multivariate Archimedean copulas are given with:

$$C(u_1,u_2,\ldots,u_d) = \varphi^{[-1]}(\varphi(u_1)+\varphi(u_2)+\ldots+\varphi(u_d), \tag{5.11}$$

where φ is a generator function with single parameter θ. In the 3-dimensional case, (5.11) is given with:

$$C(u_1,u_2,u_3) = \varphi^{[-1]}(\varphi\circ\varphi^{[-1]}(\varphi(u_1)+\varphi(u_2))+\varphi(u_3)) = C(C(u_1,u_2),u_3).$$

The simbol $\circ$ stands for composite functions. It means that the argument of the function standing before the simbol $\circ$ is the outcome of the function after the simbol $\circ$.

Similarly, one may show that:

$$C(u_1,u_2,u_3) = C(u_1,C(u_2,u_3)) = C(C(u_1,u_2),u_3) = C(C(u_1,u_3),u_2).$$

In other words, the one-parametric Archimedean copula may be written as a structure of nested bi-variate copulas with the same value of the parameter θ.

To estimate θ in (5.11), we use the maximum-likelihood approach. For that purpose one needs to calculate the corresponding probability density $c(u_1,u_2,\ldots,u_d)$. The general form of the copula density expressed through the generator function is:

$$c(u_1,u_2,\ldots,u_d) = \varphi^{-1(d)}(\varphi(u_1)+\varphi(u_2)+\ldots+\varphi(u_d))\prod_{i=1}^{d}\varphi'(u_i)$$

where $\varphi^{-1(d)}$ denotes the d-th derivative of the inverse generator function.

The canonical loglikelihood function is

$$\ln L(\theta;u_{it},i=1,\ldots,d,t=1,\ldots,T) = \sum_{t=1}^{T}\ln c_L(u_{1t},\ldots,u_{dt};\theta) \tag{5.12}$$

where d is number of attributes, t is number of observations (options), and the canonical maximum likelihood estimator of θ is

$$\hat{\theta} = \operatorname{argmax}\ln L(\theta). \tag{5.13}$$

5.4.2 Specific Derivation for Multi-Variate Clayton Copula

The generation function φ and its inverse φ^{-1} are given with:

$$\varphi(t) = \frac{1}{\theta}(t^{-\theta} - 1).$$

$$\varphi^{-1}(x) = (1 + x\theta)^{\frac{-1}{\theta}}.$$

The derivative of the inverse generator of order d is:

$$\varphi^{-1(d)}(x) = (-1)^i (1+\theta x)^{-\frac{1+d\theta}{\theta}} \prod_{j=0}^{i-1}(1+j\theta).$$

With these equations, the loglikelihood (5.12) reads:

$$\ln L(\theta; u_{it}, i = 1,\ldots,d, t = 1,\ldots,T) = \sum_{t=1}^{T}\left(\sum_{i=0}^{d-1}\log(1+i\theta) - (1+\theta)\sum_{i=1}^{d}\log(u_{it}) + (1+2\theta)\left(-\frac{1}{\theta}\log(1-d+\sum_{i=1}^{d}u_{it}^{-\theta})\right)\right). \tag{5.14}$$

Solving (5.13) for (5.14) can be calculated numerically in Matlab. The specific derivations for Frank and Gumbel copulas are given in Appendix C.

5.4.3 Fully Nested Archimedean Constructions

FNAC structure is a tree-like structure, as shown in Figure 5.5, where the basic element represents the bi-variate copula. The first two nodes u_1 and u_2 are coupled into copula $C_1(u_1,u_2)$ with θ_1. In the next step C_1 is coupled with u_3 into $C_2(C_1(u_1,u_2),u_3)$ with θ_2, and so on. Hence, the leafs in the FNAC represent the values of the marginal distributions of the attributes. The final output of the topmost copula results in:

$$C(u_1,u_2,u_3,u_4,u_5) = C_4(u_5,C_3(u_4,C_2(u_3,C_1(u_1,u_2)))). \tag{5.15}$$

In order for (5.15) to represent a valid copula structure, the following conditions have to be fulfilled (Rachev, 2003):

$$\theta_i \leq \theta_{i-1} \leq \cdots \leq \theta_1 \tag{5.16}$$

where θ_1 is the most nested dependence parameter. In addition, FNAC allows different groupings of the marginal distributions of the variables. For example, if $n = 3$, the following FNACs are possible: $(u_3,[u_1,u_2;\theta_2];\theta_1)$, $(u_2,[u_1,u_3;\theta_2];\theta_1)$, $(u_1,[u_2,u_3;\theta_2];\theta_1)$, as shown in Figure 5.6. The requirement $\theta_2 > \theta_1$ follows from (5.16), and it must be fulfilled for the examples in Figure 5.6.

Table 5.3: Values of θ_i parameters obtained with FNACs for the example in Table 2.2

Method	Clayton	Gumbel	Frank
θ_1	0.6125	2.8053	8.3609
θ_2	0.1880	1.3945	1.8536

For the example given in Table 2.2, when using the Clayton, Frank and Gumbel bi-variate copula, the FNAC given in Figure 5.6b is obtained. The different values of θ_i are given in Table 5.3. For each of the three cases given in Table 5.3 holds: $\theta_1 \geq \theta_2$ which fulfill the nesting condition given with (5.16). Hence each of the copulas may be further on used for option ranking.

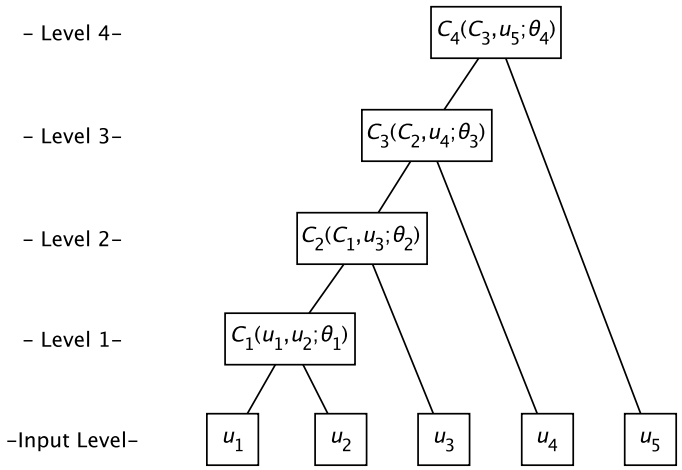
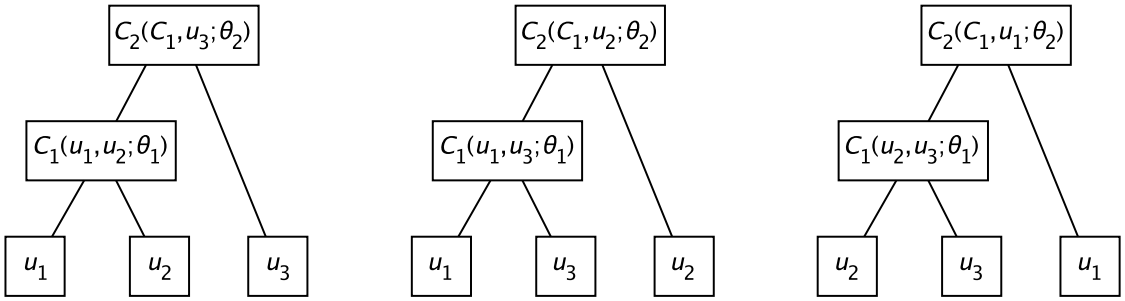


Figure 5.5: Fully nested Archimedean constructions of multi-dimensional copulas



(a) First permutation **(b)** Second permutation **(c)** Third permutation

Figure 5.6: Three permutations of the input variables

5.4.4 Partially Nested Archimedean Constructions

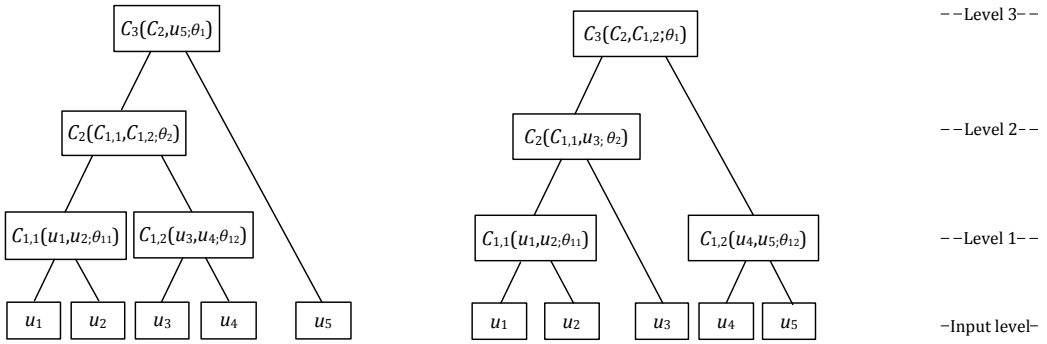
Another approach to obtain a hierarchical structure of Archimedean copulas is by plugging in bi-variate Archimedean copulas into each other, which leads to structures known as partially nested Archimedean copulas (PNAC). PNACs are more complex structures than FNACs and are found useful when it is not possible to build FNAC due to the constraints (5.16).

Two examples of PNACs for a decision problem with four attributes and an output class are presented in Figure 5.7. In order for the PNAC to represent a copula itself, the general condition is that parameters θ_i must decrease with the level of nesting, while there are no constraints on their values when two or more copulas are built on the same level (details are given in Section 5.4.5). Consequently, for the copula given in Figure 5.7a, the following nesting conditions must hold:

$$\theta_2 \leq \theta_{11},\ \theta_2 \leq \theta_{12},\ \theta_3 \leq \theta_2,$$

while regarding the copula given in Figure 5.7b the following must be fulfilled:

$$\theta_2 \leq \theta_{11},\ \theta_3 \leq \theta_{12},\ \theta_3 \leq \theta_2. \tag{5.17}$$



(a) First PNAC **(b)** Second PNAC

Figure 5.7: Partially nested Archimedean constructions with five attributes

5.4.5 Conditions for FNAC and PNAC

In this section, an explanation of the origin of equations (5.16) and (5.17) is provided. In order for C^n to represent a valid n - dimensional FNAC or PNAC, the inverse of φ in (5.7), the φ^{-1}, must be completely monotone on the interval $[0,\infty]$ as given in (Rachev, 2003, Theorem 6.8). In order for a function to be completely monotone on an interval, it must have derivatives of all orders which alternate in sign (Rachev, 2003, Definition 6.2). The inverse generator function for the Clayton copula is:

$$\varphi^{-1}(t) = (\theta t + 1)^{-1/\theta}$$

In each level $n \geq 2$, the argument into the inverse generator function φ_{i+1}^{-1} is the generator function φ_i from the previous level, leading to composite generator functions:

$$\varphi(t) = \varphi_i \circ \varphi_{i+1}^{-1}(t) = \frac{((\theta_{i+1}t+1)^{\frac{-1}{\theta_{i+1}}})^{-\theta_i} - 1}{\theta_i} \tag{5.18}$$

where i is the level in the FNAC or PNAC. In order to get a valid hierarchical copula, the composite functions must fulfill

$$(-1)^{j-1}(\varphi_i \circ \varphi_{i+1}^{-1})^{(j)}(t) \geq 0 \tag{5.19}$$

as described in (Rachev, 2003). Next, if we find the derivatives of (5.18) we get:

$$\varphi^{(1)}(t) = (\theta_{i+1}t+1)^{\frac{\theta_i-\theta_{i+1}}{\theta_{i+1}}}$$

$$\varphi^{(2)}(t) = (\theta_i-\theta_{i+1})(\theta_{i+1}t+1)^{\frac{\theta_i-2\theta_{i+1}}{\theta_{i+1}}}$$

$$\varphi^{(n)}(t) = (\theta_i-\theta_{i+1})(\theta_i-2\theta_{i+1})...(\theta_i-(n-1)\theta_{i+1})(\theta_{i+1}t+1)^{\frac{\theta_i-n\theta_{i+1}}{\theta_{i+1}}} \tag{5.20}$$

In order (5.20) to fulfill (5.19), the nesting condition (5.16) has to be fulfilled, which here receives the form: $\theta_{i+1} \geq \theta_i$.

The same may be applied for Gumbel (Rachev, 2003, Example 6.13) and Frank copulas.

Here we should note that the Fréchet-Hoefding inequality (5.2) bounds the values of the hierarchical copula in such a way that, whenever adding one more level in the FNAC or PNAC, the resulting copula will be smaller or in best case equal to the previous one. In other words, in each level of copula construction, the bounds of the resulting inverse cumulative function will become closer to each other and closer to the lower output class.

5.5 Kernel Smoothing of Attribute's Distributions in Hierarchical DEX Models

The copula-based aggregation function for option ranking is defined in the intervals given with the quantitative decision table. For example, the copula-based aggregation function for Table 6.5 leads to values in the interval $[1,3]$. These values are linearly mapped into expanded intervals defined by the third stage of QQ. The expanded interval for the example in Table 6.5 is $[0.5,3.5]$.

Next, the calculations of the class variables are propagated into the higher hierarchical levels, as input attributes. For example, the attributes **Cost** and **Safety** are input attributes for evaluation of the class attribute **Car**. If the FNAC obtained for evaluation of the attribute **Car** is built on the interval values for input attributes seen only in the Table 6.4, then the copula-based models would provide values only in these intervals. For example, the FNAC for **Car** will define its values in the interval $[1,3]$, while providing equal evaluation values of zero, or one in the intervals $[0.5,1)$ and $(3,3.5]$, respectively. To deal with this issue, one has to extend the distribution of the variables into the regions not covered with the observed data.

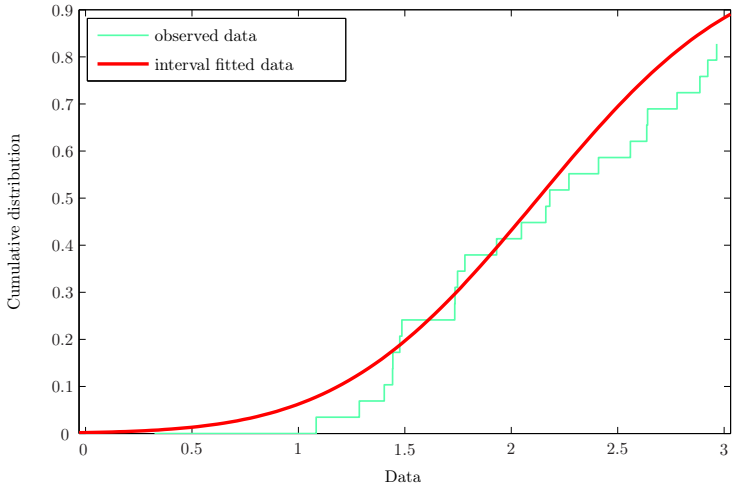


Figure 5.8: Smoothing of distribution function

The standard tool for displaying samples of data is via the empirical distribution function represented as a step function. An example is given with the green line in Figure 5.8. In addition, the inverse of the empirical distribution (the quantile) function and the histogram of the data are given with green lines in Figures 5.9 and 5.10, respectively. The advantage of this representation is that individual observations can be identified from the plot. In order to achieve improved estimation of attribute's values, we assume that the true underlying distribution function is smooth. To construct a smooth estimate, we place a kernel function, which is itself a distribution function, over each data point in the form (Bowman and Azzalini, 1997):

$$\hat{F}(y) = \frac{1}{n}\sum_{i=1}^{n} W(y - y_i; h).$$

Here the parameter h controls the standard deviation associated with the kernel function W, and hence controls the degree of smoothing applied to the data. Using kernel functions for estimation, the red line in Figures 5.8, 5.9 and 5.10 are obtained over the whole interval of the real axes.

Censoring is a technique which is applied on a data set that is incomplete due to some random cause. Censoring of data is used in cases when it is known that attribute values belong in the interval $[x,y]$, however, attribute's values are observable only in the interval $[xL,yR]$, where $x < xL$ and $y > yR$.

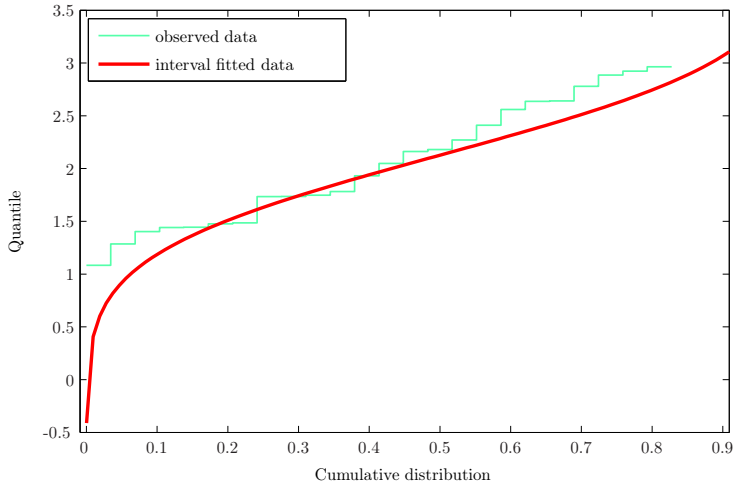


Figure 5.9: Smoothing of inverse distribution function

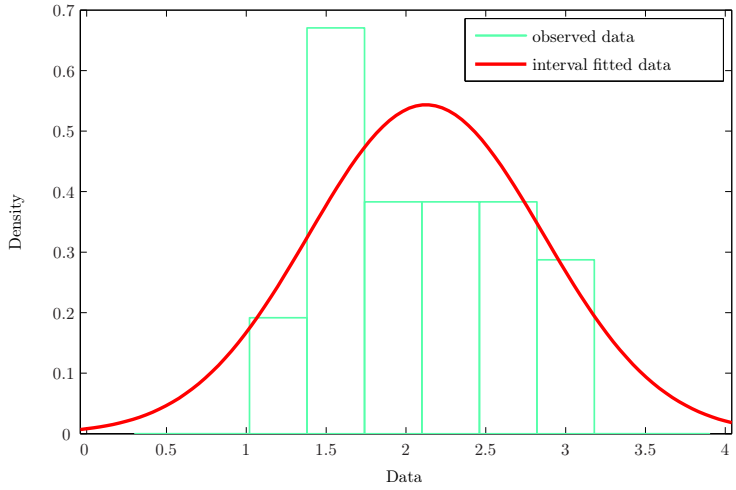


Figure 5.10: Smoothing of density function

Different censoring techniques may be found in Topp (2002) (Klein and Moeschberger, 2003). When some observations in a sample are censored, a convenient approach to estimating the distribution function with the distinctive feature that the steps are not all of size 1/n. This suggests that in the case of censored data a density estimate should be constructed of the form

$$f_c(y) = \sum_{i=1}^{n} v_i w(y - y_i; h)$$

where v_i denotes the size of the jump at observation y_i.

6 Regression Using Nested Archimedean Copulas

The aim of this thesis is to provide ranking of qualitative options as described in Chapter 2. As soon as the qualitative options are mapped into quantitative ones, using the mapping function $F : QS \rightarrow S$, where $S = \{(\mathbf{I_1}, C_1), \ldots, (\mathbf{I_k}, C_k)\}$, $\mathbf{I_i} = [A_{i1}, A_{i2}, ..., A_{in}]$, the next step is to define the function given with (2.2):

$$f : (\mathbf{I}_i, C_i) \in S \rightarrow \mathbb{R}.$$

In this thesis different regression functions are used as a solution to finding a suitable analytical form for the function given with (2.2). Regression task is employed to determine function's parameters that would best model the relation between independent variables (attributes) and the dependent variable (the class). In this chapter the regression function (2.2) is obtained by employing a copula-based approach.

When using copulas, attributes A_i are regarded as random variables. For each random variable its marginal distributions $F_i(A_i)$ are estimated. For determining the dependences among the random variables, one has to find their joint distribution $F(A_1, \ldots, A_n, A_{agg})$. In order to estimate the multi-variate joint densities with a constant estimation accuracy, the required sample size rapidly increases with the number of dimensions (Silverman, 1986). As the problem at hand provides only small sample sizes, one possibility to determine the joint distribution is by employing copula functions.

The final step of using copulas for solving the problem given in Chapter 2 is to derive the regression function. Such a relation will describe the link between the input attributes and the class attribute. The regression with copulas is performed in a way that we determine the probability density from the joint distribution by differentiating over the dependent variable (Brown et al., 2005; Wasserman, 2006). The regression using bi-variate copulas is described in (Bouyé and Salmon, 2002; Nelsen, 2006).

This thesis extends the bi-variate copula-regression algorithm to multi-variate regression using FNAC and PNAC in which the dependent variable may be positioned at any leaf in the hierarchical copula (see Figures 5.5 and 5.7). This approach is demonstrated on Clayton copula, however, the same procedure is used later in the thesis for the results obtained with Frank and Gumbel copulas (see Table 5.1).

To perform regression, first the partial derivative in respect with the dependent variable is obtained:

$$q = \frac{\partial C(u,v)}{\partial u} = -\frac{1}{\theta}(u^{-\theta} + v^{-\theta} - 1)^{-\frac{1+\theta}{\theta}}(-\theta u^{-(1+\theta)}) = (1 + u^{\theta}(v^{-\theta} - 1))^{-\frac{1+\theta}{\theta}} \tag{6.1}$$

solving for $q \in [0;1]$, where q is the quantile, leads to v:

$$v = [1 - u^{-\theta} + (qu^{1+\theta})^{-\frac{\theta}{1+\theta}}]^{-\frac{1}{\theta}} \tag{6.2}$$

Replacing u with $F(x)$ and v with $G(y)$ (Nelsen, 2006), where $F(x)$ and $G(y)$ are CDFs of the random variables x and y leads to:

$$G(y) = [1 - F(x)^{-\theta} + (qF(x)^{1+\theta})^{-\frac{\theta}{1+\theta}}]^{-\frac{1}{\theta}}. \tag{6.3}$$

Finaly, to obtain the different quantile regression curves for the variable Y one needs to find the inverse of (6.3) (Chen, 2005; Koenker, 2005):

$$y = G^{-1}[(1 - F(x)^{-\theta} + (qF(x)^{1+\theta})^{-\frac{\theta}{1+\theta}})^{-\frac{1}{\theta}}] \tag{6.4}$$

Function (6.4) represents one solution for the required function (2.2) in the problem definition given in Chapter 2. This way, a non-linear regression is performed, instead of a linear one as in the QQ method. The curve for copula regression depends on the value of θ. Different regression curves that are obtained with different values for θ in the Clayton copula are given in Figure 6.1.

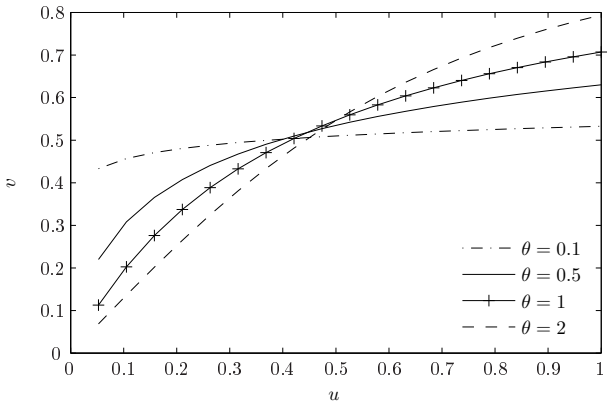


Figure 6.1: Copula-based regression curves for different values of θ in (6.2)

6.1 Quantile Regression

The non-linear regression curves (6.4) used for option ranking when using the copula-based approach, may receive different forms based on two parameters in the regression models:

1. the values of the parameters θ and
2. the quantile q.

In the first case, one may fix the quantile q and examine the curves that are obtained for different values of θ in model. Such an example is given in Figure 6.1, where the regression lines are obtained for fixed value of $q = 0.5$, while examining different values for θ in the Clayton copula-based model (6.2). Figure 6.1 shows that when the value of θ increases, the non-linear regression curves approach the perfect linearity curve ($u = v$).

In the second case, the values of θ are fixed, and regression curves for different values of the quantiles q are presented. The quantile q is a value which divides the distribution $F(x)$ so that there is a given proportion of observations below the quantile. Such examples with different values of q are given in Figures 6.2–6.5. The provided examples show the quantiles $q = 0.1, 0.2, ..., 0.9$ for Clayton and Frank copula-based regression curves, and $q = 0.01, 0.05, 0.25, 0.5, 0.75, 0.95, 0.99$ for Gumbel copulas in Figures 6.2–6.5, respectively. The figures illustrate 200 observations for pairs (u, v) and (X, Y) represented with dots, and different curves that represent the corresponding non-linear quantile regression estimates obtained from bi-variate copulas with the following parameters:

1. Clayton with uniform marginals and $\theta = 0.7$ (Figure 6.2).
2. Frank with Student marginals and two degrees of freedom for: $\theta = 2.5$ (Figure 6.3) and $\theta = -2.5$ (Figure 6.4). In the study of several random variables, the distribution function of each one is called a marginal (Papoulis, 1991). The student-t distribution of a random variable x with n degrees of freedom is given with:

$$f(x) = \frac{\gamma}{\sqrt{(1+x^2/n)^{n+1}}}$$

$$\gamma = \frac{\Gamma[(n+1)/2]}{\sqrt{\pi n \Gamma(n/2)}}$$

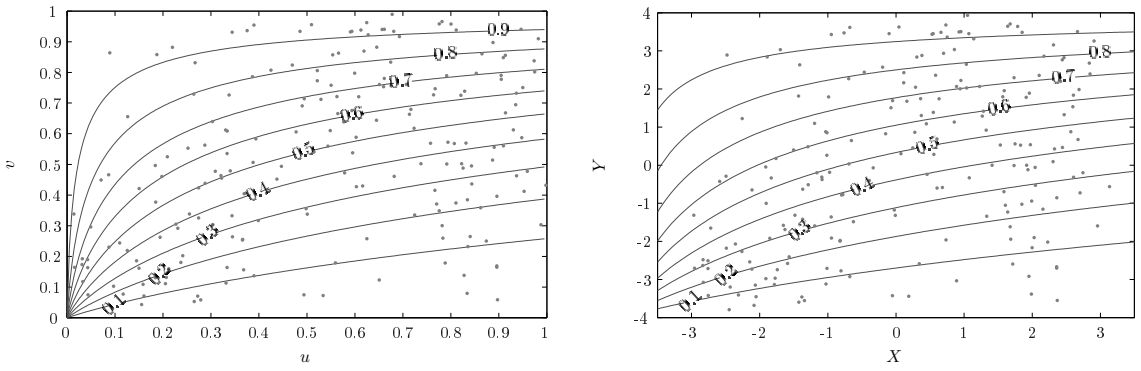
where $\Gamma(\cdot)$ is the gamma function:

$$\Gamma(n) = \int_0^\infty y^{n-1} e^{-y} dy, \;\; b > -1$$

When the argument of gamma function is integer, the function is called generalized factorial due to $\Gamma(n+1) = n\Gamma(n) = \cdots = n!$, and $\Gamma(1) = 1$. The uniform distribution of a random variable x that receives values in the interval (x_1, x_2) is given with:

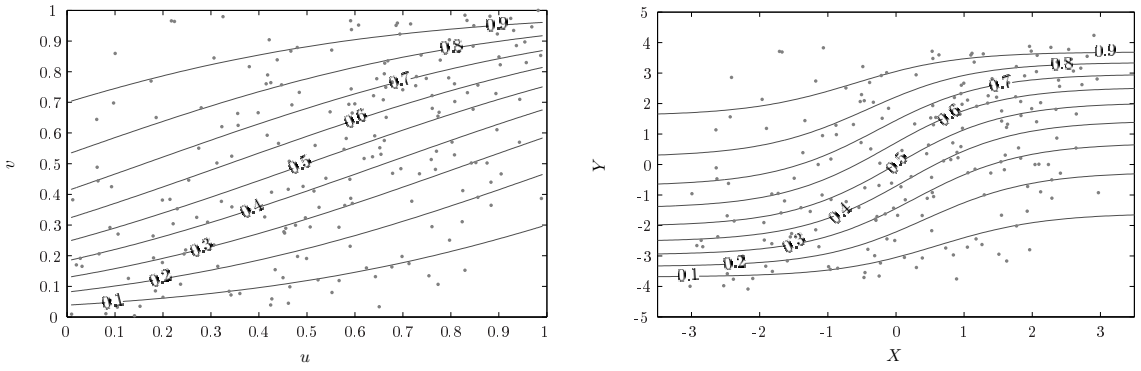
$$f(n) = \begin{cases} \frac{1}{x_2 - x_1} & x_1 \leq x \leq x_2 \\ 0 & \text{otherwise.} \end{cases}$$

3. Gumbel with uniform marginals and $\theta = 5.7681$ (Figure 6.5).



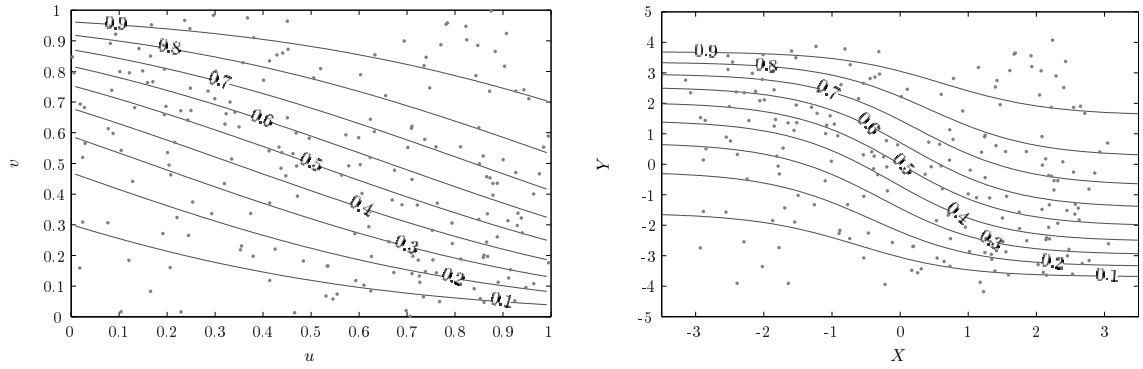
(a) **(b)**

Figure 6.2: Clayton q^{th} quantile curves (for $q = 0.1, 0.2, ..., 0.9$): (a) for (u, v) and (b) for (X, Y) under hypothesis for uniform margins and for $\theta = 0.7$



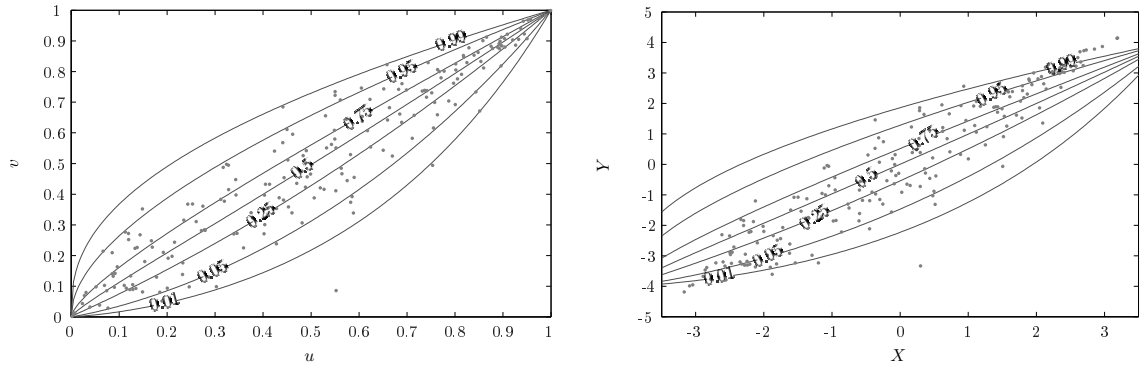
(a) **(b)**

Figure 6.3: Frank q^{th} quantile curves (for $q = 0.1, 0.2, ..., 0.9$): (a) for (u, v) and (b) for (X, Y) under hypothesis for Student margins with two degrees of freedom and for $\theta = 2.5$



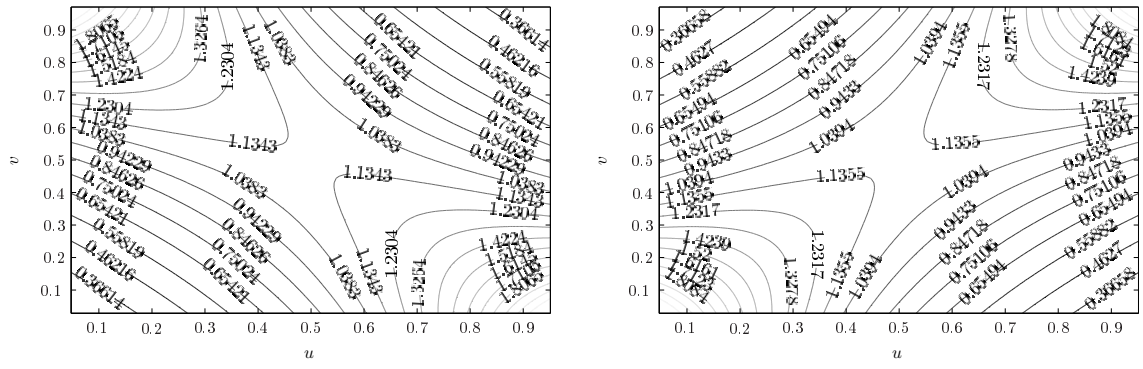
(a) **(b)**

Figure 6.4: Frank q^{th} quantile curves (for $q = 0.1, 0.2, ..., 0.9$): (a) for (u,v) and (b) for (X,Y) under hypothesis for Student margins with two degrees of freedom and for $\theta = -2.5$



(a) **(b)**

Figure 6.5: Gumble q^{th} quantile curves (for $q = 0.01, 0.05, 0.25, 0.5, 0.75, 0.95, 0.99$): (a) for (u,v) and (b) for (X,Y) under hypothesis for uniform margins and for $\theta = 5.7681$



(a) **(b)**

Figure 6.6: Frank copula contours curves: (a) for $\theta = -2.25$ and (b) for $\theta = 2.25$ under hypothesis for Student margins with two degrees of freedom

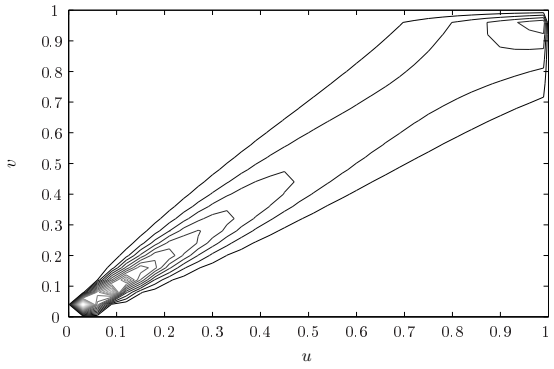


Figure 6.7: Clayton copula contours curves for $\theta = 5.7$ under hypothesis for uniform margins

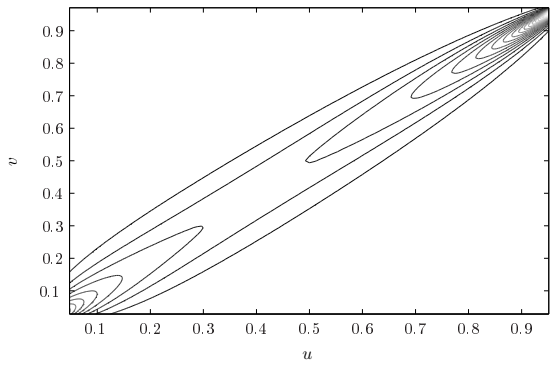


Figure 6.8: Gumbel copula contours curves for $\theta = 5.7$ under hypothesis for uniform margins

In each of the examples in the second case, different values of θ are used to provide difference in the visibility of the examples. In Figure 6.4, a negative value for θ is provided that leads to decreasing regression curves, unlike all other cases with increasing regression curves for positive values of θ.

The contour curves for the Frank, Clayton and Gumbel copulas are given in Figures 6.6, 6.7 and 6.8 respectively. The values on the curves represent the calculated values of the copulas in each of the cases.

6.2 Regression with FNAC: Different Positions of Dependent Variable in the Input Level in the FNAC

Equation (6.4) holds for bi-variate copulas or for a FNAC where the dependent variable enters the construction last, such as u_5 in Figure 5.5. For such a case, the left hand side of the FNAC can be regarded as a single variable that enters the top-most bi-variate copula together with the dependent variable. Therefore the dependent variable value can be determined as (6.4) for Clayton copula. In cases when such a FNAC breaches the conditions (5.16), one has to rearrange the order in which variables enter the FNAC. The rearranging of the order is possible since the joint distribution function is invariant on the order of the variables. Consequently, by rearranging the order in which the variables enter the FNAC, one can ensure that the resulting FNAC fulfills the condition (5.16).

To solve the regression task in this case, first a FNAC has to be obtained that fulfills the condition (5.16). Afterwards, the regression part is solved as an iterative procedure starting from the highest level of FNAC and descending down to the level which contains the dependent variable by substituting the correct quantities in (6.2). In each of the iterations, the value of q in (6.2) is substituted with a vector of regression values obtained from the previous iteration procedure.

The final regression function for obtaining the values of the dependent variable has different forms depending on the position i in the FNAC with n attributes. For example, for the Clayton FNAC, the regression function reads:

$$u_i = \begin{cases} \left[1 - v^{-\theta} + \left(qv^{1+\theta}\right)^{-\frac{\theta}{1+\theta}}\right]^{-\frac{1}{\theta}} & i = n \\ \left[1 - C_{i-2}^{-\theta_{i-1}} + \left(v_{i+1} C_{i-2}^{-1+\theta_{i-1}}\right)^{\frac{\theta_{i-1}}{1+\theta_{i-1}}}\right]^{-\frac{1}{\theta_{i-1}}} & i \in \{3,...,n-1\} \\ \left[1 - u_j^{-\theta_1} + \left(v_3 u_j^{-1+\theta_1}\right)^{-\frac{\theta_1}{1+\theta_1}}\right]^{-\frac{1}{\theta_1}} & i,j = 1,2, i \neq j. \end{cases} \tag{6.5}$$

Here and in section 6.3, the position of the variable in the FNAC and PNAC is denoted with i or j and should not be confused with the order of the attributes in both hierarchical constructions. The position is determined with counting starting from the leftmost input variable in the FNAC and PNAC, and going to the rightmost variable. For the example given in Figure 5.6 in the middle position, the position of the output variable is 2, while the output variable at this position is u_3 having a subscript 3. Further on, the equations in the thesis are derived regarding the positions in the FNAC and PNAC, while for the examples, these values have to be adopted to the names of the real variables.

6.3 Regression with PNAC: Case of Four and Five Attributes and Generalization to n Attributes

The second type of hierarchical copula structures that are used in this thesis are PNAC. The smallest PNAC may be formed with four attributes.

In the PNAC with four attributes, for example the copula $C_2(C_{1,1}, C_{1,2}; \theta_2)$ formed with u_1 - u_4 in Figure 5.7a, the regression function depending on the position i, and based on the Clayton bi-variate copula, reads:

$$u_i = \left[1 - u_j^{-\theta_k} + \left(v u_j^{-1+\theta_k}\right)^{-\frac{\theta_k}{1+\theta_k}}\right]^{-\frac{1}{\theta_k}}, v = \left[1 - {v_s}^{-\theta_r} + \left(v_p {v_s}^{1+\theta_r}\right)^{-\frac{\theta_r}{1+\theta_r}}\right]^{-\frac{1}{\theta_r}}, \tag{6.6}$$

where

$$(k, v_s, r, v_p) = \begin{cases} k = 1, v_s = C_2, r = 3, v_p = q \text{ for } i, j = 1, 2, i \neq j, \\ k = 2, v_s = C_1, r = 3, v_p = q \text{ for } i, j = 3, 4, i \neq j. \end{cases}$$

When PNAC is formed with five attributes, there are two possible constructions as given in Figure 5.7.

For PNAC with five attributes, as the one shown in Figure 5.7a, the regression function has two possible forms:

1. when the dependent variable positioned at u_5, the regression form is given with:

$$u_5 = \left[1 - {C_2}^{-\theta_3} + \left(q {C_2}^{-1+\theta_3}\right)^{-\frac{\theta_3}{1+\theta_3}}\right]^{-\frac{1}{\theta_3}}. \tag{6.7}$$

2. For other positions u_i, $i \in \{1, 2, 3, 4\}$, the regression equation is given with (6.6) where:

$$(k, v_s, r, v_p) = \begin{cases} (11, C_{12}, 2, v), \text{ where } v(v_s, r, v_p) = v(u_5, 3, q) \text{ for } i, j = 1, 2, i \neq j, \\ (12, C_{12}, 2, v), \text{ where } v(v_s, r, v_p) = v(u_5, 3, q) \text{ for } i, j = 3, 4, i \neq j. \end{cases}$$

In case of a PNAC with five attributes as the one shown in Figure 5.7b, the regression equation is given with (6.6) where:

$$(k, v_s, r, v_p) = \begin{cases} (11, u_3, 2, v), \text{ where } v(v_s, r, v_p) = v(C_{12}, 1, q) \text{ for } i, j = 1, 2, i \neq j, \\ (11, C_{11}, 2, v) \text{ where } v(v_s, r, v_p) = v(C_{12}, 1, q) \text{ for } i = 3 \text{ and } u_j = C_{11}, \\ k = 12, v_s = C_2, r = 1, v_p = q \text{ for } i, j = 4, 5, i \neq j. \end{cases}$$

The mathematical reasoning The mathematical reasoning of these equations is the following. For a PNAC that consists of four attributes, for example the copula $C_2(C_{1,1},C_{1,2};\theta_2)$ formed with u_1 - u_4 in Figure 5.7a, and the dependent variable u_i, where $i \in 1,2$, the regression function gets the following form:

$$u_i = \left[1 - u_j^{-\theta_{11}} + \left(v u_j^{-1+\theta_{11}}\right)^{-\frac{\theta_{11}}{1+\theta_{11}}}\right]^{-\frac{1}{\theta_{11}}}, i,j = 1,2, i \neq j$$

where

$$v = \left[1 - {C_2}^{-\theta_3} + \left(q {C_2}^{1+\theta_3}\right)^{-\frac{\theta_3}{1+\theta_3}}\right]^{-\frac{1}{\theta_3}}.$$

When the dependent variable is u_i, where $i \in 3,4$, the regression function reads:

$$u_i = \left[1 - u_j^{-\theta_{12}} + \left(v u_j^{-1+\theta_{12}}\right)^{-\frac{\theta_{12}}{1+\theta_{12}}}\right]^{-\frac{1}{\theta_{12}}}, i,j = 3,4, i \neq j$$

where

$$v = \left[1 - {C_1}^{-\theta_3} + \left(q {C_1}^{1+\theta_3}\right)^{-\frac{\theta_3}{1+\theta_3}}\right]^{-\frac{1}{\theta_3}}.$$

In case of a PNAC with five attributes as the one shown in Figure 5.7a, the regression function gets one of the two following forms.

For PNAC with five attributes, as the one shown in Figure 5.7a, the regression function has two possible forms:

1. In case the dependent variable is u_5, the regression form is (6.7), which is obtained from (6.2) by substituting the appropriate labels.
2. When the dependent variable is u_i where $i \in 1,2$, the regression function is:

$$u_i = \left[1 - u_j^{-\theta_{11}} + \left(v_1 u_j^{-1+\theta_{11}}\right)^{-\frac{\theta_{11}}{1+\theta_{11}}}\right]^{-\frac{1}{\theta_{11}}}, i,j = 1,2, i \neq j \tag{6.8}$$

where

$$v_1 = \left[1 - {C_{12}}^{-\theta_2} + \left(v_2 {C_{12}}^{1+\theta_2}\right)^{-\frac{\theta_2}{1+\theta_2}}\right]^{-\frac{1}{\theta_2}}$$

and

$$v_2 = \left[1 - {u_5}^{-\theta_3} + \left(q {u_5}^{1+\theta_3}\right)^{-\frac{\theta_3}{1+\theta_3}}\right]^{-\frac{1}{\theta_3}} \tag{6.9}$$

Similarly, in case when the dependent variable is u_i where $i \in 3,4$, the regression function is:

$$u_i = \left[1 - u_j^{-\theta_{12}} + \left(v_1 u_j^{-1+\theta_{12}}\right)^{-\frac{\theta_{12}}{1+\theta_{12}}}\right]^{-\frac{1}{\theta_{12}}}, i,j = 3,4, i \neq j \tag{6.10}$$

where

$$v_1 = \left[1 - {C_{11}}^{-\theta_2} + \left(v_2 {C_{11}}^{1+\theta_2}\right)^{-\frac{\theta_2}{1+\theta_2}}\right]^{-\frac{1}{\theta_2}}$$

and v_2 is given with (6.9).

In case of a PNAC with five attributes as the one shown in Figure 5.7b, the following forms of the regression functions are possible.

1. If the dependent variable is u_1 or u_2, the regression function is as given in (6.8), however, the value of v_1 in this case is given with:

$$v_1 = \left[1 - {u_3}^{-\theta_2} + \left(v_2 {u_3}^{-1+\theta_2}\right)^{-\frac{\theta_2}{1+\theta_2}}\right]^{-\frac{1}{\theta_2}}$$

   where

$$v_2 = \left[1 - {C_{12}}^{-\theta_3} + \left(v_2 {C_{12}}^{-1+\theta_3}\right)^{-\frac{\theta_3}{1+\theta_3}}\right]^{-\frac{1}{\theta_3}} \tag{6.11}$$

2. If the dependent variable is u_3, the regression function is:

$$u_3 = \left[1 - {C_{11}}^{-\theta_2} + \left(v_2 {C_{11}}^{-1+\theta_2}\right)^{-\frac{\theta_2}{1+\theta_2}}\right]^{-\frac{1}{\theta_2}}$$

   where v_2 is given with (6.11).

3. If the dependent variable is u_4 or u_5, the regression function is given with (6.10) where $i, j = 4, 5$, $i \neq j$ and only the value of v_1 changes to

$$v_1 = \left[1 - {C_2}^{-\theta_3} + \left(q {C_2}^{-1+\theta_3}\right)^{-\frac{\theta_3}{1+\theta_3}}\right]^{-\frac{1}{\theta_3}}.$$

Generalization of PNACs to n attributes When the number of attributes increases, the number of possible combinations of PNACs increases, too. However, the obtained PNACs will consist of subsets of elements presented in Figure 5.7. Hence the careful combination and repetition of these equations will lead to the required regression functions for all kinds of PNACs regardless of the number of attributes. Additionally, the maximum number of parameters θ_i that have to be estimated will always be equal to the number of input attributes.

6.4 Number of Possible FNAC and PNAC Structures

The number of possible FNAC structures is $\frac{n!}{2}$, where n is the number of variables. The possible combinations of PNACs increases with the number of attributes. However, the obtained PNACs will consist of subsets of elements presented in Figure 5.7. Hence the careful combination and repetition of equations (6.5) and (6.6) will lead to the required regression functions for all kinds of PNACs regardless of the number of attributes. Additionally, the maximum number of parameters θ_i that have to be estimated will always be equal to the number of input attributes. The number of possible full binary trees that may be used for calculating copula functions, as function of the number of its attributes, is given in Table 6.1 (Bohla and Lancaster, 2006; Sloane, 2011).

Table 6.1: Number of PNAC structures depending on the number of attributes

Number of attributes	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Number of trees	1	1	2	3	6	11	24	47	103

6.5 Running Example for Regression Using FNAC

For demonstration of the regression using FNAC, consider the example given in Table 2.2, for which different FNACs are built and their θ_i parameters are given in Table 5.3. For this example, the position of the attributes in the FNAC is u_1, u_3, u_2, as given in Figure 5.6 (b), where u_3 is the variable that is obtained from mapping the class attribute. Therefore, the regression should be performed at this position, leading to two regression iterations which use (6.5). In the first iteration the value of v_3 is calculated using the equation

$$u_i = [1 - v^{-\theta} + (qv^{1+\theta})^{-\frac{\theta}{1+\theta}}]^{-\frac{1}{\theta}}$$

where $u_i = v_3$, $q = 0.5$, $v = u_2$ and $\theta = \theta_2 = 0.1880$ for Clayton-based FNAC. Thus

$$v_3 = [1 - {u_2}^{-0.1880} + (0.5{u_2}^{1.1880})^{-\frac{0.1880}{1.1880}}]^{-\frac{1}{0.1880}}$$

In the second and last iteration, the final regression equation is:

$$u_i = [1 - u_j^{-\theta_1} + (v_3 u_j^{-1+\theta_1})^{-\frac{\theta_1}{1+\theta_1}}]^{-\frac{1}{\theta_1}} \tag{6.12}$$

where the position of the regression variable is given with $i = 2$, $j = 1$ and $\theta_1 = 0.6125$. Using (6.12) and substituting $u_i = u_2$ with the name of the variable u_3 as given in Figure 5.6 in the middle position, leads to

$$u_3 = [1 - u_1^{-0.6125} + (v_3 u_1^{-0.3875})^{-\frac{0.6125}{1.6125}}]^{-\frac{1}{1.6125}}.$$

The obtained values for u_3 are used for the last step of obtaining the values of the inverse distribution function (for example see (6.4)), which are the regression values that are required. The regression values are given in Table 6.2, in column **Clayton FNAC**. The final step in calculation of the regression values

Table 6.2: Quantitative ranking of options

No.	**A_1**	**A_2**	**C**	**Clayton FNAC**	**FNAC Normalized**
1	1	1	1	1.4151	0.8876
2	2	1	1	1.9020	1.2950
3	1	2	1	1.5744	1.0209
4	1	3	1	1.6624	1.0945
5	3	1	2	2.1685	2.2062
6	2	2	2	2.0604	2.0085
7	3	2	3	2.3228	3.2035
8	2	3	3	2.1467	2.9090
9	3	3	3	2.4062	3.3432

is application of the third stage of QQ. This stage is important because it corrects the inconsistencies

that might have occured in the second phase, and leads to consistence between the regression values and the original model. One such inconsistency that is solved using this stage, is given with options 5 and 8. Namely, option 5 has higher ranking value than option 8, however it belongs to a smaller class than option 8. This is corrected and the obtained values after application of the third stage are given in Table 6.2, in column **FNAC Normalized**.

Applying the copula method on the example given in Table 2.2, leads to the regression curves that are given in Figure 6.9. Here the regression curves are obtained with calculation of the regression values of the FNAC built with Gumbel bi-variate copula for the options in Table 2.2. The contours of the regression curves are given in Figure 6.10, respectively.

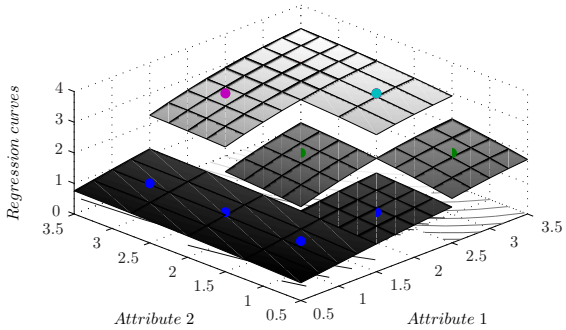


Figure 6.9: Regression curves obtained with FNAC built with Clayton bi-variate copulas

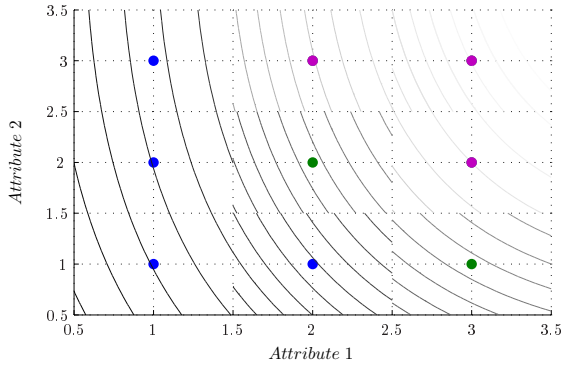


Figure 6.10: Contours of the regression curves given in Figure 6.9.

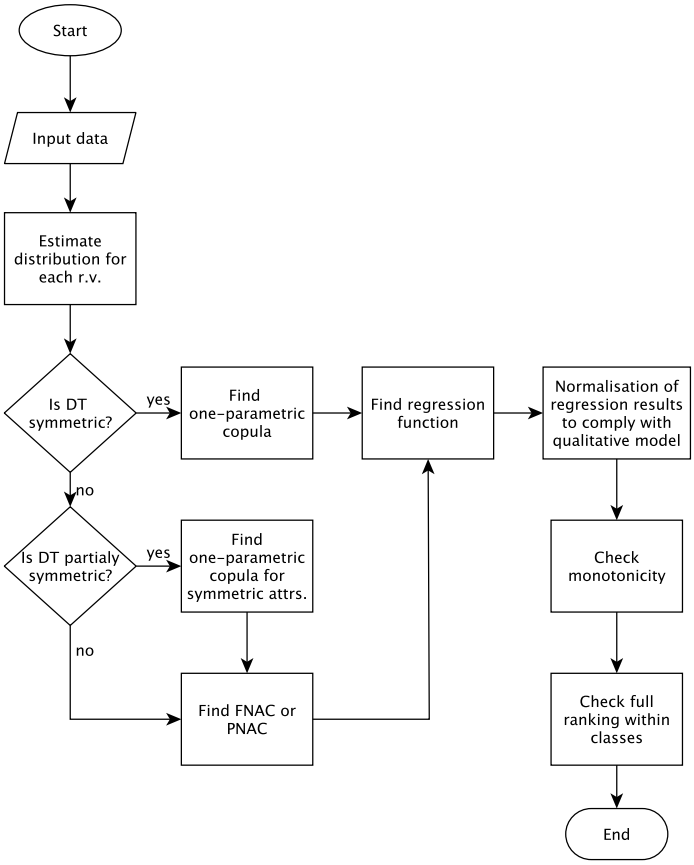
6.6 Copula-Based Option Ranking Algorithms

The approach of option ranking with copulas is a computationally much more demanding compared with the approach of option ranking with the linear methods. Therefore the goal of this section is to make this task more user-friendly by providing the algorithms required for ranking of options.

The copula-based option ranking algorithm is given as Algorithm 3. The algorithm starts with the distribution estimation of the random variables. Then it checks if the DT consists of symmetric attributes. In case of symmetric DT of random variables (see Definition 2.4 for symmetric DT), the algorithm proceeds with estimation of a one-parametric copula construction. If DT is not symmetric, the algorithm checks if the DT is partially symmetric. In case of partially symmetric DT (see Definition 2.5 for partially symmetric), the algorithm estimates one-parametric copulas for each subgroup of symmetric attributes. Next, a FNAC or PNAC is built from the obtained copulas and the rest of the attributes, if any. In case

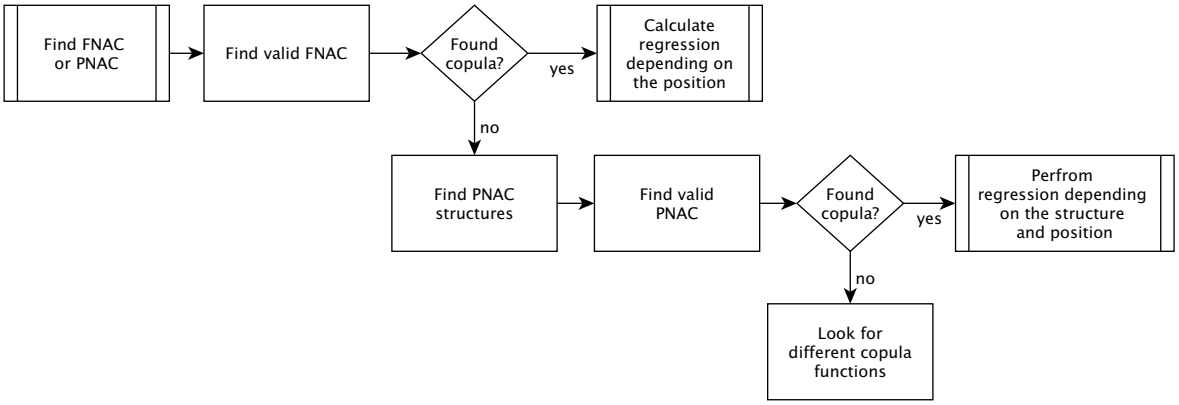
of non-symmetric DT, the algorithm searches for a valid FNAC or PNAC by using bi-variate copulas. For the obtained copula constructions, a suitable regression function is defined according to (6.1)–(6.6). Next normalization of the results is performed as given with (3.3)–(3.5). Finally full ranking of options is checked as defined in (2.4).

Algorithm 3 Implementation algorithm



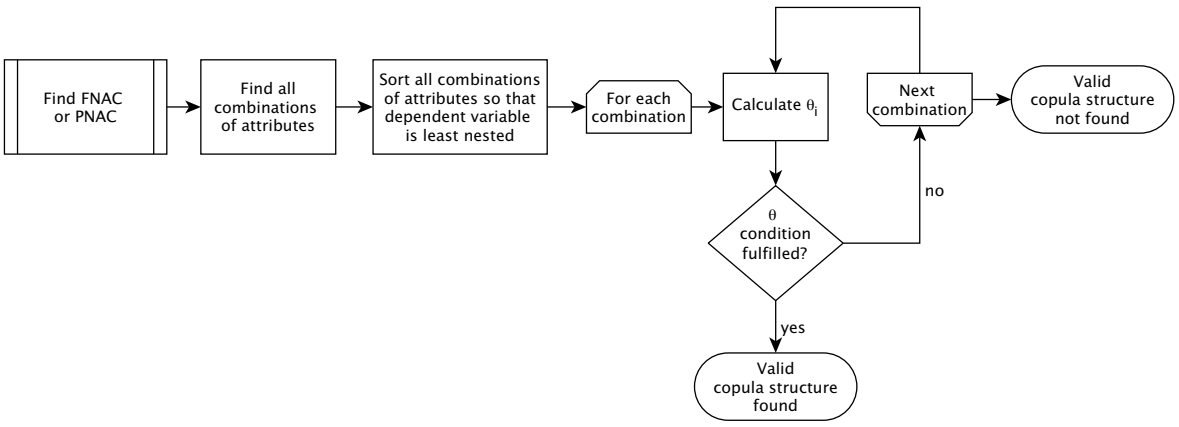
The process of finding a suitable PNAC or FNAC is presented in Algorithm 4. The tree-like structure of the FNAC has the same form regardless of the number of attributes, in contrast to the diferent tree-like shapes that the PNAC may receive, and whose number increases with the number of the attributes (see Table 6.1). Therefore it is less complex to look for a valid FNAC hence the algorithm first searches for a valid FNAC. In case it does not exist, the algorithm continues with a search for a valid PNAC.

Algorithm 4 Branching the algorithm for search of a valid hierarchical copula



To check the validity of the nested copula construction (FNAC or PNAC) one has to evaluate all possible input combinations of the attributes, as presented in Algorithm 5. The algorithm sorts the possible combinations of attributes, in a way that in the following step the simpliest cases are first examined for formation of a valid FNAC. The simplest cases are determined in terms of the regression function that will be formed for the valid FNAC. Hence the simplest FNAC is considered the one for which the dependent variable enters the construction as least nested (right most position). If such a structure fails to produce a valid copula, all possible nested constructions are examined. For each generated construction, the values of θ_i are calculated and conditions (5.16)–(5.17) are checked. If there are more solutions than one, the algorithm picks the one where the dependent variable is least nested.

Algorithm 5 Search algorithm for a valid FNAC



6.6.1 Regression Algorithms for FNAC and PNAC

For bi-variate copula, a suitable regression function linking the two variables can be found by Algorithm 6 (Nelsen, 2006).

In Algorithm 6, first the copula function is differentiated over the input variable to derive the regression function. Then the obtained expression is set equal to a quantile value q, leading to quantile regression function,.

To use Algorithm 6, the dependent variable should be placed in the right most position in the FNAC, such as the variable u_5 in Figure 5.5. Algorithm 7, on the other hand, gives a general solution for copula-

based regression with FNAC for n random variables where the regression variable seats in an arbitrary position p (Mileva-Boshkoska and Bohanec, 2012).

Algorithm 6 Regression using bi-variate copula

1: $\frac{\partial C_\theta(u,v)}{\partial u} = q$ ▷ calculate median regression for $q = \frac{1}{2}$
2: $v \leftarrow Solve(\frac{\partial C_\theta(u,v)}{\partial u} = q, v)$ ▷ see Table 5.1 for different copulas
3: $u \leftarrow F_1(x_1)$ ▷ replace u by $F_1(x_1)$
4: $v \leftarrow F_2(x_2)$ ▷ replace v by $F_2(x_2)$

Algorithm 7 Regression algorithm for FNAC structure and dependent variable in the p position

1: $v \leftarrow 0.5$
2: $q \leftarrow 0.5$ ▷ calculate median regression for $q = \frac{1}{2}$
3: **if** $p == n$ **then** ▷ if regression variable is positioned last,
4: ▷ n is the number of random variables/attributes;
5: $v \leftarrow Solve(\frac{\partial C_\theta(u,v)}{\partial u} = q, v)$ ▷ calculate v
6: **else**
7: **for** $i = 1 \rightarrow (n-p)$,
8: (or $i = 1 \rightarrow (n-2)$, when p=1,2) **do** ▷ if position of regression
9: ▷ variable other than the last
10: $q \leftarrow v$ ▷ replace q with the value of v
11: $v \leftarrow Solve(\frac{\partial C_\theta(u,v)}{\partial u} = q, v)$ ▷ recalculate the new value of v
12: $i \leftarrow i-1$
13: **end for** ▷ p is the output variable position
14: $q \leftarrow v$ ▷ replace q with the value of v
15: $v \leftarrow Solve(\frac{\partial C_\theta(u,v)}{\partial u} = q, v)$ ▷ recalculate v; if $p = 1$, $u = u_2$,
16: ▷ if $p = 2$, $u = u_1$
17: **end if**
18: $u \leftarrow F_1(x_1)$ ▷ replace u by $F_1(x_1)$
19: $v \leftarrow F_2(x_2)$ ▷ replace v by $F_2(x_2)$

Algorithm 7 performs regression in iterations. It starts with regression at the topmost copula. The obtained regression values are propagated downwards in the hierarchical structure, where the value of q is replaced with the regression values of v. The iterations continue until the dependent variable p in FNAC is reached. Finally, the regression function is obtained as in Algorithm 6, for $q = v$ from the last iteration.

The Algorithm 7 is adopted as a building block for obtaining regression functions from PNACs as well. An example of Algorithm 7 for the Clayton copula is given in Algorithm 8. The algorithm for calculation of k_c and n_c is different in case of copula-based regression. Due to the nonlinearity of the method, we need to find the minimum and maximum value that may occur when applying the built copula structure, and then apply them for calculation of k_c and n_c parameters in (3.4) and (3.5). The procedure is given in Algorithm 9. After the model is built, we may use it for evaluation of new options. The evaluation phase uses the saved model structure however it requires to modify the attribute's order according to the model and then to perform several calculations as given in Algorithm 10.

Algorithm 8 Regression algorithm for FNAC structure and dependent variable in the p position for Clayton copula

$v \leftarrow 0.5$
$q \leftarrow 0.5$ ▷ calculate median regression for $q = \frac{1}{2}$
if $p == n$ **then** ▷ if regression variable is positioned last; n is the number of random variables/attributes;
    $v \leftarrow [1 - u^{-\theta} + (qu^{1+\theta})^{-\frac{\theta}{1+\theta}}]^{-\frac{1}{\theta}}$ ▷ calculate v
else
    for $j = 1 \rightarrow (n-p)$,
(or $j = 1 \rightarrow (n-2)$, when p=1,2) **do** ▷ if position of regression variable other than the last
        $q \leftarrow v$ ▷ replace q with the value of v
        $v \leftarrow [1 - u^{-\theta} + (qu^{1+\theta})^{-\frac{\theta}{1+\theta}}]^{-\frac{1}{\theta}}$ ▷ recalculate the new value of v
        $i \leftarrow i - 1$
    end for ▷ p is the output variable position
    $q \leftarrow v$ ▷ replace q with the value of v
    $v \leftarrow [1 - u^{-\theta} + (qu^{1+\theta})^{-\frac{\theta}{1+\theta}}]^{-\frac{1}{\theta}}$ ▷ recalculate v; if $p = 1$, $u = u_2$; if $p = 2$, $u = u_1$
end if
$u \leftarrow F_1(x_1)$ ▷ replace u by $F_1(x_1)$
$v \leftarrow F_2(x_2)$ ▷ replace v by $F_2(x_2)$

Algorithm 9 Calculate k_c and n_c for copula-based regression algorithm

for $i = 1 \rightarrow c$ **do** ▷ c is the number of classes
    for $j = 1 \rightarrow nc$ **do** ▷ nc is the number of options **n** in the class c
        $min_p(j) \leftarrow C(\mathbf{n} - 0.5)$ ▷ C is the regressor from the FNAC/PNAC
        $max_p(j) \leftarrow C(\mathbf{n} + 0.5)$
    end for
    $min_c \leftarrow min(min_p)$ ▷ find the minimum value in the class c
    $max_c \leftarrow max(max_p)$ ▷ find the maximum value in the class c
    $k_c = \frac{1}{max_c - min_c}$ ▷ calculate k_c as in (3.4)
    $n_c = c + 0.5 - k_c max_c$. ▷ calculate n_c as in (3.5)
end for

Algorithm 10 Option evaluation using copula-based algorithm

1: Reorder the option according to the obtained PNAC/FNAC model.
2: Estimate the option attribute's CDF, using the distribution functions provided by the model.
3: Calculate the values of the bi-variate copula values in the FNAC / PNAC, using the parameters in the model.
4: Perform the regression task as in Algorithm 7.
5: Apply k_c and n_c on the obtained result to obtain the evaluation value.
6: Propagate the evaluation value to the next hierarchical table. ▷ Applies only for the hierarchical case.

6.6.2 Implementation of the Copula-Based Algorithms

All presented algorithms 3–10 for performing copula-based regression are developed and implemented in MATLAB, which originally implements only bi-variate copulas. To perform copula-based regression, a toolbox was developed which covers building of FNACs, PNACs and regression with hierarchical copulas on a hierarchical setting of decision tables.

6.7 Hierarchical Running Example for Usage of Copula-Based Option Ranking Algorithm

Attributes in DEX form tree structures. Such example is shown in Figure 3.1. The example represents a DEX model for cars evaluation, and it is used here to present the propagation of the regression values from one hierarchical level to the next one. As explained in section 3.3, in the hierarchical models the evaluation values, obtained from the aggregation of basic attributes into a class attribute, are propagated in the next higher level as values of an input attribute. These are further on aggregated, and the procedure is repeated to the top most aggregation (class) attribute. The evaluation values are continuous, when using QQ for aggregation, which may not capture the information about the class into which a certain option belongs to. In order to propagate the class with the evaluation value of the option, QQ introduces the third stage, of ensuring that the evaluation result belongs into the interval $[c_i - 0.5, c_i + 0.5], c_i \in C$. This third stage of QQ is used when aggregation is performed with copula functions as well. For the given example in Figure 3.1, the final evaluation of cars is obtained at the topmost hierarchical level.

The qualitative decision tables for aggregation of attributes, and their mappings into quantitative ones are presented in Tables 3.2, 3.3 and 3.4 and Tables 3.5, 3.6 and 3.7 respectively. The evaluation of options, represented as cars in the given example, is a two-step process. In the first step, copula-based models are built for each of the Tables 3.5, 3.6 and 3.7. For example, using the Frank bi-variate copulas, FNACs are built for each of the aggregated attributes **Costs**, **Safety** and **Car**. Their parameters are given in Table 6.3. The option rankings obtained with these models for each of the aggregated attributes are given in columns **Eval** in Tables 6.4, 6.5 and 6.6.

Table 6.3: Parameters of copula-based models for evaluation of car

Parmeter	Costs	Safety	Car
θ_1	5.4187	15.2057	11.5403
θ_2	2.1108	1.4573	0.9783
kc	[0.6239 0.4887 0.6239]	[0.9202 0.6255 0.2386 0.2908]	[0.5573 0.7473 0.2176 0.3284 0.4166]
nc	[0.0779 1.0226 1.4265]	[-0.0383 0.6659 2.1434 2.6474]	[0.1612 0.6051 2.2204 2.6672 2.9178]

In the second step, the evaluation values obtained for attributes **Costs** and **Safety** are propagated to the copula-based model for the aggregated attribute **Car** for final evaluation. For example, the following case:

$$if\ Price = 1\ and\ Maint. = 1\ then\ Costs = 0.6744$$

$$if\ ABS = 1\ and\ Size = 1\ then\ Safety = 0.8296$$

leads to the following evaluation by the model built for aggregation of the attribute **Car**:

$$if\ Costs = 0.6744\ and\ Safety = 0.8296\ then\ Car = 0.6102$$

Table 6.4: Option rankings of **Car**

Costs	Safety	Car	Eval
1	1	1	0.6576
2	1	1	0.6857
3	1	1	0.7160
1	2	1	1.0850
2	2	2	1.9599
3	2	3	2.6488
1	3	3	2.8578
2	3	3	2.8867
2	4	3	3.2125
3	3	4	3.7175
1	4	4	4.0956
3	4	5	4.9118

Table 6.5: Option rankings of **Costs**

Price	Maint.	Costs	Eval
1	1	1	0.6744
2	1	1	1.1318
1	2	1	0.8156
3	1	2	2.2672
2	2	2	2.0000
1	3	2	1.7328
3	2	3	3.1844
2	3	3	2.8682
3	3	3	3.3256

Table 6.6: Option rankings of **Safety**

ABS	Size	Safety	Eval
1	1	1	0.8296
2	1	1	0.9304
1	2	2	1.9656
2	2	3	2.6863
1	3	3	3.0427
2	3	4	3.8443

The same procedure is used for evaluation of all options.

This model reflects the change of the values of basic attributes in the final evaluation of options. For example, if the value of $ABS = 1$ is changed for one unit into $ABS = 2$, the final car evaluation changes from $Eval = 0.6102$ to $Eval = 0.6326$. The slight increment in the final evaluation reflects the decision maker's preferences for having a car with ABS to a car which does not have the ABS. The same may be demonstrated in case when change of the value of basic attribute changes the class into which an option belongs to. For example:

$$if\ Price = 1\ and\ Maint. = 1\ then\ Costs = 0.6744$$

$$if\ ABS = 2\ and\ Size = 2\ then\ Safety = 2.6863$$

$$if\ Costs = 0.6744\ and\ Safety = 2.6863\ then\ Car = 2.7679.$$

Now, let change the class of the basic attribute *Size* from $Size = 2$ to $Size = 3$. This changes the class into which the option belongs to, from $Safety = 3$ to $Safety = 4$, which is propagated to the next level, in which the class of the option for the attribute *Car* is changed from $Car = 3$ to $Car = 4$. This is reflected in the evaluations of options as follow:

$$if\ Price = 1\ and\ Maint. = 1\ then\ Costs = 0.6744$$

$$if\ ABS = 2\ and\ Size = 3\ then\ Safety = 3.8443$$

$$if\ Costs = 0.6744\ and\ Safety = 3.8443\ then\ Car = 3.9872.$$

7 Experimental Evaluation on Artificially Generated Data Sets

This chapter provides comparison among all regression methods that are presented in Chapters 4–6. The comparison of methods consists of evaluation of the methods in terms of fulfilling the conditions (2.3)–(2.4), and assessment of their performances in different experimental set-ups.

7.1 Datasets

Three groups of artificial datasets of decision tables are generated, with different number of input attributes and different value scales:

Dataset 1: a set of all decision tables that consist of two input attributes and attribute values $\{1,2,3\}$, with all possible combinations results in 19 683 decision tables. The data set is small, meaning that all generated decision tables can be used for processing with no requirements for expensive computer or algorithmic setups.

Dataset 2: a collection of decision tables that consist of three input attributes and the set of attribute values $\{1,2,3\}$ with all possible combinations. The whole possible set comprises of 3^{27} different tables. In order to choose a smaller sampling subset, the principle of maximum entropy is used. The maximum entropy of the discrete probability distribution is the uniform distribution, hence only the decision tables where the class attribute is uniformly distributed are examined. From the uniformly distributed subset, each 10000^{th} decision table was selected leading to a dataset of 2 278 734 decision tables.

Dataset 3: a collection of tables that consist of four input attributes. Each attribute receives equal number of values from the set of attribute values $\{1,2,3,4,5\}$. The full decision table with four attributes, each one with cardinality of five, consists of 625 options. From them we choose 25 as shown in Table 7.1. The experimental dataset 3 consists of 1 000 000 decision tables containing 25 options each. The output attributes are obtained by systematically sampling each 1000^{th} vector starting from a randomly selected first output attribute.

7.2 Evaluation Results of the Performed Experiments

Methods are evaluated and compared on each of the three datasets, denoted as three experiments. The evaluation criteria is based on the percentage of decision tables that are fully ranked, and which fulfill monotonicity property (2.3) at the same time. The monotonicity property is checked in a two-stage procedure as described in Section 2.4. First a set of groups of options is determined so that each group consists of comparable options. Then the monotonicity property (2.3) is checked within each of the groups.

Table 7.1: Input attributes of the decision tables used in dataset 3

No.	A_1	A_2	A_3	A_4
1	1	1	1	1
2	2	4	1	1
3	5	3	3	1
4	4	1	4	1
5	4	2	5	1
6	3	5	1	2
7	2	1	2	2
8	5	2	2	2
9	1	3	2	2
10	4	5	3	2
11	1	4	1	3
12	4	3	2	3
13	3	3	3	3
14	5	1	4	3
15	2	5	5	3
16	5	5	1	4
17	1	2	3	4
18	2	2	3	4
19	4	4	4	4
20	3	1	5	4
21	3	4	2	5
22	3	2	4	5
23	2	3	4	5
24	1	4	5	5
25	5	5	5	5

The experimental evaluations of the discussed methods are presented in Table 7.2. The first column in Table 7.2 are all considered methods. In addition, the union of the results obtained with all copula methods is included under the name of *All copulas*. It provides the number of solved decision tables with at least one of the proposed copula functions: Frank, Clayton or Gumbel. The graphical representations of experimental evaluations are given in Figures 7.1–7.5. All results are discussed in comparison to the original QQ method, by grouping them as follows. Firstly, all copula-based methods are compared to QQ, then the constrained-based optimization method is discussed followed by the CIPER and New CIPER. Finally, the modified QQ methods are compared to the original QQ method.

7.2.1 Results from Copula-Based Methods

Results from Experiment 1

The percentage of decision tables ranked using the three examined copula-based methods, denoted as Frank, Gumbel and Clayton are shown in Figure 7.1. The percentage of solved decision tables is notably higher than those ranked by QQ. In addition, the union of all different decision tables ranked by the three copula-based methods increases the number of fully ranked decision tables from less that 10 %, ranked by QQ, to more than 70 %. This clearly indicates that copula-based methods outperform QQ in

Table 7.2: Percentage of monotonic and fully ranked decision tables by different methods

No.	Method	Experiment 1 (%)	Experiment 2 (%)	Experiment 3 (%)
1	QQ	8.74	0.45	0.09
2	Frank	29.75	17.23	84.01
3	Gumbel	63.56	24.54	80.00
4	Clayton	36.18	54.05	96.38
5	All copulas	73.23	65.78	99.03
6	Optimization	0.15	0	0
7	CIPER	0.82	0.002	0
8	New CIPER	14.88	0.0001	0.01
9	gB	67.07	66.64	56.15
10	gC	29.78	11.79	38.75
11	gP	93.07	96.57	98.01
12	IG	73.31	80.61	89.61
13	χ^2	74.32	67.38	56.15

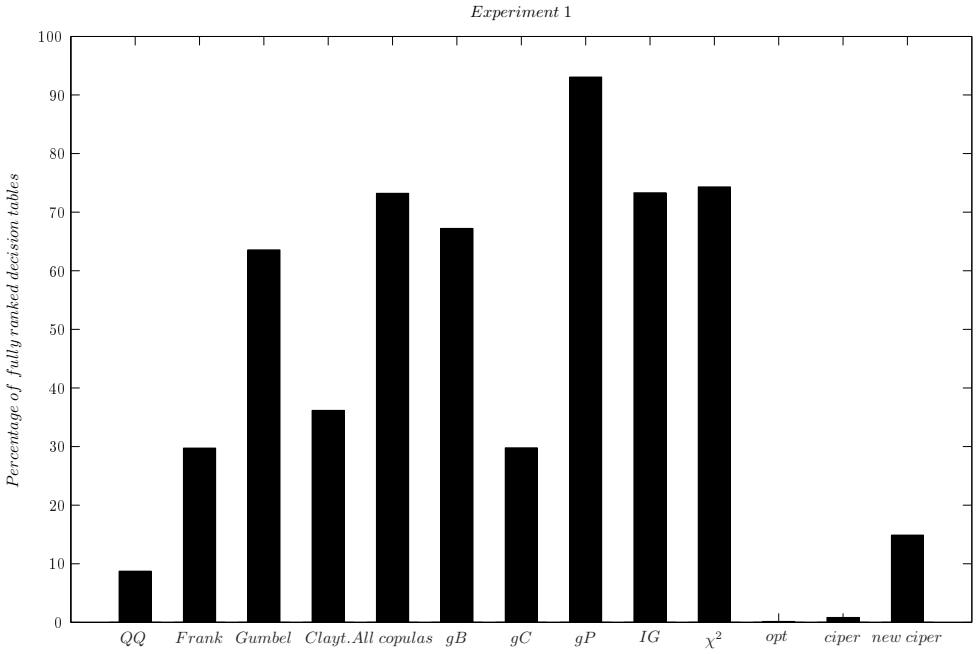


Figure 7.1: Percentage of fully ranked and monotonic decision tables obtained with different methods on dataset 1

terms of the proportion of fully ranked decision tables and indeed alleviates the QQ difficulties outlined in Chapter 3. The reasons for such a good performance of copulas in comparison to QQ may be the following:

1. The dataset 1 consists of linear, nearly linear and non-linear decision tables. Copulas are functions which are used mainly for describing non-linear data. In the dataset 1, as well as in datasets 2 and 3, the number of non-linear tables is higher than the linear ones. QQ on the other hand, is a method that performs best for linear or nearly linear decision tables.

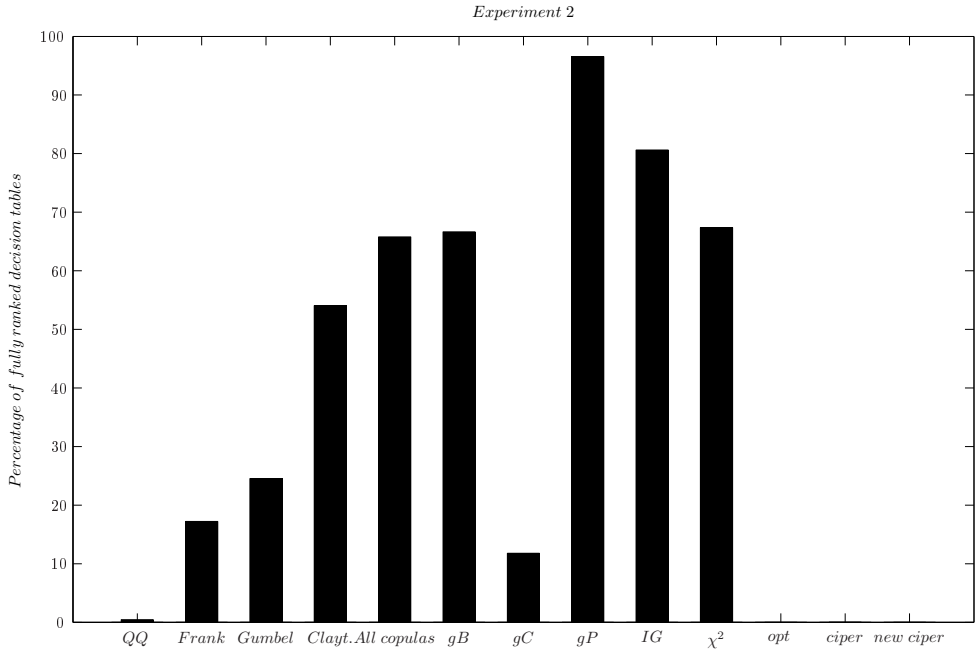


Figure 7.2: Percentage of fully ranked and monotonic decision tables obtained with different methods on dataset 2

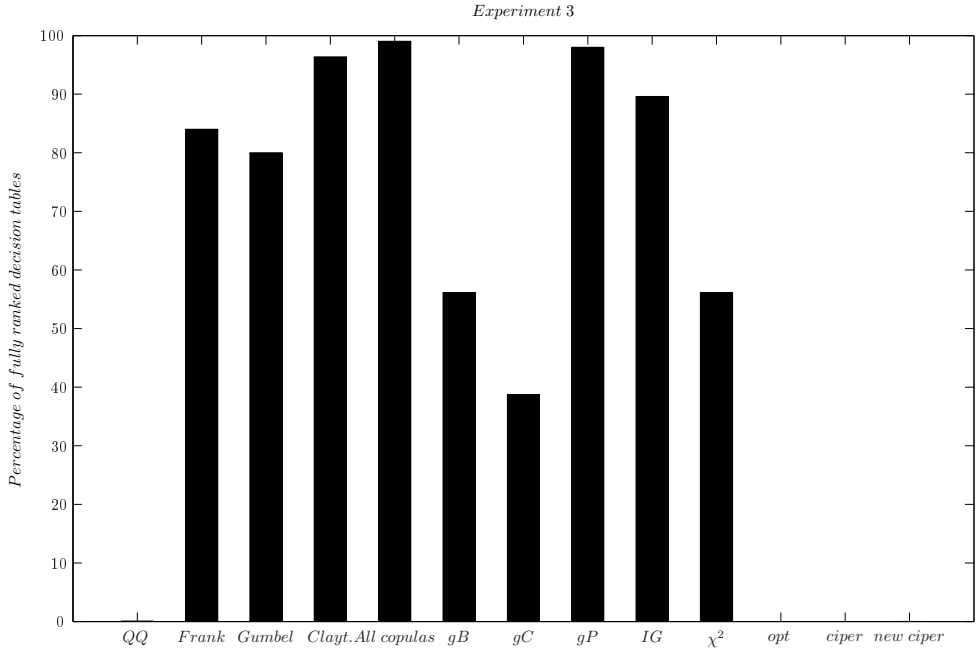


Figure 7.3: Percentage of fully ranked and monotonic decision tables obtained with different methods on dataset 3

2. For each decision table that consists of two input and one class attribute, there are three possible FNACs that may provide up to three different models that are checked for full ranking. Unlike copulas, QQ provides only one model for each decision table. The number of possible copula-based models increases with the number of attributes as explained in section 6.4, while QQ would

always build only one model regardless of the number of attributes. This increases the chances that the copula-based models would indeed provide a solution.

The percentages of fully ranked and monotone decision tables, for each of the methods, are presented in Table 1 in Appendix A.

Results from Experiment 2

In these experiments, the only possible PNAC is the one provided in Figure 7.4. Any bi-variate copula built from combination of any of the two variables, will lead to the smallest possible value of $\theta_{1i}, i \in \{1,2\}$ in Figure 7.4. Hence, in the next level, the value of θ_2 would usually increase compared to one of the values θ_{1i}. The newly obtained copula breaches PNACs constraints:

$$\theta_2 \leq \theta_{11},\ \theta_2 \leq \theta_{12}.$$

Therefore in experiment 2 the copula-based results are obtained only with FNACs.

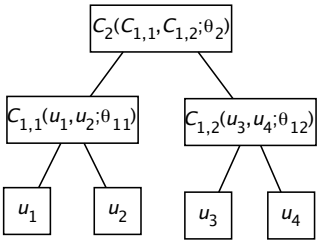


Figure 7.4: FNAC with four variables

The percentages of fully ranked decision tables with QQ and three copula-based approaches are shown in Figure 7.2. QQ fully ranks only less than 1 % of decision tables, while copula-based methods collectively rank more than 65 %. The reasons why copula-based models provide better results than QQ are similar as in dataset 1. Here the number of non-linear tables is much higher than the linear or nearly linear tables, and the number of models built with FNAC using one copula function is up to 12, versus only one when using QQ. This increases the possibility that the copula-based model would provide the desired ranking.

The percentages of fully ranked and monotone decision tables, for each of the methods, are presented in Table 2 in Appendix A.

Results from Experiment 3

We used both FNACs, given in Figure 5.5, and PNACs, given in Figure 5.7, to build copula-based models.

The obtained results are presented in Figure 7.5. From the nine bars, the first three represent results obtained with Clayton copula-based method, the next three are results obtained with Frank copula-based method, and the last three with Gumbel copula-based method. In each of the three groups, the first bar are results obtained with PNACs, the second are results obtained with FNACs, and the last are combined results from the first two. Compared with the results from ranking obtained with QQ in Figure 7.3, these results show an increase of the number of fully ranked decision tables from less than 0.1 %, solved by QQ, to more than 99 %, when the results of all copula-based methods are combined. For the rest of the decision tables, it was not possible to build a FNAC or PNAC with the examined copulas, or results lead to partial ranking.

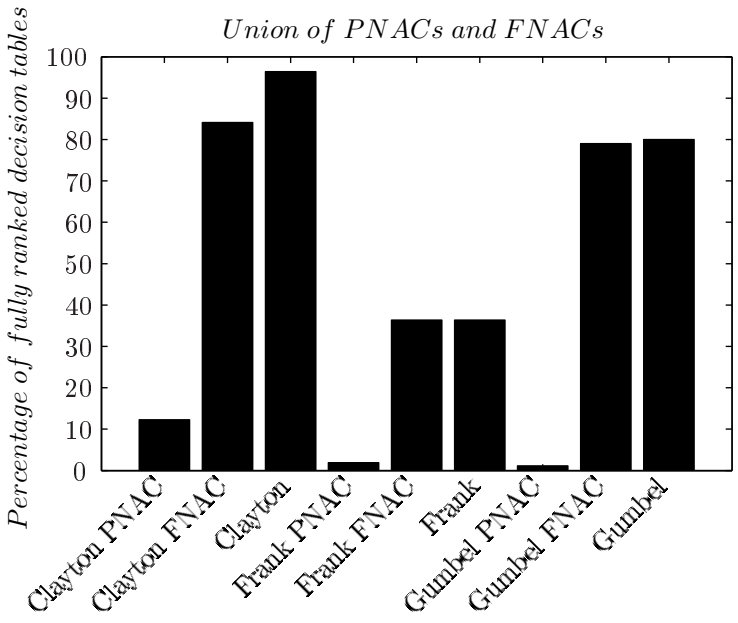


Figure 7.5: Union of PNACs and FNACs

These results confirm that we have found a way of increasing the number of fully ranked decision tables using the copula-based method. Additionally, the performance of the copula-based method improves with the number of attributes and their cardinality.

7.2.2 Results from Experiments Based on Constraint Optimization Approach

The results of the constraint optimization approach are presented in Table 7.2. They show that the reformulation of the problem as constrained optimization task does not lead to better results than those obtained with the current state-of-the-art QQ method. Instead it leads to definition of constraints that are overly stringent thus no feasible point could be found in the second and third experiment. Several research directions arise for overcoming this issues:

1. redefinition of the objective function (for example minimization of the error between the estimated class attribute $\hat{C}$ and the real class C (Kosmidou et al., 2008))

2. relaxing these constrains (for example define the constraints for comparable options so that monotonicity holds, and introduce the third stage of QQ).

These are left for future work.

7.2.3 Results from Experiments Based on CIPER and New CIPER

Although polynomial functions have several promising properties such as simple form, computational easiness of use and independence of the used metric (linear data transformations result in a polynomial model being mapped to another polynomial model), the experiments here showed that polynomials exhibit worse results compared to the rest of the models. This is a consequence of the unwanted polynomial properties as well as the desired properties of the resulting ranking function (Shestopaloff, 2011):

1. Polynomial models have a shape/degree tradeoff. It means that complicated data structures, are modeled with high degree polynomials, leading to high number of estimated parameters. High-

degree polynomial models are known for oscillatory behavior at the edges of the interval in which they are fitted, leading to poor interpolatory properties.

2. Additionally, when they provide good data fits, the goodness of fit may rapidly deteriorate outside the data range leading to poor extrapolatory properties.

3. Polynomial models have poor asymptotic properties. In comparison to copulas that use function mapping $[0,\infty) \rightarrow [0,1]$, polynomials have an infinite response if some variable takes an infinite value.

4. The obtained coefficients in the polynomial models may be sensitive to small variations in the data.

5. Although many of the solutions here fulfill the monotonicity property required by the ranking function, polynomial models obtained with CIPER and New CIPER provide ties in the rankings in most cases.

Details of the obtained results are given in Tables 6 and 7 in Appendix A.

7.2.4 Results Obtained with QQ when Modified with Impurity Functions

Although some of the linear methods exhibit the best results in comparison to the rest of the methods, linear methods have several drawbacks. Some of them are the following (Seeber and Lee, 2003):

1. Exhibit only linear relationships

   Linear models tell how much a change in a variable effects the outcome variable leading to a measurement of linear relationships between dependent and independent variables. However this may not always be correct. A simple example is the relationship between income and age. In this example the relationship between two variables should be curved, since income tends to rise in the elderly parts of life, then to flatten and decline in retirement. One possible improvement may be achieved with fitting of piecewise linear functions (Greco et al., 2008), in which the independent variable is partitioned into intervals and a separate linear function is fit to each interval. Still, the following two drawbacks hold also for piecewise linear functions.

2. Provide a relationship only with the mean of the dependent variable

   Linear regression provides a relationship between the mean of the dependent and independent variables. However, the mean is not a complete description of a single variable. Consequently the linear regression is not a complete description of relationships among variables. In order to deal with this problem, one should use quantile regression, which is used with copulas in this thesis.

3. Variables must be independent

   Linear regression assumes that the data at hand are independent, meaning that the attributes' values of one option have nothing to do with those of another. Although this situation is frequent, it is not always sensible. A typical example are student test scores when students come from various classes, grades and schools. Students from the same class tend to be similar, thus they are not independent. For example, such students would come from the same neighborhoods, they have the same teachers, etc. Unlike linear models, copulas exhibit different types of dependencies expressed via the values of θ, such as tail dependence and concordance.

According to the results of experiments, gP given with (4.5) provides best results in all three cases in terms of monotonicity and full ranking. In spite of that, there are cases where copulas outperform them:

1. The example provided in Table 7.3, consists of attributes and a class with different distributions. In this case the regression values obtained with the QQ and its modifications, represented with the column *Impurity* 7.3, are the same and provide only partial ranking of options, as options 5 and 7 are ranked the same.

2. The example provided in Table 7.4 consists of two symmetric attributes. There are six different options, where one may argue that the methods should provide six different ranks. QQ based methods manage to provide five, while copulas provide six different ranks. The reasoning comes from the fact that the option 2, which receives the value of 2 for both attributes, is classified in the lest preferred class 1. This option differs from the rest two options in class 1 by having all vales of its attributes other than 1, hence one could expect that it would receive different evaluation from the other two options.

Table 7.3: Example 1 of fully ranked options only with copula functions

No.	A_1	A_2	C	QQ	Imp.	Clayton	ClaytonN	Frank	FrankN	Gumbel	GumbelN
1	1	1	1	1.0000	1.0000	1.0258	1.3473	1.2313	0.9213	1.2324	1.2895
2	2	1	2	1.7500	1.7500	2.2435	2.2907	2.1146	1.7335	1.9282	1.7362
3	3	2	2	2.2500	2.2500	3.4073	2.4508	3.9744	2.1642	3.9253	1.9865
4	1	3	2	2.0000	2.0000	1.6780	2.2129	1.5394	1.6003	1.7406	1.7127
5	3	1	3	2.9000	2.9000	2.8785	3.3582	3.5736	3.0142	3.4790	2.8920
6	1	2	3	2.7000	2.7000	1.4625	3.1707	1.3773	2.5774	1.4122	2.6777
7	2	2	3	2.9000	2.9000	2.7656	3.3433	2.3555	2.7719	2.1892	2.7583
8	2	3	3	3.1000	3.1000	3.0115	3.3758	2.6223	2.8250	2.6487	2.8059
9	3	3	3	3.3000	3.3000	3.6475	3.4600	4.4189	3.1823	4.6694	3.0155

Table 7.4: Example 2 of fully ranked options only with copula functions

No.	A_1	A_2	C	QQ	Imp.	Gumbel	GumbelN	Clayton	ClaytonN	Frank	FrankN
1	3	1	1	1.0000	1.0000	1.7130	1.2515	0.7677	1.1514	1.6332	0.8102
2	2	2	1	1.0000	1.0000	1.9188	1.3895	2.0767	1.3928	1.8483	1.1175
3	1	3	1	1.0000	1.0000	1.7130	1.2515	0.7677	1.1514	1.6332	0.8102
4	2	1	2	2.0000	2.0000	1.6680	2.4587	0.7677	2.3244	1.5744	1.9232
5	1	2	2	2.0000	2.0000	1.6680	2.4587	0.7677	2.3244	1.5744	1.9232
6	1	1	3	2.6667	2.6667	1.5850	2.7087	0.7344	2.9848	1.4803	2.5454
7	3	2	3	3.1667	3.1667	2.0941	2.7941	2.1426	3.1793	2.0561	2.9517
8	2	3	3	3.1667	3.1667	2.0941	2.7941	2.1426	3.1793	2.0561	2.9517
9	3	3	3	3.3333	3.3333	2.4341	2.8511	3.3108	3.3408	2.4388	3.2218

7.2.5 Time Execution of Methods

One parameter which is of interest is the time needed to compute the different models and the outcomes from the regression functions. The results are given in Table 7.5 and are obtained by averaging the time over 100 decision tables for all methods. The first and second columns in Table 7.5 give the number of options and input attributes respectively. The third column provides the time that is needed to calculate

all QQ based methods simultaneously, and the next three columns provide time needed to get results with the copula-based method that uses Clayton, Frank and Gumbel copulas respectively, and the last column gives the execution time for constraint optimization method. All experiments are performed on UNIX based system equipped with Intel(R) Core(TM) i7 CPU 870 @2.93GHz.

Table 7.5: Time execution in seconds of different methods averaged over 100 calculations

No. of options	No. of inputs	All QQ	Clayton	Frank	Gumbel	Constraint optimization
9	2	0.0349	0.3069	0.2883	0.2749	0.0575
27	3	0.1159	0.9934	0.6022	0.7718	1.4285
25	4	0.1017	1.8255	1.0135	0.8935	1.6216

It is clear that time execution is higher for the copula-based methods in comparison to other methods and it increases with the number of input attributes. This can be explained with the fact that the number of possible FNAC structures increases with the number of input attributes, therefore the time for searching the solution increases as well.

7.3 Summary

The motivation of the presented experiments was to investigate the usage of different functions for evaluation of options so that (2.3)–(2.5) are fulfilled. Presented results show that the usage of copulas is justified by the number of possible solutions that they provide in each of the experiments. Based on the presented results one may conclude the following.

- The best results in the first and the second experiment are obtained with the modified QQ method with the Gini population function.
- The best results in the third experiment are obtained using the combined copula-based results and QQ method modified with the Gini population.
- The combinations of copula-based results presented in Figure 7.5 show that combination of PNACs and FNACs improve the final evaluation of the copula-based regression method.
- Although the modified QQ with Gini population performs best, there are cases which may be solved only with the copula-based method, such the ones presented in Tables 7.3 and 7.4.

8 Illustrative Examples

In this chapter, the copula-based approaches are illustrated on three examples. The examples illustrate the methodology on small decision tables, which consists of three to five attributes that receive values from the subset $\{1,2,3,4,5\}$. The examples were specially chosen to highlight several important properties of copula-based approach [1]:

1. to show that monotonicity condition may be fulfilled when using FNAC for copula-based regression, for cases when QQ fails to fulfill it,
2. to show that monotonicity condition may be fulfilled when using PNAC for copula-based regression, for cases when QQ and FNACs fail to fulfill it, and
3. to demonstrate the behavior of copulas when dealing with symmetric decision tables

8.1 FNAC Solves the Breaching Monotonicity

The first example in this chapter demonstrates the ability of FNAC to produce regression functions which overcome the problem of breaching monotonicity that is present when using QQ. The example that is considered for this purpose is given in Table 8.1. The first column is the option number and the next four columns are the quantitative values of attributes followed by the corresponding class. The last four columns in Table 8.1 represent the calculations obtained with QQ, and with the different copula-based regression functions from FNACs built with Clayton, Gumbel and Frank bi-variate copulas. The values of the θ_i parameters and the permutations of the attributes in the FNACs for each of the copula-based methods is given in Table 8.2. The values of the θ_i fulfill (5.16), thus each of the obtained regression functions is considered as applicable for further examination.

In contrast to copula-based methods, QQ breaches the monotonicity in several occasions. For example, options 3 and 5 in Table 8.1, show that option 5 is better or at least as good as option 3 for all attributes. Consequently one would expect that the methods would rank option 5 better than option 3. However, this is not the case with QQ calculations, which provide higher ranking value to option 3. The same kind of behavior may be noticed with the following pairs of options: {(16,20), (17,20), (18,20),(19,20), (21,22), (21,24), (21,23), (21,25), (23,25), (24,25)}. Unlike QQ calculations, the values obtained with FNACs based on bi-variate copulas produce correct ranking which does not breach the monotonicity of the comparable options and provides full ranking of options. Hence this example shows that this kind of failures of the QQ method that we want to address may be solved with the regression functions obtained from FNACs.

[1] Methods that use modified QQ with impurity functions, polynomial functions and constraint optimization are also applied on the three examples, and the results are given in Appendix B

Table 8.1: Rankings obtained by different methods

No.	A_1	A_2	A_3	A_4	Class	QQ	Clayton	Gumbel	Frank
1	4	2	5	1	1	0.6699	1.1780	0.7464	0.5533
2	4	5	3	2	1	1.3015	1.3077	0.8658	0.7733
3	2	2	3	4	1	1.3301	1.2213	0.7822	0.6038
4	3	1	5	4	1	0.7326	1.2208	0.7772	0.5877
5	3	2	4	5	1	1.0485	1.3121	0.8700	0.7709
6	2	4	1	1	2	2.3701	2.2092	1.5730	1.5042
7	4	1	4	1	2	1.6771	2.2551	1.5822	1.5074
8	2	1	2	2	2	2.1366	2.2310	1.5770	1.5051
9	5	5	1	4	2	2.3744	2.3563	1.6154	1.5773
10	1	4	5	5	2	1.6256	2.3600	1.6167	1.5769
11	3	5	1	2	3	3.3791	3.1702	2.7387	2.5233
12	5	1	4	3	3	2.7120	3.2411	2.7890	2.5435
13	2	5	5	3	3	2.6209	3.3483	2.9085	2.6660
14	3	4	2	5	3	3.2095	3.3256	2.8788	2.6473
15	2	3	4	5	3	2.8182	3.3299	2.8835	2.6386
16	5	3	3	1	4	3.9091	4.1862	3.5710	3.5257
17	1	4	1	3	4	4.3857	4.0395	3.5470	3.5059
18	4	3	2	3	4	4.1378	4.2549	3.5883	3.5642
19	3	3	3	3	4	3.9674	4.2722	3.5934	3.5781
20	5	5	5	5	4	3.6143	4.4639	3.7306	4.1639
21	1	1	1	1	5	5.3492	5.1125	4.7125	4.5003
22	5	2	2	2	5	5.0527	5.3395	4.9213	4.5444
23	1	3	2	2	5	5.1548	5.2669	4.8352	4.5111
24	1	2	3	4	5	4.9261	5.3069	4.8752	4.5228
25	4	4	4	4	5	4.6508	5.4512	5.1798	4.9086

Table 8.2: Permutations and values of θ_i parameters obtained with FNACs

Method	Clayton	Gumbel	Frank
θ_1	0.1123	1.1419	1.1975
θ_2	0.0729	1.1248	0.2506
θ_3	0.0193	1.0561	0.2021
θ_4	$1.45 \cdot 10^{-6}$	1.0129	0.1756
permutation	1-2-4-3-5	1-2-4-3-5	1-2 4-3 5

8.2 PNAC Solves the Breaching Monotonicity

Regression function originating from a PNAC manages to fulfill all required conditions: the monotonicity, full ranking ad consistency of options, by using only the information provided in Table 8.3. The calculations obtained with QQ, and with the regression function obtained form PNAC built with Frank copulas are presented in the last two columns of Table 8.3. In the given example, a regression function obtained from PNAC built with Frank copulas provides full option ranking. On the other hand, QQ breaches the monotonicity condition (2.3) for pairs (6,7), (6,10), (7,8), (7,10), (8,10), (9,10), (18,19),

Table 8.3: Rankings obtained with QQ and Frank copula using the PNAC given in Figure 5.7a, before and after applying equations (3.3)–(3.5)

No.	A_1	A_2	A_3	A_4	Class	QQ	Frank PNAC
1	2	4	1	1	1	1.3663	1.2650
2	5	2	2	2	1	0.8669	0.7409
3	5	1	4	3	1	0.6337	0.7313
4	2	2	3	4	1	0.9060	1.0777
5	3	1	5	4	1	0.6702	1.0213
6	4	2	5	1	2	1.9640	1.6499
7	2	1	2	2	2	2.1754	1.5710
8	4	4	4	4	2	1.9802	2.1447
9	1	4	5	5	2	2.2128	2.2984
10	5	5	5	5	2	1.7872	2.3867
11	4	3	2	3	3	3.1848	2.7232
12	3	3	3	3	3	3.2488	2.8536
13	3	4	2	5	3	3.2262	3.1023
14	3	2	4	5	3	2.7512	2.9082
15	2	3	4	5	3	3.0527	3.2106
16	5	3	3	1	4	3.8831	3.8229
17	4	1	4	1	4	3.6848	3.5794
18	1	4	1	3	4	4.3152	4.0117
19	2	5	5	3	4	4.1305	4.3784
20	5	5	1	4	4	3.9456	4.1405
21	1	1	1	1	5	5.0429	4.5738
22	3	5	1	2	5	5.2787	5.2563
23	1	3	2	2	5	5.1599	4.9629
24	4	5	3	2	5	5.0352	5.3526
25	1	2	3	4	5	4.7213	4.9564

(18,20), (21,24), (21,25), (22,24), (23,24).

As a lesson from the last two examples, it is recommend to build PNAC in cases when regression functions obtained from FNAC fail to provide full ranking of options. The background for this recommendation comes from the fact that when working with copulas, it is less complex to calculate the regression functions obtained with FNAC than with PNAC.

8.3 Evaluation of Symmetric Decision Tables

Options in symmetric decision tables are evaluated with one-parametric copulas as explained in section 5.4.1. One such example is given in Table 8.4. It is taken from a real case example, which is further discussed in section 9.2. In this example, the qualitative values of attributes *A*–*D* are converted into quantitative one so that they form a monotone decision table, represented with columns A_1–D_1. Symmetric are the attributes *A* and *B*.

Table 8.4: Utility function (**A**–**D**) and mapping from the qualitative attributes into quantitative ones ($\mathbf{A_1}$–$\mathbf{D_1}$). Basic attributes are: **A**–*Generality*, **B**–*Scalability* and **C**–*NoiseSensitivity*. The aggregated attribute is **D**–*Robustness*.

No	A	B	C	D	A_1	B_1	C_1	D_1
1	'low'	'low'	'high'	'very low	1	1	1	1
2	'med'	'low'	'high'	'low'	2	1	1	2
3	'high'	'low'	'high'	'low'	3	1	1	2
4	'low'	'med'	'high'	'low'	1	2	1	2
5	'med'	'med'	'high'	'low'	2	2	1	2
6	'low'	'high'	'high'	'low'	1	3	1	2
7	'low'	'low'	'med'	'low'	1	1	2	2
8	'med'	'low'	'med'	'low'	2	1	2	2
9	'low'	'med'	'med'	'low'	1	2	2	2
10	'low'	'low'	'low'	'low'	1	1	3	2
11	'high'	'med'	'high'	'med'	3	2	1	3
12	'med'	'high'	'high'	'med'	2	3	1	3
13	'high'	'high'	'high'	'med'	3	3	1	3
14	'high'	'low'	'med'	'med'	3	1	2	3
15	'med'	'med'	'med'	'med'	2	2	2	3
16	'high'	'med'	'med'	'med'	3	2	2	3
17	'low'	'high'	'med'	'med'	1	3	2	3
18	'med'	'high'	'med'	'med'	2	3	2	3
19	'med'	'low'	'low'	'med'	2	1	3	3
20	'high'	'low'	'low'	'med'	3	1	3	3
21	'low'	'med'	'low'	'med'	1	2	3	3
22	'med'	'med'	'low'	'med'	2	2	3	3
23	'low'	'high'	'low'	'med'	1	3	3	3
24	'high'	'high'	'med'	'high'	3	3	2	4
25	'high'	'med'	'low'	'high'	3	2	3	4
26	'med'	'high'	'low'	'high'	2	3	3	4
27	'high'	'high'	'low'	'very high'	3	3	3	5

Here QQ method and its modifications that use impurity functions (4.4)–(4.6) provide equal weight values w_i for each of the attributes in (3.2). Hence these cases represent an intersection of the QQ methods because all of them provide the same option evaluation. Evaluations and option ranks using QQ method are given in the column QQ in Table 8.5 and Table 8.6 respectively.

Table 8.6 shows that QQ (as well as its modifications with impurity functions) divide the options into 7 different ranks, Clayton divides them in 8 and Frank and Gumbel provide 10 different ranks. CIPER manages to distinguish seven ranks, while New CIPER finds a polynomial structure that distinguishes

among all 27 options. The question that has to be answered is which ranking one should consider as the most acceptable? In other words, which information in the decision table one may use in order to distinguish among the different rankings?

Firstly, in cases of symmetric attributes, an acceptable aggregation function is the one which provides the same ranking for the symmetric attributes. Therefore the condition for full ranking given with (2.4) has to be relaxed for symmetric attributes. Consequently, the ranking obtained with New CIPER is not acceptable. Considering the heuristics based on which New CIPER works, and relaxing it to work for symmetric attributes, leads to the results obtained with CIPER. The ranking with CIPER are the same as the ranking obtained with QQ.

Table 8.5: Option evaluation using QQ method and Clayton, Frank and Gumbel copulas

No.	$\mathbf{A_1}$	$\mathbf{B_1}$	$\mathbf{C_1}$	$\mathbf{D_1}$	**QQ**	Clayton	ClaytonN	Gumbel	GumbelN	Frank	FrankN
1	1	1	1	1	1.0000	1.6598	1.0000	2.1738	0.5898	2.3049	1.4773
2	2	1	1	2	1.8750	1.6610	0.8865	2.1955	1.5657	2.3229	2.1857
3	3	1	1	2	2.1250	1.6610	0.8865	2.2077	1.5868	2.3325	2.1938
4	1	2	1	2	1.8750	1.6610	0.8865	2.1955	1.5657	2.3229	2.1857
5	2	2	1	2	2.1250	1.6631	0.8886	2.2483	1.6569	2.3655	2.2214
6	1	3	1	2	2.1250	1.6610	0.8865	2.2077	1.5868	2.3325	2.1938
7	1	1	2	2	1.8750	1.6610	0.8865	2.1955	1.5657	2.3229	2.1857
8	2	1	2	2	2.1250	1.6631	0.8886	2.2483	1.6569	2.3655	2.2214
9	1	2	2	2	2.1250	1.6631	0.8886	2.2483	1.6569	2.3655	2.2214
10	1	1	3	2	2.1250	1.6610	0.8865	2.2077	1.5868	2.3325	2.1938
11	3	2	1	3	2.8750	1.6631	2.3494	2.2793	2.6020	2.3900	2.6353
12	2	3	1	3	2.8750	1.6631	2.3494	2.2793	2.6020	2.3900	2.6353
13	3	3	1	3	3.1250	1.6631	2.3494	2.3224	2.6381	2.4241	2.6418
14	3	1	2	3	2.8750	1.6631	2.3494	2.2793	2.6020	2.3900	2.6353
15	2	2	2	3	2.8750	2.7437	2.6219	2.3896	2.6943	2.4821	2.6529
16	3	2	2	3	3.1250	2.7461	2.6225	2.4832	2.7727	2.5591	2.6676
17	1	3	2	3	2.8750	1.6631	2.3494	2.2793	2.6020	2.3900	2.6353
18	2	3	2	3	3.1250	2.7461	2.6225	2.4832	2.7727	2.5591	2.6676
19	2	1	3	3	2.8750	1.6631	2.3494	2.2793	2.6020	2.3900	2.6353
20	3	1	3	3	3.1250	1.6631	2.3494	2.3224	2.6381	2.4241	2.6418
21	1	2	3	3	2.8750	1.6631	2.3494	2.2793	2.6020	2.3900	2.6353
22	2	2	3	3	3.1250	2.7461	2.6225	2.4832	2.7727	2.5591	2.6676
23	1	3	3	3	3.1250	1.6631	2.3494	2.3224	2.6381	2.4241	2.6418
24	3	3	2	4	4.0000	2.7500	3.6229	2.6311	3.6854	2.6814	3.5519
25	3	2	3	4	4.0000	2.7500	3.6229	2.6311	3.6854	2.6814	3.5519
26	2	3	3	4	4.0000	2.7500	3.6229	2.6311	3.6854	2.6814	3.5519
27	3	3	3	5	5.0000	3.8617	4.7091	2.8956	4.7587	2.9035	4.5487

The number of different options that are not symmetric among themselves, in Table 8.6 is 10. Therefore one may argue that methods that provide 10 different ranks are better than those that provide less ranks, under the assumption that monotonicity and consistency are fulfilled in both cases. To support this argument for the particular example, one has to consider the rank of the options according to the classes attribute. It classifies the options from 1 to 5, with preference order given with $1 \prec 2 \prec 3 \prec 4 \prec 5$. Can

Table 8.6: Ranks of options using QQ and FNACs based on Clayton, Frank and Gumbel bi-variate copula

No.	A_1	B_1	C_1	D_1	QQ	Clayton	Gumbel	Frank
1	1	1	1	1	1	1	1	1
2	2	1	1	2	2	2	2	2
3	3	1	1	2	3	3	3	3
4	1	2	1	2	2	2	2	2
5	2	2	1	2	3	4	4	4
6	1	3	1	2	3	3	3	3
7	1	1	2	2	2	2	2	2
8	2	1	2	2	3	4	4	4
9	1	2	2	2	3	4	4	4
10	1	1	3	2	3	3	3	3
11	3	2	1	3	4	5	5	5
12	2	3	1	3	4	5	5	5
13	3	3	1	3	5	5	6	6
14	3	1	2	3	4	5	5	5
15	2	2	2	3	4	6	7	7
16	3	2	2	3	5	7	8	8
17	1	3	2	3	4	5	5	5
18	2	3	2	3	5	7	8	8
19	2	1	3	3	4	5	5	5
20	3	1	3	3	5	5	6	6
21	1	2	3	3	4	5	5	5
22	2	2	3	3	5	7	8	8
23	1	3	3	3	5	5	6	6
24	3	3	2	4	6	8	9	9
25	3	2	3	4	6	8	9	9
26	2	3	3	4	6	8	9	9
27	3	3	3	5	7	9	10	10

this information somehow lead to further separations of option ranks from 7 to 10?

One may notice that the only option that belongs to class 1 has evaluation of all attributes as 1. Options belonging to class 2 have one or two attributes that are evaluated with 1. Options that belong to class 3 have one or none of the attributes evaluated with 1, while none of the options in classes 4 and 5 have evaluation of attributes equal to one. One may conclude that more preferred options are the ones with less evaluation of attributes to one. This information is incorporated in Clayton, Frank and Gumbel copulas, and hence it leads to three more ranks than QQ and CIPER methods.

9 Applications of Copula-Based Method for Option Ranking

This chapter demonstrates the applicability of the copula-based regression method for option ranking on hierarchically combined decision tables which are characteristic decision problems with larger number of attributes, usually more than five. In the following two sections two examples are provided showing how and when the copula-based regression for option ranking in hierarchical settings may be used. In the first example the copula-based decision support system (DSS) is used for assessment of 840 electrically commutated (EC) motors (Mileva-Boshkoska et al., 2013). In the second example a model for ranking of workflows that are uploaded by researchers on the website www.myexperiment.org is built (Mileva-Boshkoska et al., 2012).

9.1 Assessment of Electrically Commutated Motors

Quality assessment of finished products (QAFP) is usually the final step in a manufacturing line. It is performed by aggregating feature set values according to a set of pre-defined preferences and rules. Such a task can be implemented by applying concepts of decision support systems (DSS).

A typical structure of QAFP system is shown in Figure 9.1. Such a system has two inputs: features, extracted from the performed measurements, and expert's preferences regarding the final quality evaluation. These inputs are processed through the two main stages: integration of system's features and expert's preferences, and definition of a copula-based DSS for assessment of overall quality and ranking of finished products. Implementation of these steps has to ensure high sensitivity to the variations in the quality of the finished product. Consequently, each segment of the system has to be custom-made for the problem in hand.

The input of the system is a set of measured features calculated from the acquired vibrations generated by the examined electronically commutated (EC) motor. Furthermore, the system employs available expert's knowledge provided in DEX model tree. Employing copula-based regression functions resulted in a full quality ranking of EC motors. The system was evaluated on a batch of 840 motors. Due to the different background of each segment of this problem, each one is described in details in a depth that is required to understand the problem at hand.

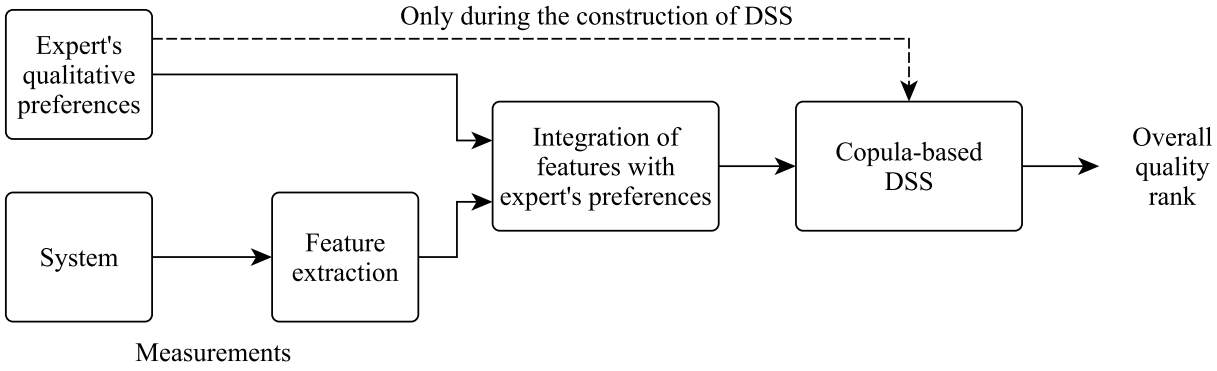


Figure 9.1: Structure of the end-quality assessment system

The extracted feature set should contain the most useful features for determining the fault condition of the monitored system (Vachtsevanos et al., 2006). In absence of faults, feature values should belong to the set of nominal (admissible) values. Any discrepancy in one or several features is regarded as a presence of fault, hence indicating decrease in the overall quality. In order to meet such requirements, two steps should be performed: the most informative features should be selected based on existing fault models, and extraction of their values should be performed using fast and accurate signal processing techniques.

The problem of specifying the most informative feature set in the case of electrical motors has been addressed by many authors (Boškoski et al., 2011; Didier et al., 2007; Juričić et al., 2001; Röpke and Filbert, 1994; Sasi et al., 2001). Generally features are extracted from vibration and/or electrical signals. In our particular case, the features were extracted solely from vibration signals. From a plethora of available signal processing methods, we opted for the well established approach using envelope analysis (Jardine et al., 2006; Peng and Chu, 2004; Randall and Antoni, 2011; Sawalhi et al., 2007). As a pre-processing step we used spectral kurtosis (Antoni, 2006) and cyclostationary analysis (Boškoski et al., 2010), due to their capabilities of selecting the most appropriate frequency band for envelope analysis, hence significantly improving the sensitivity of the approach.

The construction of the copula-based DSS starts with integration of features with expert's preferences. The integration addresses the issues of fusing information from the extracted features into an abstract quality rank based on a set of pre-defined expert's preferences. Usually these preferences are expressed using qualitative grades. For this purpose, qualitative aggregation functions are suitable candidates, such as the ones proposed in the DEX methodology. As DEX uses qualitatively described aggregation functions, it leads to partial ranking of the options at hand. To achieve a full ranking of options, we employ copula functions. Unlike linear regression functions, which tend to provide partial ranking when two or more attributes receive the same weight value (Mileva-Boshkoska and Bohanec, 2011) or evidential reasoning method which sometimes leads to evaluations that are not in line with the expert's expectations (Boškoski et al., 2011), the usage of DEX and copula-based regression leads to high sensitivity to small variations of the input values. This process produces twofold output information. Firstly, the constructed copula-based DSS yields a grade (also called class) to which the examined EC motor belongs. Secondly, it produces a rank value that can be employed for ordering the EC motors within each grade. Therefore, one can easily specify the position of each finished EC motor within the population of produced units based on its quality rank.

Implementing quality assessment systems has the potential for creating sustainable competitive advantage (Reed et al., 2000). The proposed quality assessment system allows immediate detection of quality change for each produced item individually. Furthermore, the quality assessment output can be used by any subsequent manufacturing execution system, which will handle any general changes in the production quality (Ertugrul and Aytac, 2009; Orth et al., 2012).

9.1.1 Feature Selection and Organization in DEX Structure

According to several surveys, bearing faults represent the most common cause for failure of mechanical drives (Albrecht et al., 1986; Crabtree, 2010). Besides bearing faults, in the context of EC motors, rotor faults are also frequent. Therefore we propose a feature set that describes these two groups of faults.

Proper selection of the feature set is crucial for the overall effectiveness of the quality assessment. When analyzing vibration signals generated under constant operating conditions, feature values are usually the amplitudes of particular spectral components.

Rotor faults Due to improper manufacturing or improper assembly, rotor faults include:

- mass unbalance, and

- misalignment faults.

The presence of either of the faults influences the mass displacement on the rotor, hence changing its moment of inertia. Under constant rotational speed, such a change can be detected by analysing the generated vibrations and it is generally expressed as an increase of the amplitudes of the spectral components at the rotational frequency f_{rot} and its higher harmonics $n \times f_{rot}$, $n \in \{2,3,\cdots\}$ (Xu and Marangoni, 1994).

Bearing faults Bearings in EC motors are the most susceptible element to mechanical faults. During the manufacturing process the most common causes for introducing bearing faults are improper bearing lubrication, improper mounting and alignment, as well as improper handling during the assembly process.

The detection of these faults is a challenging task. Vibrations, caused by a bearing fault, originate from impacts produced by the rolling elements hitting a damaged place. Each time a hit occurs, an excitation of system eigenmodes occurs in terms of an impulse response $s(t)$. The frequency of occurrence of these impulse responses can be estimated using the rotational speed f_{rot} of the rotating ring and the physical characteristics of the bearing, i.e. the pitch diameter D, the rolling element diameter d, the number of rolling elements Z, and the contact angle α (see Figure 9.2). Using these parameters the bearing fault frequencies can be calculated according to the relations shown in Table 9.1 (Tandon and Choudhury, 1999).

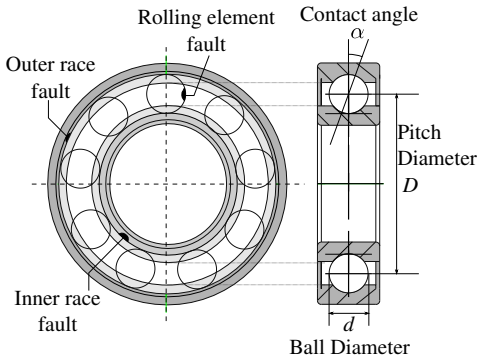


Figure 9.2: Bearing dimensions used for the calculation of the bearing's characteristic frequencies

Table 9.1: Bearing frequencies (Tandon and Choudhury, 1999)

Name	Relation to the rotational frequency f_{rot}
Bearing pass frequency inner race (BPFI)	$f_{BPFI} = \frac{Zf_{rot}}{2}\left(1+\frac{d}{D}cos\alpha\right)$
Bearing pass frequency outer race (BPFO)	$f_{BPFO} = \frac{Zf_{rot}}{2}\left(1-\frac{d}{D}cos\alpha\right)$
Fundamental train frequency (FTF)	$f_{FTF} = \frac{f_{rot}}{2}\left(1-\frac{d}{D}cos\alpha\right)$
Ball spin frequency (BSF)	$f_{BSF} = \frac{Df_{rot}}{2d}\left(1-\left(\frac{d}{D}cos\alpha\right)^2\right)$

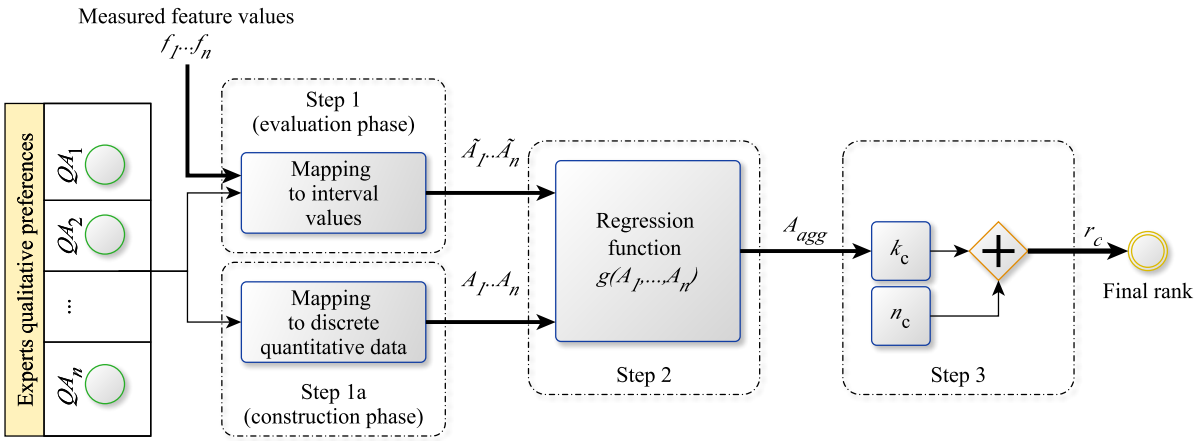


Figure 9.3: From qualitative attributes and quantitative features to final quantitative evaluation

9.1.2 Implementation Within the Quality Assessment System

The implementation of the copula-based quality assessment system follows the steps shown in Figure 9.3. Firstly, one has to present the decision maker's preferences in a qualitative decision table. Then, a mapping from qualitative attributes into quantitative ones has to be performed. Finally, for each quantitative decision table, a copula function is defined. After the completion of these steps, one can apply the Algorithms 6 and 7 for copula-based regression described in section 6.6.

DEX Hierarchical Model

Assessing the overall motor quality rank directly from the measured features is a rather difficult task. Therefore, the problem is transformed into a hierarchical decision making model in which the overall *mechanical quality* rank is obtained by aggregating two simpler attributes: *rotor quality* and *bearings quality* as shown in Table 9.2. The former attribute can be directly assessed from the measured features described in subsection 9.1.1. The latter attribute is still complex as it can be further decomposed into four simpler attributes: *inner ring quality*, *outer ring quality*, *quality of the rolling elements* and *quality of the bearing cage*. These four attributes can be assessed from the measured features describing bearing condition, as shown in section 9.1.1. Based on this logical structure a DEX hierarchical model is built, which is shown in the first column of Table 9.2, where the aggregated attributes are given with upper cases and the basic attributes are given in plain letters.

Following the expert's preferences and knowledge, each attribute in the proposed hierarchical structure was described or aggregated using the expert's defined scale with five qualitative values:

$$QC = \{not\ satisfactory, good, very\ good, excellent, top\}.$$

For instance, the aggregation of the basic attributes *BPFI* and $2\times BPFI$ into attribute *Inner ring* is given in the first three columns of Table 9.3. These aggregations may be interpreted as a set of **if-then** rules, for instance, the last row in Table 9.3 can be interpreted as follows:

if BPFI is Top and $2\times$ BPFI is Top **then**
  Inner ring is Top
end if.

Table 9.2: DEX model tree and qualitative and quantitative evaluations of EC motors 744 and 9

Attribute	Evaluation of motor 744		Evaluation of motor 9	
	Qualitative	Quantitative	Qualitative	Quantitative
MECHANICAL QUALITY	not satisfactory	1.3389	very good	3.3056
ROTOR QUALITY	top	4.9702	top	5.0302
f_{rot}	top	4.7851	top	5
$2 \times f_{rot}$	top	5	top	5
Variance	top	5	top	5
BEARINGS QUALITY	not satisfactory	0.7592	good	1.7899
INNER RING	good	1.9583	very good	2.8051
Bpfi	good	1.7096	very good	2.8227
2×Bpfi	very good	2.6319	excellent	2.8830
OUTER RING	good	1.8009	good	2.1889
Bpfo	good	1.9043	very good	3.0203
2×Bpfo	very good	2.1660	good	2.0238
ROLL ELEMENTS	not satisfactory	0.8810	very good	2.8454
Bsf	not satisfactory	1.2396	very good	2.9361
2×Bsf	very good	2.6100	excellent	2.9288
Ftf	good	2.6279	very good	3.0974

9.1.3 Qualitative to Quantitative Value Mapping

After defining the qualitative model, the next step is to obtain the quantitative model. For that reason, each of the qualitatively defined expert rules and preference values are mapped into a quantitative one. In the qualitative model, the preferences given by the decision maker are:

$$not\ satisf. \prec good \prec very\ good \prec excellent \prec top$$

where the sign $\prec$ stands for "is strictly less preferred than". In the quantitative model, these values are mapped into $\{1, 2, 3, 4, 5\}$ respectively. In addition, the sign $\prec$ is mapped into $<$, where $<$ stands for 'is greater than'. The mapping ensures that the more preferred values are mapped into greater numbers. An example of the mapping is given in the last three columns of Table 9.7, where the qualitative values of attributes *Bpfi*, $2\times$ *Bpfi* and *Inner ring* are mapped into quantitative values of A_1, A_2 and C respectively.

Based on the QQ model given in Table 9.2, six linear aggregation functions are built for each of the aggregated attributes. Each aggregation function is built from the quantitative decision table, such as the one given with the last three columns in Table 9.3. The aggregation functions are obtained by applying (3.1)–(3.5). For the given example in Table 9.3, function g in (3.1) reads:

$$g = 0.5600A_1 + 0.5600A_2 - 0.9200. \tag{9.1}$$

After applying (3.4)–(3.5) on (9.1) we obtain:

$$f_c = \begin{cases} 0.2976g + 0.6071, & \text{if } c = 1 \\ 0.5952g + 1.0476, & \text{if } c = 2 \\ 0.5952g + 1.3810, & \text{if } c = 3 \\ 0.5952g + 1.7143, & \text{if } c = 4 \\ 0.8929g + 0.8214, & \text{if } c = 5. \end{cases}$$

Here f_c is an estimation of C in Table 9.3. The obtained quantitative value f_c for the attribute *Inner ring* is then propagated to the aggregation function in the next level, which estimates the quantitative value of *Bearings quality*. This procedure is repeated until the final evaluation of the *Mechanical Quality* is achieved.

Table 9.3: Expert defined rules for aggregation of the attribute *Inner ring* and mapping from the qualitative attribute values into quantitative ones.

BPFI	**2xBPFI**	**Inner ring**	A_1	A_2	C
'not satisf.'	*'not satisf.'*	*'not satisf.'*	1	1	1
'not satisf.'	*'good'*	*'not satisf.'*	2	1	1
'not satisf.'	*'very good'*	*'not satisf.'*	1	3	1
'not satisf.'	*'excellent'*	*'not satisf.'*	1	4	1
'not satisf.'	*'top'*	*'not satisf.'*	1	5	1
'good'	*'not satisf.'*	*'not satisf.'*	2	1	1
'good'	*'good'*	*'good'*	2	2	2
'good'	*'very good'*	*'good'*	2	3	2
'good'	*'excellent'*	*'very good'*	2	4	3
'good'	*'top'*	*'very good'*	2	5	3
'very good'	*'not satisf.'*	*'not satisf.'*	3	1	1
'very good'	*'good'*	*'good'*	3	2	2
'very good'	*'very good'*	*'very good'*	3	3	3
'very good'	*'excellent'*	*'very good'*	3	4	3
'very good'	*'top'*	*'excellent'*	3	5	4
'excellent'	*'not satisf.'*	*'not satisf.'*	4	1	1
'excellent'	*'good'*	*'very good'*	4	2	3
'excellent'	*'very good'*	*'very good'*	4	3	3
'excellent'	*'excellent'*	*'excellent'*	4	4	4
'excellent'	*'top'*	*'excellent'*	4	5	4
'top'	*'not satisf.'*	*'not satisf.'*	5	1	1
'top'	*'good'*	*'very good'*	5	2	3
'top'	*'very good'*	*'excellent'*	5	3	4
'top'	*'excellent'*	*'excellent'*	5	4	4
'top'	*'top'*	*'top'*	5	5	5

9.1.4 Integration of Feature Values and Expert's Preferences

Experts' preferences may be subjective, even inconsistent, and may differ between experts. Therefore in the development of this model we paid special attention to the consistency of the evaluation model. First, whenever possible, we closely followed the ISO 10816 standard, which denotes the maximal allowed vibrations for particular drives. Second, additional requirements were specified by the manufacturer itself as well as the targeted customers. Finally, the DEX model was built which is guaranteed to be complete and consistent.

The actual integration of the measured feature values and the expert's preferences is performed using fuzzification. The expert's preference is towards motors with lower vibrations, hence lower feature values are more preferred. The maximum allowed value for each feature determines the limit for the *not satisfactory* grade. This limit was determined either by governing standard rules or by the company's

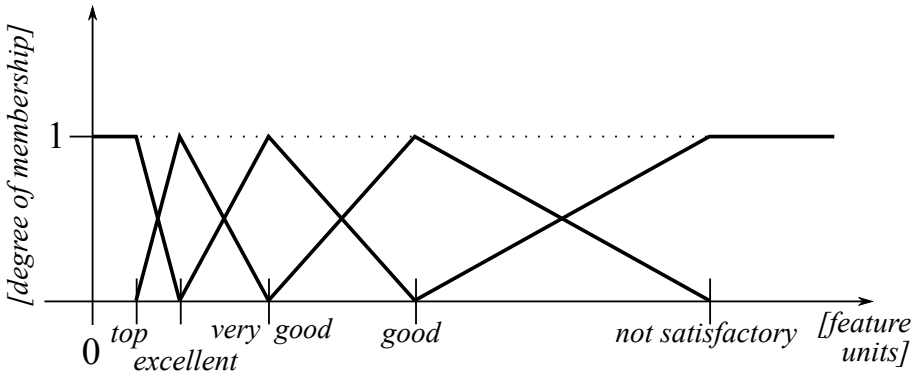


Figure 9.4: Intervals for mapping feature values to quantitative ones

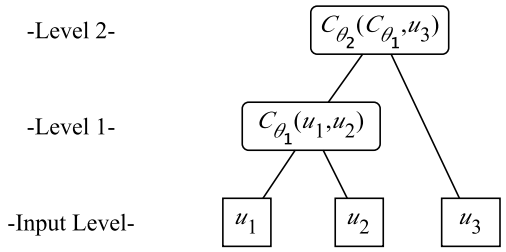


Figure 9.5: FNAC structure for aggregation of the attribute *Inner Ring*

quality requirements. The remaining interval below the limit for the *not satisfactory* grade was divided dyadically, as shown in Figure 9.4. Such mapping ensures more sensitivity at lower feature values. The feature values f_i are fuzzified and mapped accordingly into the expert's defined interval $[1,5]$, employing the relation:

$$\tilde{A}_i(f_i) = \sum_n \mu_n(f_i) c_n. \tag{9.2}$$

Here $\mu_n(f_i)$ is the membership function of the n^{th} rule as given in Figure 9.4, and $c_n \in \{1,2,3,4,5\}$ are values of the class attribute.

9.1.5 Constructing the Copula-Based Regression Functions

According to the model shown in Table 9.2 there are six aggregation tables. For each of the decision tables, there are two possibilities: to build FNACs or one-parametric copulas. Both approaches are given in continuation.

Constructing FNACs

For each decision table, a FNAC based on the Frank bi-variate copula was built. Hence six copula-based regression functions were derived. The obtained values from (9.2) enter the appropriate copula-based regression functions, and for each of them a copula-based regression value is calculated. For the given example in Table 9.3, we built a FNAC, such as the one given in Figure 9.5. When building the FNAC, the attributes enter the input level at any position as long as the obtained FNAC fulfills (5.16). For the attribute *Inner Ring*, in the obtained FNAC, first the attribute *BPFI* is coupled with the output *Inner ring* forming a copula C_{θ_1} with parameter $\theta_1 = 5.5$. The obtained copula is then coupled with the attribute

$2\times BPFI$ leading to the copula C_{θ_2} with parameter $\theta_2 = 2.5529$. The values of θ_1 and θ_2 fulfill (5.16) thus, we may proceed with the regression procedure. Here, the position of the dependent variable *Inner ring* is $p = 2$ hence we use Algorithm 7 for regression.

The regression part is solved as an iterative procedure starting from the highest level of FNAC and descending down to the level which contains the dependent variable by substituting the correct quantities for q in the Frank regression equation (see last column in Table 5.1). In each of the iterations, the value of q is substituted with a vector of regression values obtained from the previous iteration procedure. For example, in the first iteration the value of v_3 is obtained using Frank-based regression equation given in Table 5.1 that leads to:

$$v_3 = \frac{1}{2.5529} \log \frac{-e^{2.5529}(1-0.5+0.5e^{2.5529u_3})}{-e^{2.5529}+0.5e^{2.5529}-0.5e^{2.5529u_3}}. \tag{9.3}$$

The values obtained with (9.3) are propagated to the next iteration, the last one, that provides the final copula regression equation v:

$$v = \frac{1}{5.5} \log \frac{-e^{5.5}(1-v_3+v_3e^{5.5u_1})}{-e^{5.5}+v_3e^{5.5}-v_3e^{5.5u_1}}.$$

The regression values v are afterwards normalized in order to retain consistency with the qualitative model as defined in (3.4)–(3.5) (Stage 3 in Figure 3.2):

$$f_c = \begin{cases} 0.2667F^{-1}(v)+0.4752, & \text{if } c=1 \\ 0.5564F^{-1}(v)+1.0214, & \text{if } c=2 \\ 0.3659F^{-1}(v)+1.9901, & \text{if } c=3 \\ 0.4639F^{-1}(v)+2.3345, & \text{if } c=4 \\ 1.0834F^{-1}(v)+0.1608, & \text{if } c=5. \end{cases} \tag{9.4}$$

The obtained values from (9.4) are propagated in the higher hierarchical level, where they are used as inputs in the next regression function. The procedure is recursively repeated up to the topmost decision table. The result of the topmost decision table is regarded as the overall quality rank for each motor.

Constructing One-Parametric Copulas

In this approach, for each decision table, a one-parametric Clayton multi-variate copula was built. Hence six copula-based regression functions were derived. For each copula, only one parameter θ is obtained. For example, for the attribute *Mechanical quality*, the copula has parameter $\theta_1 = 0.2953$.

The regression is performed in one step by using the regression equation for the Clayton copula given in Table 5.1:

$$v = (1-u^{-0.2953}+(0.5u^{1.2953})^{-\frac{0.2953}{1.2953}})^{-\frac{1}{0.2953}}.$$

In the last equation, the values of u are obtained by building one-parametric copula with $\theta_1 = 0.2953$ with all input attributes. This is possible due to the associative property of the Archimedean copulas: $C(C(u_1,u_2),u_3) = C(u_1,C(u_2,u_3))$. The regression values v are afterwards normalized in order to retain consistency with the qualitative model as defined in (3.4)–(3.5) (Stage 3 in Figure 3.2):

$$f_c = \begin{cases} 0.2801F^{-1}(v)+0.9049, & \text{if } c=1 \\ 1.6325F^{-1}(v)-1.1857, & \text{if } c=2 \\ 1.8293F^{-1}(v)-1.1183, & \text{if } c=3 \\ 2.7279F^{-1}(v)-2.9315, & \text{if } c=4 \\ 5.7322F^{-1}(v)-10.6250, & \text{if } c=5. \end{cases}$$

9.1.6 The Final Evaluation of EC Motors and Two Characteristic Examples

The Assessment Rig

Each EC motor is tested using the assessment rig shown in Figure 9.6. The rig consists of a fixed pedestal on top of which a metal disk is positioned. The metal disk holds three rubber dampers that suspend the tested EC motor. The experiment starts by positioning the EC motor vertically on the rubber dampers in such a way that the drive-end bearing is on the bottom. Afterwards, two accelerometers are positioned on the motor housing nearest to the both bearings. The test-rig minimizes the environmental influence, hence guaranteeing sufficiently constant experimental conditions.

The data acquisition process commences as soon as the nominal rotational speed is reached. Firstly, both vibration signals are low-pass filtered with cut-off frequency at 22 kHz. Afterwards, both signals are sampled at 60 kHz. During the whole data acquisition process the nominal rotational speed of $f_{rot} =$ 38 Hz is maintained. Each acquisition process lasts 8 seconds. After finishing the acquisition the motor is decelerated down to the stop position.

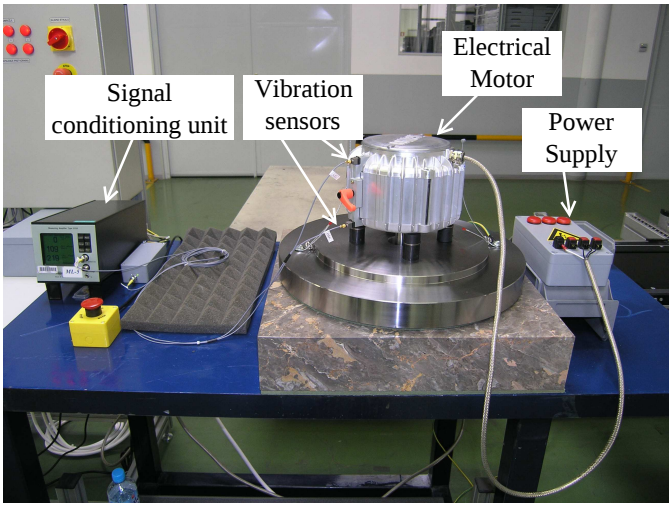


Figure 9.6: The prototype assessment point

Results on a Test Batch of Motors

During the evaluation process a test batch of 840 EC motors was analyzed. The overall quality rank is shown in Figure 9.7. For the purpose of testing the system on small differences in data, during the initial start-up of the line, motors with intentionally introduced various mechanical faults were evaluated. Consequently, there are many motors with different overall quality rank.

From the results shown in Figures 9.7 and 9.8, it is clearly visible that the quality ranks of the tested motors are spread over the interval $[0.5, 5.5]$. This is an indication that the proposed copula-based DSS is highly sensitive even to minor variations in the motor quality. Unlike methods that use weighted utility functions, where options with *not satisfactory* feature values are ranked highly, this approach averts such performance. Such example is given in (Boškoski et al., 2011), where the first 130 motors were evaluated using evidential reasoning approach which lead to cases where final evaluations were inconsistent with the expert's preferences. Besides the overall quantitative evaluation, the calculated rank also shows the class of each EC motor.

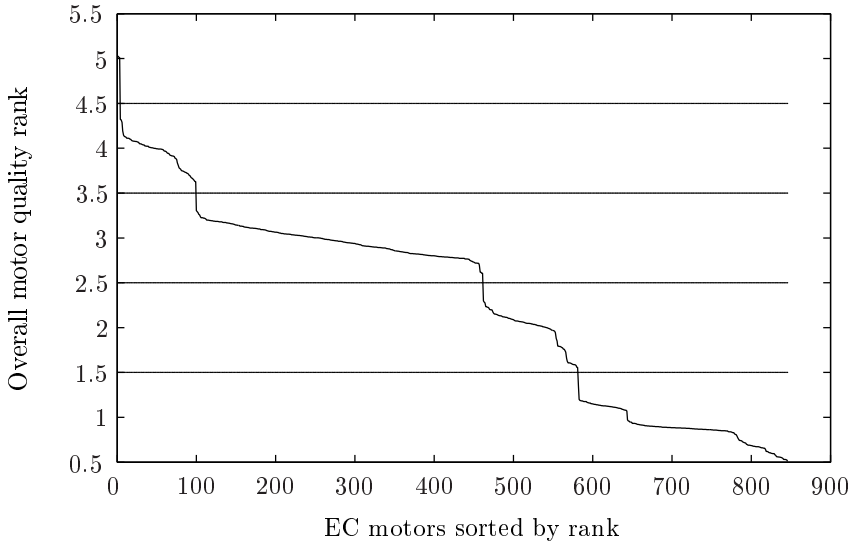


Figure 9.7: Rankings obtained with Frank FNAC

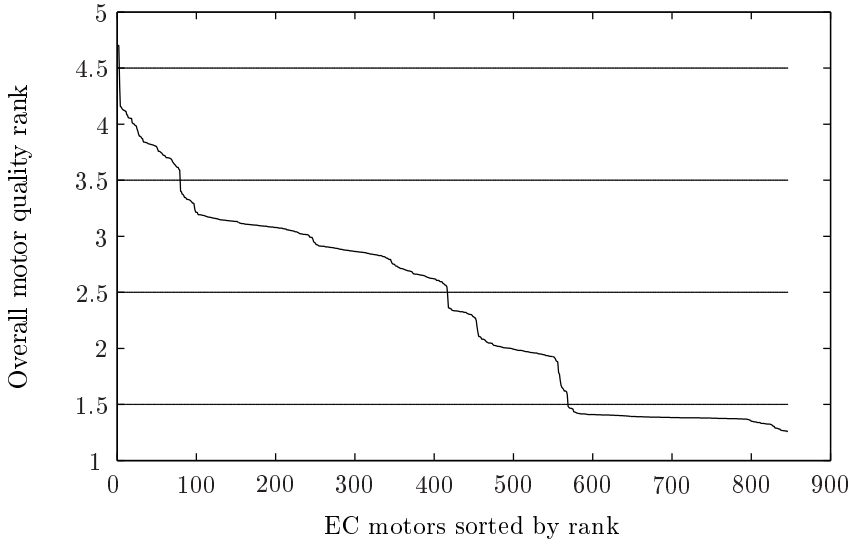


Figure 9.8: Rankings obtained with one-parametric Clayton copula

Detailed Analysis of Two Characteristic Examples

The effectiveness of the proposed copula-based DSS can be best illustrated by detailed analysis of two characteristic cases. One case refers to the dominance of the *not satisfactory* grade and the other on ranking of motors whose quality belongs between two adjacent grades.

The First Case From the expert's preferences (see Table 9.3) it is clearly visible that if any of the attributes acquires a value that belongs to the grade *not satisfactory*, the examined motor will obtain overall qualitative rank that belongs to the lowest class. The 744^{th} EC motor is the case with the highest quality rank from the class of *not satisfactory* motors. The measured features for this motor are given in Table 9.2.

From Table 9.2 it is visible that the rotor quality belongs to the highest grade, denoted as *top*. Namely, the values of f_{rot}, $2 \times f_{rot}$ and *Variance* belong to the interval $[5 \pm 0.5]$, where the first value denotes the

class *top*. Additionally, one may notice that bearing features mostly belong to the qualitative class *good*, for instance Ftf, $Bpfo$ and $Bpfi$ which have values in the interval $[2 \pm 0.5]$, and *very good*, such as $2 \times Bpfo$, $2 \times Bpfi$ and $2 \times Bsf$ that have values in the interval $[3 \pm 0.5]$. Unlike them one feature describing the condition of the rolling element Bsf has a value that belongs to the interval $[1 \pm 0.5]$, which is a quantitative employment of the *not satisfactory* class. This qualitative value is propagated through all levels of the hiearhical structure of the model, hence leading to *not satisfactory* evaluation of the higher level aggregated attributes *Roll elements, Bearing quality* and final qualitative evaluation of *Mechanical quality*. Consequently, the overall quality rank is just 1.333, which clearly states that the particular EC motor is of *not satisfactory* quality.

The second case The second case is the 9$^{\text{th}}$ EC motor, whose overall quality rank is 3.3056, which belongs to the qualitative class *very good*. The calculated features for this EC motor are given in the last column in Table 9.2.

According to the measured features, the bearing quality of the 9$^{\text{th}}$ motor can be easily graded as *very good*, since most of the features have value from the interval spanned by this grade. Still, the qualitative value of the attribute $2\times$*Bpfo* is *good*, and this value is propagated up in the hierarchy leading to qualitative evaluation of *Bearing Quality* to *good*, as defined by the expert's preferences. Rotor features, on the other hand, undoubtedly state that this particular case has *top* quality of the rotor. Consequently, the overall motor quality is *very good*, however, the numerical rank suggests that the quality is very close to the next higher class *excellent*. These examples show that the hierarchy of attributes aids the process of integration of expert's preferences into the final quality evaluation of motors.

Differences in rankings between linear and copula-based models Additionally, we have examined the differences in rankings for different values of $q \in \{0.1, 0.25, 0.75, 0.9\}$ and QQ, and compared it with the rankings of the model built with $q = 0.5$, which are presented in Figure 9.9. It may be noticed that ranking variations in the different copula-based models are substantially smaller than the ranking variations obtained by QQ. QQ fails to distinguish among 29 pairs of motors, which it ranked the same. Therefore this ranking is considered less appropriate than the copula-based method.

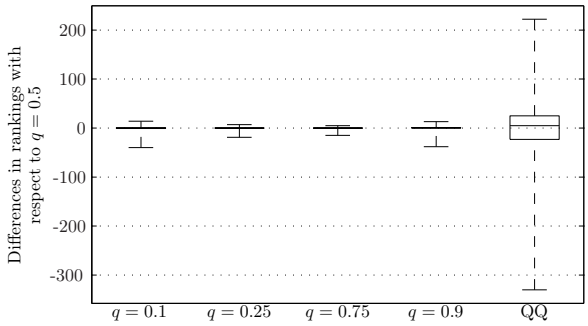


Figure 9.9: Differences in rankings between models built with different quantiles q and QQ, and the selected model built with $q = 0.5$

9.1.7 Discussion

The quality ranking of EC motors was regarded as a hierarchical decision making task, in which the final motor's quality is aggregated from the quality of its components. The proposed solution is a copula-based decision support system. The input of the system is a set of measured features calculated from the acquired vibrations generated by the examined EC motor. Furthermore, the system employes available

expert's knowledge condensed in DEX qualitative decision table. Employing copula-based regression functions resulted in a full quality ranking of EC motors. The system was evaluated on a batch of 840 motors.

The merging of expert's knowledge with DEX, and employment of copula-based regression leads to a final evaluation system with four properties. First, the qualitative evaluation of each EC motor provides easily understandable quality description. Second, the system has ability of distinguishing small variations of the input features. Therefore, each EC motor is assigned a quantitative value, leading to distinct evaluation of all EC motors in the test batch. Third, the hierarchical decomposition of the problem gives explanation how the qualities of each of the lower level components lead to the final evaluation. Therefore, besides the process of quality assessment, such a system can be seamlessly employed as a fault detection module that is able to perform fault evaluation too. Finally, the proposed evaluation system for EC motors leads to rankings that are fully in compliance with the decision maker's (or expert's) preferences and the required regulations.

The output of the proposed quality assessment system was employed for deciding whether the produced EC motor satisfies the proposed quality requirements. Besides this application, there are two more immediate possibilities for its usage. First, the output of this system can also be used for monitoring and supervisory control of the production process, i.e. it can be used as input for any subsequent manufacturing execution system. Second, the same concepts applied for the implementation of the quality assessment system can be employed for on-line condition monitoring of running motors. The overall quality assessment as well as the intermediate outputs can be used as features for estimating the motor's remaining useful life.

9.2 Assessment of Workflows

In this section a one-parametric copula-based regression model and a QQ-based model are proposed for assessment of data-mining workflows (DW). The DEX model for the DW has been developed in the frame of the FP7 STREP project e-LICO (eLICO, 2012). In this section two approaches for modification of the second stage of QQ are examined. In the first one, different functions are investigated for the estimation of the weights w_i in (3.1) based on the impurity functions (4.4)–(4.6) defined in Section 4.1. In the second approach one-parametric copula-based regression functions are employed in order to rank the different DW.

9.2.1 Data-Mining Workflow Assessment

The aim of the DW assessment is to provide a set of measures that describe individual workflows (Žnidaršič et al., 2011). An example of a DW is given in Figure 9.10 (Antulov, 2011; Bošnjak et al., 2011). The workflows may be described with different attributes such as the number of nodes, number of edges, number of paths, density of paths etc. These attributes were used to build a qualitative multi-attribute evaluation model using DEXi. The model tree and the preferential values of the attributes are shown in Table 9.4. In the model in Table 9.4, the basic attributes are given in plain letters, while the aggregated attributes are given with upper case letters. The attributes' left-most and right-most values are the least and most preferred ones, respectively. Each aggregated attribute has an associated utility function such as the one given for the attribute *Robustness* in Table 9.7, in columns *A*–*D*. For the purpose of performing regression for option ranking, these qualitative utilities are mapped into quantitative ones, as shown in columns A_1–D_1.

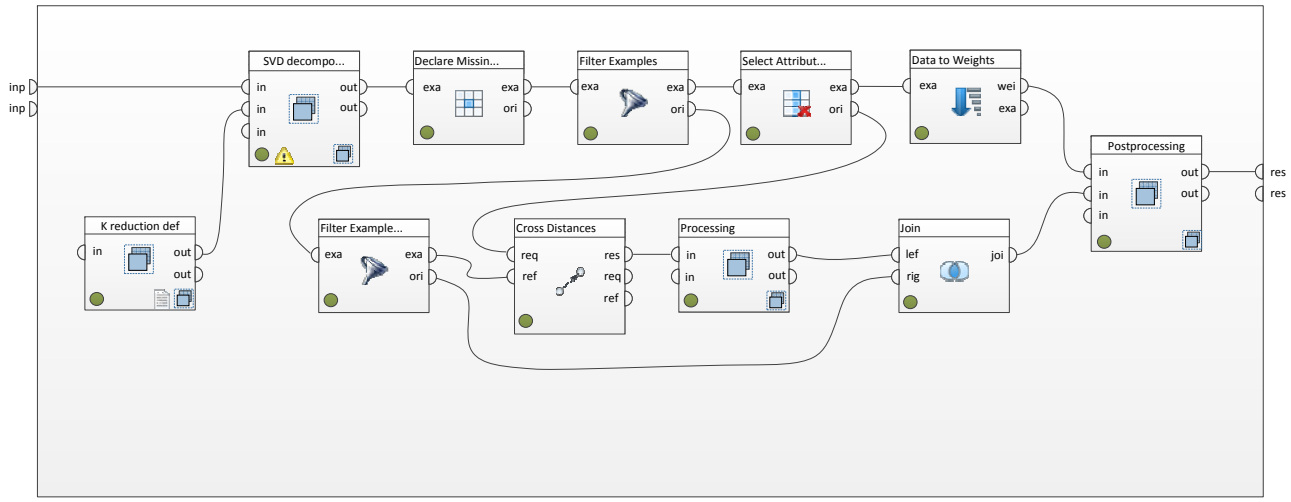


Figure 9.10: Example of a workflow for a collaborative filtering recommender system (Antulov, 2011; Bošnjak et al., 2011)

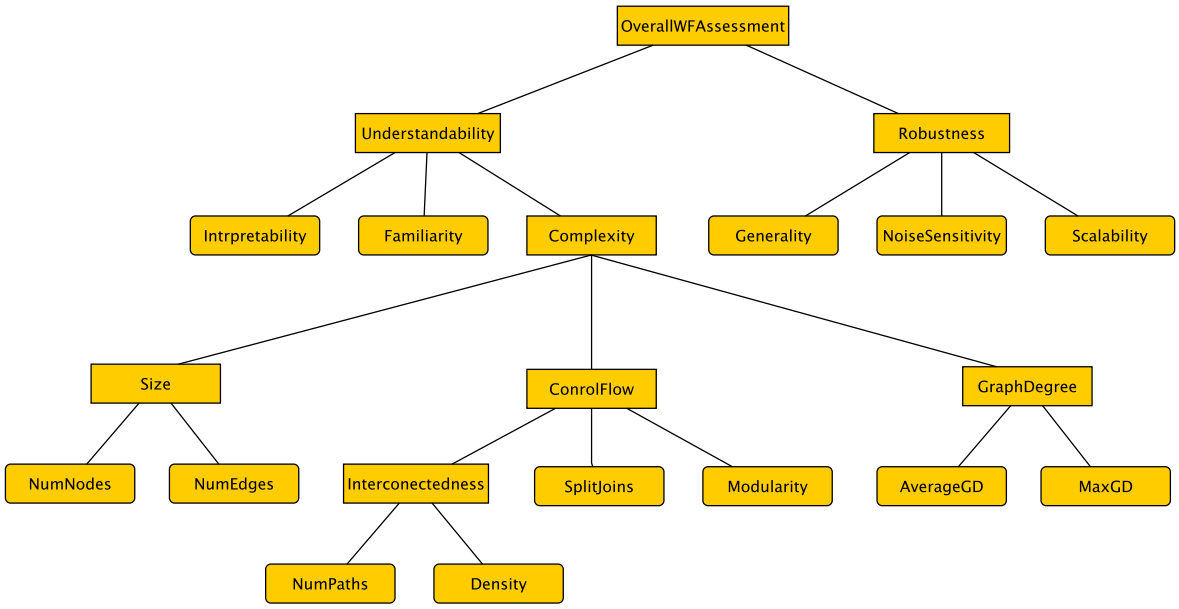


Figure 9.11: Example of DEX hierarchical tree for assessment of data-mining workflows

9.2.2 Models for Ranking DW Options

To provide option ranking within each class, different linear regression functions were developed using the QQ method and its modifications for weights estimation in (3.1) based on the impurity functions (4.4)–(4.6). Firstly, the qualitative values are mapped into quantitative one, using a mapping function which preserves the order of the qualitative attributes' values. Such example is given in Table 9.7, where the qualitative values of the attributes A, B, C and D are mapped into quantitative values A_1, B_1, C_1 and D_1. The preferential values for the attributes A and B are $\{low, medium, high\}$, where the value of *high* is most preferred, in contrast to the the value of *low* which is least preferred. The qualitative values $\{low, medium, high\}$ are mapped into the ordered numbers $\{1,2,3\}$, respectively. On the other hand, the preferential values for attribute C are $\{high, medium, low\}$, where

Table 9.4: DEXi model tree and attribute scales for assessment of data-mining workflows

Attribute	Scale
OVERALL ASSESSMENT	bad, acceptable, appropriate, good, *excellent*
├─ UNDERSTANDABILITY	very low, low, medium, high, *very high*
│ ├─ Interpretability	low, medium, *high*
│ ├─ Familiarity	low, medium, *high*
│ └─ COMPLEXITY	very high, high, medium, low, *very low*
│ ├─ SIZE	high, medium, *low*
│ │ ├─ NumNodes	high, medium, *low*
│ │ └─ NumEdges	high, medium, *low*
│ ├─ CONTROL FLOW	bad, acceptable, *good*
│ │ ├─ INTERCONNECTEDNESS	high, medium, *low*
│ │ │ ├─ NumPaths	many, some, *few*
│ │ │ └─ Density	high, medium, *low*
│ │ ├─ SplitJoints	high, medium, *low*
│ │ └─ Modularity	poor, *good*
│ └─ GRAPH DEGREE	high, medium, *low*
│ ├─ AverageGD	high, medium, *low*
│ └─ MaxGD	high, medium, *low*
└─ ROBUSTNESS	very low, low, medium, high, *very high*
├─ Generality	low, medium, *high*
├─ NoiseSensitivity	high, medium, *low*
└─ Scalability	low, medium, *high*

the value of *low* is most preferred, while the value of *high* is least preferred. Therefore, the values $\{high, medium, low\}$ are mapped into the ordinal numbers $\{1,2,3\}$, respectively, hence ensuring that preference order is kept in the quantitative space. In the same manner, the preferential values of the attribute D, $\{very\ low, low, medium, high, very\ high\}$, are mapped into $\{1,2,3,4,5\}$, respectively.

9.2.3 Experimental Evaluation

For each aggregated attribute given in Table 9.4, a proper quantitative model is defined. For example, the qualitative model of the attribute *Size* given in Table 9.5 is maped into a quantitative one presented in the first three columns in the Table 9.6. The quantitative model is used to develop six linear regression functions based on the linear QQ method and a one-parametric copula-based regression functions. An example of the ranking with QQ and the Frank one-parametric copula is given in the last two columns of the Table 9.6, denoted as *QQ*, for the ranking with QQ and *FC* for the ranking with the Frank copula. The obtained numbers show that the copula-based method provides six different ranking levels, unlike QQ, which provides five different ranking levels. All of the linear regression methods based on QQ provide the same number of ranking levels for options in the symmetric decision tables.

Examples of evaluating different workflows using Frank copula Four DW are presented to demonstrate the copula-based evaluation of different workflows. All of them are qualitatively ranked as 'good' by the model given in Table 9.4. The first two DWs, DW1 and DW2, are differently evaluated on several attributes: *Familiarity*, *COMPLEXITY*, *SIZE*, *NumNodes*, *NumEdges*, *Generality*, *NoiseSensitivity* and *Scalability*. The quantitative values of the attributes of the two workflows, DW1 and DW2, are

Table 9.5: Aggregation of the attribute **Size**

No.	NumNodes	NumEdges	Size
1	high	high	high
2	high	medium	high
3	high	low	medium
4	medium	high	high
5	medium	medium	medium
6	medium	low	low
7	low	high	medium
8	low	medium	low
9	low	low	low

Table 9.6: Quantitative mapping of the attribute **Size**

NumNodes	NumEdges	Size	QQ	FC
1	1	1	0.8333	0.6852
1	2	1	1.1667	0.9191
1	3	2	2.0000	1.7680
2	1	1	1.1667	0.9191
2	2	2	2.0000	2.1193
2	3	3	2.8333	2.7631
3	1	2	2.0000	1.7680
3	2	3	2.8333	2.7631
3	3	3	3.1667	3.1561

given in Table 9.8, where basic attributes are given with plain letters, while the aggregated attributes are given with upper case letters. Their copula-based regression values using one-parametric Frank copula are given in Table 9.9. In the Table 9.9 all values are in the intervals $c \pm 0.5$, hence one may use them for ranking of options. For example, the value of the attribute *OveralWFAss* for both options DW1 and DW2 in Table 9.8 is 4, however DW1 has greater or equal values for all attributes except $NoiseSensitivity$. The difference is reflected in the regression values for *Size*, *Robustness* and *OverAllAssessment* given in the columns DW1 and DW2 in Table 9.9. The difference is propagated up to the topmost attribute *OveralWFAss* resulting in greater ranking of DW1 than DW2.

The other two workflows, 'Mod 1 of DW1' and 'Mod 2 of DW1', represent modifications of DW1. They show how minor changes in some of the attributes affect the overall evaluation of the model. Therefore two modifications of the aggregated attribute *Robustness* of DW1 are investigated, by changing the values of the basic attributes $Generality$ and $NoiseSensitivity$. The results are given in the last two columns 'Mod 1 of DW1' and 'Mod 2 of DW1' in the Table 9.9. In the first modification, the values of the two attributes $NoiseSensitivity$ and $Generality$ change. The value of the *OverAllAssesment* increases leading to better rank than DW1. In 'Mod 2 of DW1', the values of $Generality$ and $NoiseSensitivity$ are exchanged with those in DW1. The final evaluation is given in Table 9.9. Results show that the evaluation of DW1 and 'Mod 2 of DW1' remain the same, due to the change in the symmetric attribute.

These results demonstrate two things. Firstly, one-parametric copula-based methods may provide more ranking levels than the linear models. Secondly, the small changes in the basic attributes lead to different copula-based regression values, and to different rankings within the same class. At the same time, the symmetric property of the decision tables is kept.

Table 9.7: Utility function (A-D) and mapping from the qualitative attributes into quantitative ones (A_1 - D_1). Basic attributes are: A - *Generality*, B - *Scalability* and C - *NoiseSensitivity*. The aggregated attribute is D - *Robustness*.

No	**A**	**B**	**C**	**D**	A_1	B_1	C_1	D_1
1	'low'	'low'	'high'	'very low	1	1	1	1
2	'low'	'med'	'high'	'low'	1	2	1	2
3	'low'	'high'	'high'	'low'	1	3	1	2
4	'low'	'low'	'med'	'low'	1	1	2	2
5	'low'	'med'	'med'	'low'	1	2	2	2
6	'low'	'high'	'med'	'med'	1	3	2	3
7	'low'	'low'	'low'	'low'	1	1	3	2
8	'low'	'med'	'low'	'med'	1	2	3	3
9	'low'	'high'	'low'	'med'	1	3	3	3
10	'med'	'low'	'high'	'low'	2	1	1	2
11	'med'	'med'	'high'	'low'	2	2	1	2
12	'med'	'high'	'high'	'med'	2	3	1	3
13	'med'	'low'	'med'	'low'	2	1	2	2
14	'med'	'med'	'med'	'med'	2	2	2	3
15	'med'	'high'	'med'	'med'	2	3	2	3
16	'med'	'low'	'low'	'med'	2	1	3	3
17	'med'	'med'	'low'	'med'	2	2	3	3
18	'med'	'high'	'low'	'high'	2	3	3	4
19	'high'	'low'	'high'	'low'	3	1	1	2
20	'high'	'med'	'high'	'med'	3	2	1	3
21	'high'	'high'	'high'	'med'	3	3	1	3
22	'high'	'low'	'med'	'med'	3	1	2	3
23	'high'	'med'	'med'	'med'	3	2	2	3
24	'high'	'high'	'med'	'high'	3	3	2	4
25	'high'	'low'	'low'	'med'	3	1	3	3
26	'high'	'med'	'low'	'high'	3	2	3	4
27	'high'	'high'	'low'	'very high'	3	3	3	5

Table 9.8: Example of workflow options

Attributes	DW 1	DW 2	Mod 1 of DW 1	Mod 2 of DW 1
OVERALL ASSESSMENT	4	4	4	4
UNDERSTANDABILITY	3	3	3	3
Interpretability	1	1	1	1
Familiarity	3	2	3	3
COMPLEXITY	4	3	4	4
SIZE	3	1	3	3
NumNodes	2	1	2	2
NumEdges	3	2	3	3
CONTROL FLOW	3	3	3	3
INTERCONECTEDNESS	3	3	3	3
NumPaths	2	2	2	2
Density	3	3	3	3
SplitsJoins	2	2	2	2
Modularity	2	2	2	2
GRAPH DEGREE	2	2	2	2
AverageGD	2	2	2	2
MaxGD	2	2	2	2
ROBUSTNESS	3	3	3	3
Generality	3	2	2	1
NoiseSensitivity	1	2	2	3
Scalability	2	3	2	2

Table 9.9: Evaluation of different workflows using one-parametric Frank copula

Attributes	DW 1	DW 2	Mod 1 of DW 1	Mod 2 of DW 1
OVERALL ASSESSMENT	3.7088	3.6266	3.7165	3.7088
UNDERSTANDABILITY	2.6103	1.6224	2.6103	2.6103
COMPLEXITY	3.6024	2.5797	3.6024	3.6024
SIZE	2.7631	0.9192	2.7631	2.7631
CONTROL FLOW	2.6222	2.6222	2.6222	2.6222
INTERCONECTEDNESS	2.7433	2.7433	2.7433	2.7433
GRAPH DEGREE	2.0082	2.0082	2.0082	2.0082
ROBUSTNESS	2.6020	2.7727	2.6943	2.6020

10 Conclusions

10.1 Contributions of the thesis

This thesis addresses the problem of full, monotonic and consistent ranking of a set of qualitative multi-attribute options which are derived with the DEX methodology. The existing QQ algorithm, developed for this purpose, is based on the assumption that when qualitative data are suitably mapped into discrete quantitative ones, they form monotone or nearly linear functions. The main limitation of QQ is that in many cases it fails to model non-linear functions. Consequently, the main goal of this thesis was to present a methodology that overcomes this limitation.

The main novelty of the work in this thesis can be associated with three most relevant results:

1. We have proposed four different and novel QQ-based methods for estimating a regression function, including the use of impurity functions for weights estimation in linear regression functions, the use of polynomial functions for regression, the use of optimization techniques for providing a regression function. The main focus is on the fourth method, i.e., hierarchical copula-based constructions for regression.

2. We have experimentally evaluated and compared the proposed methods.

3. We have applied the proposed copula-based solution on two real case examples.

The search for a regression function in all four proposed methods initially exploits the existing QQ method, which maps the qualitative options into discrete quantitative ones and then employes linear regression with least squares for ranking. In the first proposal, QQ is modified by introducing impurity functions for weights estimation in the linear regression function: χ^2, information gain and three estimators of the Gini index (the Gini covariance, the Gini population and the Gini Breiman). The second proposal introduces polynomial functions by employing the CIPER and the New CIPER algorithms to heuristically search for the best polynomial for ranking. The third proposal redefines the problem as a constraint optimization problem. It tries to find a function that fulfills the constraints arising from the requirement of consistent ranking of options. The final method proposes a replacement of the linear functions in QQ with copula-based functions. This approach leads to the usage of FNACs and PNACs built with three kinds of bi-variate Archimedean copulas: Frank, Clayton and Gumbel. The usage of bi-variate copulas for ranking led to the following novelties:

- We have developed an algorithm for using simple bi-variate copulas as aggregation functions for ranking of multi-attribute options;

- we have derived quantile regression equations for the FNAC and PNACs, which allow positioning of the dependent variable at arbitrary leaf position in FNAC and PNACs;

- we have developed an algorithm for using the copula-based method in hierarchically connected decision tables by introducing censoring.

An important advantage of the proposed copula-based method is the use of distribution functions of attributes in the modeling process, instead of the attribute values. This makes the method applicable to attributes with different scale units (for example kilometers, seconds, intervals), or for scale-free attributes. Furthermore, the introduction of copulas for solving the task of full ranking of discrete options does not limit the method to only one type of variables. It can be used without modification for continuos, probabilistic or combined attributes.

Having defined the four modifications of QQ, the next step was their evaluation. To achieve the evaluation and comparison of all methods, three artificially generated data sets of decision tables were designed. To assess the performance of each of the methods, monotonicity (2.3), full ranking (2.4) and consistency (2.5) were examined for each decision table. The experiments lead to the conclusion that the Gini population measure provides results fastest and solves most of the cases. However, there are several advantages of copulas that were evident from the simulations:

- Copula functions can provide more ranking levels for options in symmetric or partially symmetric decision tables. These are cases where QQ methods provide the same weight value for the symmetric attributes.

- Unlike all other examined methods, which provide only one solution of full rankings, copulas can provide all different possible combinations of allowed rankings, due to the different FNACs and PNACs that may be built.

The applicability of the copula-based method is presented on two real cases of hierarchical models. The results show that the method performs well for modeling non-linear qualitative decision tables.

Finally, the copula-based algorithms were implemented in MATLAB. MATLAB provides functions only for the basic calculations of bi-variate copula. To be able to perform the experiments, a toolbox was developed which covers building of FNACs, PNACs and regression with hierarchical copulas on a hierarchical setting of decision tables. Another toolbox was developed for performing regression with impurity functions for a hierarchical setting of decision tables and several functions were developed to automatically define constrains, i.e., possible objective functions for a given decision table in the algorithm that uses optimization for providing a regression function.

We can thus conclude that modeling with copulas is a good choise for aggregation of the numerical utilities of qualitative decision options.

10.2 Future Work

The presented results which cover the copula-based method for option ranking raise the following challenges:

1. The first one is to include different types of mappings from the qualitative to the quantitative space, and investigate the sensitivity of the copula-based method when the mapping changes;

2. The second is to investigate the usage of different copula families for option ranking;

3. Probably the biggest challenge would be to determine patterns for mixing different copula families in the FNACs and PNACs which would guarantee monotonicity of the constructions and provide full ranking of options simultaneously.

Other research directions for future work arise from the optimization approach, where improvement of results may be expected with:

1. More careful design of the objective function (for example minimization of the error between the estimated class attribute and the real class value) and/or
2. Relaxing the constraints, i.e., define constraints for comparable options so that the monotonicity holds;
3. Introducing the third stage of QQ to ensure consistence between QDT and DT.

11 Acknowledgements

I would first like to thank my mentor, Prof. Dr. Marko Bohanec, who gave me the freedom to explore, and yet kept reminding me to focus on the problems appreciated in the research community.

I wish to express my gratitude to the committee members, Prof. Dr. Sašo Džeroski, Prof. Dr. Vladislav Rajkovič and Asst. Prof. Dr. Martin Žnidaršič, for their valuable comments and remarks.

Having good empirical data at hand was important. Prof. Dr. Đani Juričić provided the EC motors data. Prof. Dr. Marko Bohanec provided the data-mining workflow data.

I would like to thank all E8 members for providing a cosy atmosphere to work. In particular, I would like to thank Dr. Aleksandar Pečkov for providing me the software New CIPER, and instructions on how to use it. I would like to express my gratitude to my office colleagues and friends, Nejc and Dragana. I thank Nejc for his questions and discussions about copulas. I thank Dragana for becoming my best friend here, and for making the student days easier at work, after work, during weekends and while cycling on the slovenian mountains.

I would like to thank the sweetest person in my entire life, my daughter Jana, whose lovely smile was the source of my strength to finish this PhD.

I thank my parents for supporting me to give up the comfort that I enjoyed in my country and to explore the unknown. I thank my mother for her loving care during my sickness, and to my father for his absolute understanding during our talks.

Last but not least, I thank my husband Pavle, for his patient questions about my progress, for the long discussions of my problems of all kinds, and for making my dreams come true always in the last 14 years. Most of all, I thank him for his persistent belief in me, despite the whole toil and stress.

Finally, I would like to acknowledge the support of the Ad Futura Program of the Slovene Human Resources and Scholarship Fund and the support of the Slovenian Research Agency through the Research Program J2-2353 and the Project P2-0103.

12 References

Albrecht, P. F.; Appiarius, J. C.; Shrama, D. K. Assessment of the reliability of motors in utility applications. *IEEE Transactions of Energy Conversion* **EC-1**, 39–46 (1986).

Angilella, S.; Greco, S.; Lamantia, F.; Matarazzo, B. Assessing non-additive utility for multicriteria decision aid. *European Journal of Operational Research* **158**, 734–744 (2004).

Antoni, J. The spectral kurtosis: application to the vibratory surveillance and diagnostics of rotating machines. *Mechanical Systems and Signal Processing* **20**, 308–331 (2006).

Antulov, N. Example of a workflow for a svd user-based collaborative filtering recommender system. http://www.myexperiment.org/workflows/2109.html. (access: 26.12.2012).

Bana e Costa, C.; De Corte, J.; Vansnick, J. MACBETH. In: Meskens, N.; Roubens, M. (eds.) *Advances in Decision Analysis, Book Series: Mathematical Modelling: Theory and Applications*. 131–157, Kluwer Academic Publishers (1999).

Baracskai, Z.; Dörfler, V. Automated fuzzy-clustering for doctus expert system. In: *Paper presented at International Conference on Computational Cybernetics , Siófok, Hungary* (2003).

Beliakov, G.; Pradera, A.; Calvo, T. *Aggregation Functions: A Guide for Practitioners* (Springer-Verlag Berlin Heidelberg, 2007).

Berg, D.; Aas, K. Models for construction of multivariate dependance: A comparison study. *European Journal of Finance* **15**, 639–659 (2009).

Beuthe, M.; Scannella, G. Comparative analysis of uta multicriteria methods. *European Journal of Operational Research* 246 – 262 (2001).

Bohanec, M. *Odločanje in modeli* (DMFA, Ljubljana, 2006).

Bohanec, M. *DEXi: Program for Multi-Attribute Decision Making: User's manual : version 4.00*. IJS Report DP-11340, Jožef Stefan Institute, Ljubljana (2013).

Bohanec, M.; Rajkovič, V. DEX: An expert system shell for decision support. *Sistemica* **1**, 145–157 (1990).

Bohanec, M.; Rajkovič, V.; Bratko, I.; Zupan, B.; Žnidaršič, M. Dex methodology: Thirty three years of qualitative multi-attribute modelling. In: *Proceedings of the 15th International Conference Information Society IS 2012*. 31–34 (2012).

Bohanec, M.; Urh, B.; Rajkovič, V. Evaluation of options by combined qualitative and quantitative methods. *Acta Psychologica* **80**, 67–89 (1992).

Bohla, E.; Lancaster, P. Implementation of a markov model for phylogenetic trees. *Journal of theoretical Biology* **239**, 324–333 (2006).

Bouyé, E.; Salmon, M. Dynamic copula quantile regressions and tail area dynamic dependence in forex markets. *European Journal Of Finance* **15**, 721–750 (2002).

Boškoski, P.; Petrovčič, J.; Musizza, B.; Juričić, Đ. Detection of lubrication starved bearings in electrical motors by means of vibration analysis. *Tribology International* **43**, 1683 – 1692 (2010).

Boškoski, P.; Petrovčič, J.; Musizza, B.; Juričić, Đ. An end-quality assessment system for electronically commutated motors based on evidential reasoning. *Expert Systems with Applications* **38**, 13816 – 13826 (2011).

Bošnjak, M.; Antulov-Fantulin, N.; Šmuc, T.; Gamberger, D. Constructing recommender systems workflow templates in rapidminer. In: *Proceedings of the RapidMiner Community Meeting And Conference (RCOMM), Dublin, Ireland* (2011).

Bowman, A. W.; Azzalini, A. *Applied Smoothing Techniques for Data Analysis. The Kernel Approach with S-Plus Illustrations* (Oxford University Press, 1997).

Breiman, L.; Friedman, J. H.; Olshen, R. A.; Stone, C. J. *Classification and Regression Trees*. *Statistics/Probability Series* (Wadsworth Publishing Company, Belmont, California, U.S.A., 1984).

Brent, R. *Algorithms for Minimization without Derivatives* (Prentice-Hall, Englewood Cliffs, New Jersey, 1993).

Brown, L.; Cai, T.; Zhang, R.; Zhao, L.; Zhou, H. A root-unroot transform and wavelet block thresholding approach to adaptive density estimation. *Unpublished* (2005).

Chen, C. An introduction to quantile regression and the quantreg procedure. http://www2.sas.com/proceedings/sugi30/213-30.pdf. (access: 1.11.2013).

Corrente, S.; Greco, S.; Slowinski, R. Multiple criteria hierarchy process in robust ordinal regression. *Decision Support Systems* **53**, 660 – 674 (2012).

Crabtree, C. J. Survey of commercially available condition monitoring systems for wind turbines. *Technical Report*, Durham University, School of Engineering and Computing Science (2010).

Despa, S. Quantile regression. http://www.cscu.cornell.edu/news/statnews/stnews70.pdf. (access: 5.7.2012).

Didier, G.; Ternisien, E.; Caspary, O.; Razik, H. A new approach to detect broken rotor bars in induction machines by current spectrum analysis. *Mechanical Systems and Signal Processing* **21**, 1127–1142 (2007).

Doyleand, J.; Thomason, R. H. Background to qualitative decision theory. *AI Magazine* **20**, 55–68 (1999).

Durante, F.; Sempi, C. Copula theory: An introduction. *Copula Theory and Its Applications* **198**, 3–31 (2010).

eLICO. e-LICO: An e-laboratory for interdisciplinary collaborative research in data mining and data-intensive science. http://elico.rapid-i.com. (access: 26.1.2012).

Ertugrul, I.; Aytac, E. Construction of quality control charts by using probability and fuzzy approaches and an application in a textile company. *Journal of Intelligent Manufacturing* **20**, 139–149 (2009). http://dx.doi.org/10.1007/s10845-008-0230-1.

Figueira, J.; Greco, S.; Ehrgott, M. *Multi Criteria Decision Analysis: State of the art survays.* (Springer Verlag, Boston, Dordrecht, London, 2005).

Fischer, M.; Kock, C.; Schluter, S.; Weigert, F. An empirical analysis of multivariate copula models. *Quantitative Finance* **9**, 839–854 (2009).

Fisher, R. A. On a distribution yielding the error functions of several well known statistics. In: *Proceedings of the Congress for Mathematics, Toronto*. **2**, 805–813 (1924).

Forsythe, G.; Malcolm, M.; Moler, C. *Computer Methods for Mathematical Computations* (Prentice-Hall, Engelwood Cliffs, N. J., 07632, 1976).

Greco, S.; Matarazzo, B. Decision rule approach. In: Figueira, J.; Greco, S.; Ehrgott, M. (eds.) *Multi Criteria Decision Analysis: State of the art survays.* (Springer Verlag, Boston, Dordrecht, London, 2005).

Greco, S.; Matarazzo, B.; Slowinski, R. Rough sets theory for multicriteria decision analysis. *European Journal of Operational Research* **129**, 1–47 (2001).

Greco, S.; Mousseau, V.; R.Slowinski. Ordinal regression revisited: Multiple criteria ranking using a set of additive value functions. *European Journal of Operational Research* **191**, 415 – 435 (2008).

Greco, S.; Mousseau, V.; Slowinski, R. Utagms–int: Robust ordinal regression for value functions handling interacting criteria. *Technical Report*, Laboratoire Génie Industriel, Ecole Centrale Paris (2012).

Greco, S.; Słowiński, R.; Figueira, J.; Mousseau, V. *Trends in Multi Criteria Decision Analysis*, chapter Robust Ordinal Regression (Springer New York, 2010).

Hand, D. J.; Till, R. J. A simple generalisation of the area under the ROC curve for multiple class classification problems. *Mach. Learn.* **45**, 171–186 (2001).

Härdle, W.; Simar, L. *Applied Multivariate Statistical Analysis* (Springer-Verlag Berlin Heidelberg, 2007).

Hillier, F.; Lieberman, G. *Introduction to Operations Research* (The McGraw-Hill Companies, New Delhi, 2001).

Hofert, M. Sampling Archimedean Copulas. *Computational Statistics and Data Analysis* 5163–5174 (2008).

Izenman, A. J. *Modern Multivariate Statistical Techniques: Regression, Classification, and Manifold Learning* (Springer Science+Business Media, LLC, 233 Spring Street, New York, NY 10013, USA, 2008).

Jacquet-Lagreze, E.; Siskos, J. Assessing a set of additive utility functions for multicriteria decision-making, the uta method. *European journal of Operational Research* **10**, 151–164 (1982).

Jacquet-Lagreze, E.; Siskos, Y. Preference disaggregation: 20 years of MCDA experience. *European Journal of Operational Research* **130**, 233–245 (2001).

Jardine, A.; Lin, D.; Banjevič, D. A review on machinery diagnostics and prognostics implementing condition-based maintenance. *Mechanical Systems and Signal Processing* **20**, 1483–1510 (2006).

Joe, H. *Multivariate Models and Dependence Consepts* (Chapman and Hall/CRC, CRC Press LLC, 2000 N.W. Corporate Bvd., Boca Raton, Florida 33431, 1997).

Juričić, D.; Moseler, O.; Rakar, A. Model-based condition monitoring of an actuator system driven by a brushless DC motor. *Control Engineering Practice* **9**, 545–554 (2001).

Kaas, G. V. An exploratory technique for investigating large quantities of categorical data. *Applied Statistics* **29**, 119–127 (1980).

Klein, J. P.; Moeschberger, M. L. *Survival Analysis Techniques for Censored and Truncated Data* (Springer, New York, 2003).

Koenker, R. *Quantile Regression* (Cambridge university press, The Edinburgh Building, Cambridge cb2 2ru, UK, 2005).

Kononenko, I. *Strojno učenje* (Fakulteta za računalništvo in informatiko, Ljubljana, 1997).

Kosmidou, K.; Doumpos, M.; Zopounidis, C. (eds.) *Country risk evaluation: methods and applicaitons* (Springer Science+Business Media, LLC, 2008).

Larichev, O. Method ZAPROS for multicriteria alternatives ranking and the problem of incomparability. *INFORMATICA* **12**, 89–100 (2001a).

Larichev, O. Ranking multicriteria alternatives: The method ZAPROS III. *European Journal of Operational Research 131* **131**, 550–558 (2001b).

Leemis, L. M.; Park, S. K. *Discrete-Event Simulation: A First Course* (Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 2005).

Louis, W. On uncertainty measures used for decision tree induction. In: *Proceedings of the International Congress on Information Processing and Management of Uncertainty in Knowledge Based Systems, IPMU96*. 413–418 (1996).

Malakooti, B. Systematic decision process for intelligent decision making. *Journal of Intelligent Manufacturing* **22**, 627–642 (2011).

Martinez, W. L.; Martinez, A. R. *Computational Statistics Handbook with Matlab* (Chapman and Hall/CRC, CRC Press LLC, 2000 N.W. Corporate Blvd., Boca Raton, Florida 33431, 2002).

Mileva-Boshkoska, B.; Bohanec, M. Ranking of qualitative decision options using copulas. In: Klatte, D.; Lüthi, H.-J.; Schmedders, K. (eds.) *Operations Research Proceedings* (2011).

Mileva-Boshkoska, B.; Bohanec, M. A method for ranking non-linear qualitative decision preferences using copulas. *International Journal of Decision Support System Technology* **4**, 1–17 (2012).

Mileva-Boshkoska, B.; Bohanec, M.; Boškoski, P.; Juričić, Đ. Copula-based decision support system for quality ranking in the manufacturing of electronically commutated motors. *Journal of Intelligent Manufacturing* (2013). http://dx.doi.org/10.1007/s10845-013-0781-7.

Mileva-Boshkoska, B.; Bohanec, M.; Žnidaršič, M. Experimental evaluation of methods for ranking qualitatively assesed data-mining worksflows. In: Respício, A.; Burstein, F. (eds.) *Fusing decision support systems into the fabric of the context: [presented at 16th IFIP WG8.3 International Conference on Decision Support Systems, June 28-30 2012, Anávissos, Greece]*. **238**, 175–184 (Amsterdam: IOS Press, 2012).

Moshkovich, O.; Larichev, H. ZAPROS-LM– A method and system for ordering multiattribute alternatives. *European Journal of Operational Research* **82**, 503–521 (1995).

Myung, I. Tutorial on maximum likelihood estimation. *Journal of Mathematical Psychology* **47**, 90–100 (2003).

Nelsen, R. B. *An Introduction to Copulas* (Springer, New York, 2006), 2nd edition.

Noorbakhsh, F. International convergence or higher inequality in human development? Evidence for 1975- 2002. *Advancing Development: Core Themes in Global Economics* 149–167 (2007).

Orth, P.; Yacout, S.; Adjengue, L. Accuracy and robustness of decision making techniques in condition based maintenance. *Journal of Intelligent Manufacturing* **23**, 255–264 (2012).

Papoulis, A. *Probability, Random Variables, and Stohastic Processes* (McGraw-Hill, New York, 1991), 3rd edition.

Pečkov, A.; Džeroski, S.; Todorovski, L. A minimal description length scheme for polynomial regression. In: *Twelfth Pacific-Asia Conference on Knowledge Discovery and Data Mining, LNCS*. **5012**, 284–295 (Osaka, Japan, 2008).

Peng, Z.; Chu, F. Application of the wavelet transform in machine condition monitoring and fault diagnostics: a review with bibliography. *Mechanical Systems and Signal Processing* **18**, 199–211 (2004).

Rachev, S. T. (ed.) *Handbook of heavy tailed distributions in Finance* (Elsevier/North Holland, Amsterdam, 2003).

Raileanu, L. E.; Stoffel, K. Theoretical comparison between the gini index and information gain criteria. *Annals of Mathematics and Artificial Intelligence* **41**, 77–93 (2000).

Randall, R. B.; Antoni, J. Rolling element bearing diagnostics: "A tutorial". *Mechanical Systems and Signal Processing* **25**, 485 – 520 (2011).

Reed, R.; Lemak, D. J.; Mero, N. P. Total quality management and sustainable competitive advantage. *Journal of Quality Management* **5**, 5 – 26 (2000).

Rissanen, J. Mdl denoising. *IEEE Transactions on Information Theory* **46**, 2537–2543 (2000).

Röpke, K.; Filbert, D. Unsupervised classification of universal motors using modern clustering algorithms. *In Proc. SAFEPROCESS'94, IFAC Symp. on Fault Detection, Supervision and Technical Processes* **II**, 720–725 (1994).

Roy, B. Paradigms and challenges. In: *Multi Criteria Decision Analysis: State of the art survays*. 3–24 (Springer Science+Business Media, Inc., New York, 2005).

Sasi, B.; Payne, A.; York, B.; Gu, A.; Ball, F. Condition monitoring of electric motors using instantaneous angular speed. In: *paper presented at the Maintenance and Reliability Conference (MARCON), Gatlinburg, TN,* (2001).

Savu, C.; Trade, M. Hierarchical Archimedean Copulas. In: *International Conference on High Frequency Finance* (Konstanz, Germany, 2006).

Sawalhi, N.; Randall, R.; Endo, H. The enhancement of fault detection and diagnosis in rolling element bearings using minimum entropy deconvolution combined with spectral kurtosis. *Mechanical Systems and Signal Processing* **21**, 2616–2633 (2007).

Seeber, G. A. F.; Lee, A. J. *Linear Regression Analysis* (John Wiley and Sons, Inc., Hoboken, New Jersey., 2003).

Shestopaloff, Y. *Properties of sums of some elementary functions and their application to computational and modeling problems* (SP MAIK Nauka/Interperiodica, 2011).

Silverman, B. W. *Densiy Estimation for Statistics and Data Analysis* (Chapman and Hall/CRC, CRC Press LLC, 2000 N.W. Corporate Blvd., Boca Raton, Florida 33431, 1986).

Sklar, A. Random Variables, Joint Distribution Functions, and Copulas. *Kibernetika* **9** (1973).

Sklar, A. *Distributions with Fixed Marginals and Related Topics - Random Variables, Distribution functions, and Copulas - A Personal Look Backward and Forward*, *volume 28* (Institute of Mathematical Statistics, Hayward, CA., 1996).

Sloane, N. J. A. The on-line encyclopedia of integer sequences. http://oeis.org/. (access: 14.11.2011).

Tandon, N.; Choudhury, A. A review of vibration and acoustic measurement methods for the detection of defects in rolling element bearings. *Tribology International* **32**, 469–480 (1999).

Todorovski, L.; Ljubič, P.; Džeroski, S. Inducing polynomial equations for regression. In: *Proceedings of the 15th European Conference of Machine Learning* (Springer, Berlin, 2004).

Topp, R. *Regression and residual analysis in linear models with censored data*. Ph.D. thesis, Dem Fachberaich Statistik der Universitat Dortmund (2002).

Trivedi, P.; David, Z. Copula modeling: An introduction for practitioners. *Foundations and Trends in Econometrics* 1–111 (2006).

Vachtsevanos, G.; Lewis, F. L.; Roemer, M.; Hess, A.; Wu, B. *Intelligent Fault Diagnosis and Prognosis for Engineering Systems* (Wiley, New Jersey, 2006).

Walters, E. J.; Morrell, C. H.; Auer, R. E. An investigation of the median-median method of linear regression. *Journal of Statistics Education* **14** (2006).

Wasserman, L. *All of Nonparametric Statistics* (Springer Texts in Statistics, Springer Science+Business Media, Inc., 233 Spring Street, New York, NY 10013, USA, 2006).

Xu, K. How has the literature on gini index evolved in the past 80 years. *Technical Report*, Dalhousie University, Department of Economics (2004).

Xu, M.; Marangoni, R. Vibration analysis of a motor-flexible coupling-rotor system subject to misalignment and unbalance, part I: Theoretical model and analyses. *Journal of Sound and Vibration* **176**, 663–679 (1994).

Žnidaršič, M.; Bohanec, M.; Trdin, N.; Šmuc, T.; Piškorec, M.; Bošnjak, M. D7.3.v1 Non-functional WF assessment. *Technical Report*, Jožef Stefan Institut and Ruđer Bošković Institute (2011).

Zopounidis, C.; Pardalos, P. *Handbook of Multicriteria Analysis*. *Applied Optimization* (Springer, Heidelberg, 2010).

Publications related to the dissertation

Original scientific article

Mileva-Boshkoska, B.; Bohanec, M. A method for ranking non-linear qualitative decision preferences using copulas. *International Journal of Decision Support System Technology* **4**, 1–17 (2012).

Mileva-Boshkoska, B.; Bohanec, M.; Boškoski, P.; Juričić, Đ. Copula-based decision support system for quality ranking in the manufacturing of electronically commutated motors. *Journal of Intelligent Manufacturing* (2013). http://dx.doi.org/10.1007/s10845-013-0781-7.

Published scientific conference contribution

Bohanec, M.; Mileva-Boshkoska, B. Experimental evaluation of polynomial and copula functions for qualitative option ranking. In: *22nd International Conference on Multiple Criteria Decision Making, Malaga, Spain*. 183 (2013).

Mileva-Boshkoska, B.; Bohanec, M. Non-linear methods for ranking qualitative non-monotone decision preferences. In: Bohanec, M.; Gams, M.; Rajkovič, V.; Urbančič, T.; Benrnik, M.; Mladenić, D.; Grobelnik, M.; Heričko, M.; Kordeš, U.; Markič, O.; Lenarčič, J.; Žlajpah, L.; Gams, A.; Brodnik, A. (eds.) *Zbornik 13. mednarodne multikonference Informacijska družba - IS 2010, 11.-15. oktober 2010: zvezek A: volume A, (Informacijska družba)*. 31–34 (Ljubljana: Institut Jožef Stefan, 2010).

Mileva-Boshkoska, B.; Bohanec, M. Copula regression based ranking of non-linear decision options. In: Petelin, D.; Tavčar, A.; Rožič, B.; Pogorelc, B. (eds.) *3rd Jožef Stefan International Postgraduate School Students Conference, 2011, Ljubljana, Slovenija. Zbornik prispevkov.* 91–97 (2011a).

Mileva-Boshkoska, B.; Bohanec, M. Ranking of non-linear qualitative decision preferences using copulas. In: Dargam, F.; Delibasic, B.; Hernandez, J. E.; Liu, S.; Ribeiro, R.; Zarate, P. (eds.) *Proceedings of the EWG-DSS London-2011 Workshop on Decision Systems, June 23rd-24th, 2011, London*. 48 (Institut de Research en Informatique de Toulouse, 2011b).

Mileva-Boshkoska, B.; Bohanec, M. Ranking of qualitative decision options using copulas. In: *Book of abstracts of International Conference on Operations Research, 2011, Zurich, Switzerland.* 32 (Zurich: IFOR: = Institute for Operations Research, 2011c).

Mileva-Boshkoska, B.; Bohanec, M. Ranking of qualitative decision options using copulas. In: Diethard, K. (ed.) *Operations research proceedings 2011: selected papers of the International Conference on Operations Research, 2011, Zurich, Switzerland.* 103–108 (Berlin; Heidelberg, 2012).

Mileva-Boshkoska, B.; Bohanec, M.; Žnidaršič, M. Experimental evaluation of methods for ranking qualitatively assesed data-mining worksflows. In: Respício, A.; Burstein, F. (eds.) *Fusing decision support systems into the fabric of the context: [presented at 16th IFIP WG8.3 International Conference on Decision Support Systems, June 28-30 2012, Anávissos, Greece]*. **238**, 175–184 (Amsterdam: IOS Press, 2012).

Index of figures

Index of tables

Index of algorithms

Appendix

A Distribution of results with different tables

No.	**Method**	**Fully ranked** $(\%)$	**Breached monotonicity** $(\%)$
1	QQ	8.90	91.10
2	Frank	49.72	0
3	Gumbel	63.56	0
4	Clayton	36.18	0
5	All copulas	73.23	0

Table 1: Distribution of full ranking results of tables with two input and one output attributes

No.	**Method**	**Fully ranked** $(\%)$	**Breached monotonicity** $(\%)$
1	QQ	5.20	94.80
2	Frank	41.14	0
3	Gumbel	20.00	0
4	Clayton	26.10	0
5	All Copulas	52.44	0

Table 2: Distribution of full ranking results of tables with three input and one output attribute.

No.	**Method**	**Fully ranked** $(\%)$	**Breached monotonicity**$(\%)$
1	QQ	0.09	99.91
2	Frank	84.01	0
3	Gumbel	80.00	0
4	Clayton	96.38	0
5	All Copulas	99.03	0

Table 3: Distribution of full ranking results of tables with four input and one output attributes

Copula type	Constructed tables (%)	Full ranking (%)
Clayton FNAC	84.21	84.21
Clayton PNAC	14.71	12.26
Frank FNAC	83.03	83.03
Frank PNAC	1.91	1.91
Gumbel FNAC	78.98	78.98
Gumbel PNAC	1.3	1.13

Table 4: Distribution of the tables solved with copulas with four input and one output attribute

No. of dataset	No. of tables	# obtained solutions (%)	Monotonicity fulfilled (%)
1	19 683	0.4014	0.1575
2	2 278 734	0	0
3	1 000 000	0	0

Table 5: Time execution in seconds of different methods averaged over 100 calculations

No. of dataset	No. of tables	Monotonic (%)	# solutions fulfilling monotonicity and without ties (%)
1	19 683	79.08	0.8230
2	2 278 734	89.79	0.0018
3	1 000 000	59.69	0

Table 6: Polynomial interpolation with CIPER

No. of dataset	No. of tables	Monotonic (%)	# solutions fulfilling monotonicity and without ties (%)
1	19 683	16.99	14.88
2	2 278 734	0.0006	0.0001
3	1 000 000	0.01	0.01

Table 7: Polynomial interpolation with CIPER NEW

Table 8: Percentage of fully ranked tables obtained with QQ modified with impurity functions

Method	Dataset 1 (%)	Dataset 2 (%)	Dataset 3 (%)
gB	67.23	66.64	86.84
gC	30.65	11.79	57.67
gP	93.08	96.57	99.91
IG	75.85	80.61	95.76
χ^2	77.71	66.64	86.84

B Illustrative examples: rankings with different methods

This Appendix provides results from different methods on the examples provided in Chapter 8, which are provided in Tables 9, 11 and 13. For each result, the three properties are checked: monotonicity (2.3), full ranking (2.4) and consistency (2.5), which are given in Tables 10, 12 and 14. Bolded numbers in Tables 9, 11 and 13 represent ties in the results.

B.1 Appendix to Section 8.1

Table 9 gives the rankings obtained with modified QQ methods, CIPER and New CIPER, while Table 10 tells which of the properties: monotonicity (2.3), full ranking (2.4) and consistency (2.5), are fulfilled by each method. None of the methods fulfill all three properties.

Table 9: Rankings obtained by different methods: FNACs solves the breaching monotonicity

No.	A_1	A_2	A_3	A_4	Cls	gB	gCov	gPop	InfG	χ^2	CIP.	New CIP.
1	4	2	5	1	1	0.8125	1.1667	0.9463	0.8354	0.8125	3	2.0416
2	4	5	3	2	1	1.2344	0.6806	1.1707	1.2273	1.2344	3	2.9505
3	2	2	3	4	1	0.7656	0.8056	0.7824	0.7727	0.7656	3	3.0265
4	3	1	5	4	1	0.9375	1.3194	1.0995	0.9739	0.9375	3	1.6342
5	3	2	4	5	1	1.1406	1.1250	1.2176	1.1666	1.1406	3	2.4236
6	2	4	1	1	2	1.7832	1.6327	1.7224	1.7699	1.7832	3	3.1766
7	4	1	4	1	2	1.8363	2.2653	1.9304	1.8528	1.8363	3	2.3653
8	2	1	2	2	2	1.6504	1.9184	1.6667	1.6538	1.6504	3	3.4481
9	5	5	1	4	2	2.3496	1.8469	2.2995	2.3462	2.3496	3	3.2981
10	1	4	5	5	2	2.2345	2.3673	2.3333	2.2496	2.2345	3	2.8781
11	3	5	1	2	3	2.7931	2.6368	2.7402	2.7838	2.7931	3	2.8122
12	5	1	4	3	3	2.8793	3.3632	3.0500	2.9300	2.8793	3	2.7924
13	2	5	5	3	3	3.2069	3.2684	3.2598	3.2162	3.2069	3	2.7848
14	3	4	2	5	3	3.1897	2.9526	3.1207	3.1869	3.1897	3	2.9813
15	2	3	4	5	3	3.0690	3.2579	3.1501	3.0945	3.0690	3	2.6771
16	5	3	3	1	4	**3.8033**	**4.0047**	**3.8442**	**3.8122**	**3.8033**	3	3.3753
17	1	4	1	3	4	3.6393	3.6215	3.6312	3.6376	3.6393	3	3.5883
18	4	3	2	3	4	3.8279	3.9019	3.8389	3.8329	3.8279	3	3.3261
19	3	3	3	3	4	**3.8033**	**4.0047**	**3.8442**	**3.8122**	**3.8033**	3	2.9506
20	5	5	5	5	4	4.3607	4.3785	4.3688	4.3624	4.3607	3	3.0927
21	1	1	1	1	5	4.6250	4.6512	4.6250	4.6250	4.6250	3	4.1451
22	5	2	2	2	5	5.0735	5.0233	5.0627	5.0744	5.0735	3	3.7740
23	1	3	2	2	5	4.8824	4.8023	4.8697	4.8780	4.8824	3	3.6427
24	1	2	3	4	5	4.9853	5.1047	5.0051	4.9891	4.9853	3	3.3739
25	4	4	4	4	5	5.3750	5.3488	5.3750	5.3750	5.3750	3	2.8397

Table 10: Fulfillment of properties by different methods for the example in Table 9

Method	**Monotonicity** (2.3)	**Full ranking** (2.4)	**Consistency** (2.5)
gB	yes	**no**	yes
gC	yes	**no**	yes
gP	yes	**no**	yes
IG	yes	**no**	yes
χ^2	yes	**no**	yes
CIPER	yes	**no**	**no**
New CIPER	**no**	yes	**no**

B.2 Appendix to Section 8.2

Table 11 gives the rankings obtained with modified QQ methods, CIPER and New CIPER, while Table 12 tells which of the properties: monotonicity (2.3), full ranking (2.4) and consistency (2.5), are fulfilled by each method. None of the methods fulfill all three properties.

Table 11: Rankings obtained by different methods: PNACs solves the breaching monotonicity

No.	$\mathbf{A_1}$	$\mathbf{A_2}$	$\mathbf{A_3}$	$\mathbf{A_4}$	**Cls**	**gB**	**gCov**	**gPop**	**InfG**	$\boldsymbol{\chi^2}$	**CIP.**	**New CIP.**
1	2	4	1	1	1	0.7079	0.6303	0.7108	0.7104	0.7079	3	3.8170
2	5	2	2	2	1	**1.0787**	**1.0806**	1.0630	1.0728	**1.0787**	3	2.5633
3	5	1	4	3	1	1.2921	1.3602	1.2892	1.2896	1.2921	3	1.9889
4	2	2	3	4	1	**1.0787**	**1.0806**	1.0493	1.0807	**1.0787**	3	2.9001
5	3	1	5	4	1	1.2809	1.3697	1.2815	1.2848	1.2809	3	2.2367
6	4	2	5	1	2	1.8814	2.1516	1.9219	1.8838	1.8814	3	2.5663
7	2	1	2	2	2	1.6186	1.7254	1.6182	1.6184	1.6186	3	3.0694
8	4	4	4	4	2	2.1442	2.0533	2.1454	2.1447	2.1442	3	2.7811
9	1	4	5	5	2	2.0737	1.9631	2.0790	2.0797	2.0737	3	3.1200
10	5	5	5	5	2	2.3814	2.2746	2.3818	2.3816	2.3814	3	2.5205
11	4	3	2	3	3	2.8475	2.7778	2.8301	2.8413	2.8475	3	2.9181
12	3	3	3	3	3	2.8136	2.8120	2.8278	2.8152	2.8136	3	3.0244
13	3	4	2	5	3	**3.1864**	2.7350	3.1341	**3.1848**	**3.1864**	3	3.0508
14	3	2	4	5	3	**3.1864**	3.2650	3.1722	**3.1848**	**3.1864**	3	2.3587
15	2	3	4	5	3	3.1525	3.0342	3.1508	3.1587	3.1525	3	2.8006
16	5	3	3	1	4	3.9635	4.1394	4.0309	3.9627	3.9635	3	2.7714
17	4	1	4	1	4	3.7760	4.3333	3.8475	3.7738	3.7760	3	2.4526
18	1	4	1	3	4	3.6927	3.6667	3.6977	3.6947	3.6927	3	3.8517
19	2	5	5	3	4	4.2135	4.0727	**4.3023**	4.2291	4.2135	3	3.4942
20	5	5	1	4	4	4.3073	3.9636	**4.3023**	4.3053	4.3073	3	3.2377
21	1	1	1	1	5	4.6468	4.8661	4.6439	4.6465	4.6468	3	3.4590
22	3	5	1	2	5	5.1468	4.7165	5.1314	5.1465	5.1468	3	3.8143
23	1	3	2	2	5	4.9246	4.8661	4.9223	4.9273	4.9246	3	3.6623
24	4	5	3	2	5	5.3532	5.0866	5.3561	5.3535	5.3532	3	3.4263
25	1	2	3	4	5	5.0992	5.2835	5.0735	5.0991	5.0992	3	3.1363

Table 12: Fulfillment of properties by different methods for the example in Table 11

Method	**Monotonicity** (2.3)	**Full ranking** (2.4)	**Consistency** (2.5)
gB	yes	**no**	yes
gC	**no**	**no**	yes
gP	yes	**no**	yes
IG	yes	**no**	yes
χ^2	yes	**no**	yes
CIPER	yes	**no**	**no**
New CIPER	**no**	yes	**no**

B.3 Appendix to Section 8.3

Table 13 gives the rankings obtained with modified QQ methods, CIPER and New CIPER, while Table 14 tells which of the properties: monotonicity (2.3), full ranking (2.4) and consistency (2.5), are fulfilled by each method. Only New CIPER fulfills all three properties.

Table 13: Rankings obtained by different methods: symmetric attributes

No.	A_1	A_2	A_3	Cls	gB	gCov	gPop	InfG	χ^2	CIP.	New CIP.
1	1	1	1	1	1.0000	1.0000	1.0000	1.0000	1.0000	1.1111	1.2559
2	2	1	1	2	**1.8750**	**1.8750**	**1.8750**	**1.8750**	**1.8750**	**1.6667**	1.7333
3	3	1	1	2	**2.1250**	**2.1250**	**2.1250**	**2.1250**	**2.1250**	**2.2222**	2.2379
4	1	2	1	2	**1.8750**	**1.8750**	**1.8750**	**1.8750**	**1.8750**	**1.6667**	1.7587
5	2	2	1	2	**2.1250**	**2.1250**	**2.1250**	**2.1250**	**2.1250**	**2.2222**	2.2670
6	1	3	1	2	**2.1250**	**2.1250**	**2.1250**	**2.1250**	**2.1250**	**2.2222**	2.2892
7	1	1	2	2	**1.8750**	**1.8750**	**1.8750**	**1.8750**	**1.8750**	**1.6667**	1.7023
8	2	1	2	2	**2.1250**	**2.1250**	**2.1250**	**2.1250**	**2.1250**	**2.2222**	2.2016
9	1	2	2	2	**2.1250**	**2.1250**	**2.1250**	**2.1250**	**2.1250**	**2.2222**	2.2098
10	1	1	3	2	**2.1250**	**2.1250**	**2.1250**	**2.1250**	**2.1250**	**2.2222**	2.1758
11	3	2	1	3	**2.8750**	**2.8750**	**2.8750**	**2.8750**	**2.8750**	**2.7778**	2.8025
12	2	3	1	3	**2.8750**	**2.8750**	**2.8750**	**2.8750**	**2.8750**	**2.7778**	2.8283
13	3	3	1	3	**3.1250**	**3.1250**	**3.1250**	**3.1250**	**3.1250**	**3.3333**	3.3946
14	3	1	2	3	**2.8750**	**2.8750**	**2.8750**	**2.8750**	**2.8750**	**2.7778**	2.7280
15	2	2	2	3	**2.8750**	**2.8750**	**2.8750**	**2.8750**	**2.8750**	**2.7778**	2.7446
16	3	2	2	3	**3.1250**	**3.1250**	**3.1250**	**3.1250**	**3.1250**	**3.3333**	3.3066
17	1	3	2	3	**2.8750**	**2.8750**	**2.8750**	**2.8750**	**2.8750**	**2.7778**	2.7449
18	2	3	2	3	**3.1250**	**3.1250**	**3.1250**	**3.1250**	**3.1250**	**3.3333**	3.3152
19	2	1	3	3	**2.8750**	**2.8750**	**2.8750**	**2.8750**	**2.8750**	**2.7778**	2.6970
20	3	1	3	3	**3.1250**	**3.1250**	**3.1250**	**3.1250**	**3.1250**	**3.3333**	3.2453
21	1	2	3	3	**2.8750**	**2.8750**	**2.8750**	**2.8750**	**2.8750**	**2.7778**	2.6880
22	2	2	3	3	**3.1250**	**3.1250**	**3.1250**	**3.1250**	**3.1250**	**3.3333**	3.2493
23	1	3	3	3	**3.1250**	**3.1250**	**3.1250**	**3.1250**	**3.1250**	**3.3333**	3.2278
24	3	3	2	4	**4.0000**	**4.0000**	**4.0000**	**4.0000**	**4.0000**	**3.8889**	3.9128
25	3	2	3	4	**4.0000**	**4.0000**	**4.0000**	**4.0000**	**4.0000**	**3.8889**	3.8379
26	2	3	3	4	**4.0000**	**4.0000**	**4.0000**	**4.0000**	**4.0000**	**3.8889**	3.8293
27	3	3	3	5	5.0000	5.0000	5.0000	5.0000	5.0000	4.4444	4.4580

Table 14: Fulfillment of properties by different methods for the example in Table 13

Method	**Monotonicity** (2.3)	**Full ranking** (2.4)	**Consistency** (2.5)
gB	yes	**no**	yes
gC	yes	**no**	yes
gP	yes	**no**	yes
IG	yes	**no**	yes
χ^2	yes	**no**	yes
CIPER	yes	**no**	yes
New CIPER	yes	yes	yes

C Specific derivations for multi-variate Frank and Gumbel copulas

C.1 Specific derivation for multi-variate Frank copula

The generator function

$$\varphi(x) = -\ln\left(\frac{e^{-\theta x}-1}{e^{-\theta}-1}\right). \tag{1}$$

First derivative of the generator (1)

$$\frac{d\varphi}{dx} = \frac{\theta e^{-\theta x}}{e^{-\theta x}-1}.$$

The inverse generator for Frank copula is:

$$\varphi^{-1}(x) = 1 + \frac{x - \ln\left(1 - e^{\theta} + e^{\theta+x}\right)}{\theta}. \tag{2}$$

Finally, the derivative of the inverse generator (2) of order i is:

$$\begin{aligned}\frac{d^i\varphi^{-1}}{dx^i} = \frac{e^{\theta}-1}{\theta}\frac{e^{\theta+x}}{\left(e^{\theta}-e^{\theta+x}-1\right)^i} \\ \times \sum_{j=1}^{i-1} B_{i-1,j} e^{(i-j-1)(\theta+x)}\left(e^{\theta}-1\right)^{j-1},\end{aligned}$$

where $B_{i,j} = (i-j+1)B_{i-1,j-1} + jB_{i-1,j}$ and $B_{i1} = B_{ii} = 1$.

C.2 Specific derivation for multi-variate Gumbel copula

The generator function of Gumbel copula is given with:

$$\varphi(x) = (-\ln x)^{\theta}. \tag{3}$$

The first derivative of the generator (3)

$$\frac{d\varphi}{dx} = \frac{(-\ln x)^{\theta}\theta}{x\ln x}.$$

The inverse generator function is:

$$\varphi^{-1}(x) = \exp\left(-x^{1/\theta}\right). \tag{4}$$

The first derivative of (4) is:

$$\varphi^{-1[1]} = \frac{e^{-x^{\frac{1}{\theta}}}x^{\frac{1}{\theta}-1}}{\theta}.$$

The second derivative of (4) is:

$$\varphi^{-1[2]} = \frac{e^{-x^{\frac{1}{\theta}}}x^{\frac{2}{\theta}-2}}{\theta^2} - \frac{\left(\frac{1}{\theta}-1\right)e^{-x^{\frac{1}{\theta}}}x^{\frac{1}{\theta}-2}}{\theta}.$$

Third derivative of (4) is:

$$\varphi^{-1[3]} = -\frac{e^{-x^{\frac{1}{\theta}}} x^{\frac{3}{\theta}-3}}{\theta^3} + \frac{\left(\frac{1}{\theta}-1\right) e^{-x^{\frac{1}{\theta}}} x^{\frac{2}{\theta}-3}}{\theta^2} + \frac{\left(\frac{2}{\theta}-2\right) e^{-x^{\frac{1}{\theta}}} x^{\frac{2}{\theta}-3}}{\theta^2} - \frac{\left(\frac{1}{\theta}-2\right)\left(\frac{1}{\theta}-1\right) e^{-x^{\frac{1}{\theta}}} x^{\frac{1}{\theta}-3}}{\theta}.$$

Fourth derivative of (4) is:

$$\varphi^{-1[4]} = \frac{1}{\theta^4}\left\{ e^{-x^{\frac{1}{\theta}}} x^{\frac{1}{\theta}-4} \times \left[6\theta^3 + 11\theta^2\left(x^{\frac{1}{\theta}}-1\right) + x^{3/\theta} - 6x^{2/\theta} + 7x^{\frac{1}{\theta}} + 6\theta\left(x^{2/\theta} - 3x^{\frac{1}{\theta}} + 1\right) - 1\right]\right\}.$$

Fifth derivative of (4) is:

$$\varphi^{-1[5]} = \frac{1}{\theta^5}\left\{ e^{-x^{\frac{1}{\theta}}} x^{\frac{1}{\theta}-5} \times \left[-24\theta^4 - 50\theta^3\left(x^{\frac{1}{\theta}}-1\right) - 35\theta^2\left(x^{2/\theta} - 3x^{\frac{1}{\theta}} + 1\right) - x^{4/\theta} + 10x^{3/\theta} - 25x^{2/\theta} + 15x^{\frac{1}{\theta}} - 10\theta\left(x^{3/\theta} - 6x^{2/\theta} + 7x^{\frac{1}{\theta}} - 1\right) - 1\right]\right\}.$$

D Biography

Biljana Mileva Boshkoska was born on 11 January 1979 in Skopje, Macedonia. She has completed the secondary education at Nikola Karev gymnasium in Skopje, and obtained a diploma and a master degree at the Faculty of Electrical Engineering and Information Technologies in Skopje in 2002 and 2008. She worked on several EU projects on environment and vocational education from 2002–2006. She worked as a teaching assistant at the Faculty of Electrical Engineering and Information Technologies until 2009. In 2009 she started her PhD at the Jožef Stefan Postgraduate School under the supervision of Prof. Dr. Marko Bohanec. The results of her research work are published in several journal and conference publications.