Horizon 2020 Programme
Instrument: Innovation Action

Proactive Risk Management through Improved Cyber Situational Awareness

PROTECTIVE
PROACTIVE RISK MANAGEMENT

Start Date of Project: 2016-09-01

Duration: 36 months

D3.2 Meta-alerts ranking and prioritisation mechanisms report

Deliverable Details	
Deliverable Number	**D3.2**
Revision Number	**E**
Author(s)	**PSNC/GMV/AIT/SYNYO/TUDA/UOXF**
Due Date	**0817**
Delivered Date	**31/08/2017**
Reviewed by	**ITTI, UOXF**
Dissemination Level	**PU**
Contact Person EC	**Alina-Maria Bercea**

Contributing Partners	
1.	**PSNC (contributor, deliverable responsible)**
2.	**GMV (contributor)**
3.	**AIT (contributor)**
4.	**SYNYO (contributor)**
5.	**TUDA (contributor)**
6.	**UOXF (contributor, reviewer)**
7.	**ITTI (reviewer)**

The research leading to these results has received funding from the European Union's Horizon 2020 Research and Innovation Programme, under grant agreement no 700071.

Revision History

Revision	By	Date	Changes
E	PSNC	31/08/2017	Version Submitted to Agency
B3	PSNC	30/08/2017	Addresses UOXF suggestions.
B2	PSNC, AIT	28/08/2017	Incorporation of ITTI, UOXF and AIT suggestions. Transforming from Google Docs to MS Word.
B1	ITTI	25/08/2017	Updated with ITTI comments
A3	PSNC, AIT, GMV, SYNYO, TUDA, UOXF	23/08/2017	Integrated inputs from partners. Internal review ready version
A2	PSNC	12/06/2017	Working version on Google Docs
A1	PSNC	31/05/2017	Internal structure and ToC suggestion

The research leading to these results has received funding from the European Union's Horizon 2020 Research and Innovation Programme, under grant agreement no 700071.

Executive Summary

This deliverable contains an overview of methodology that can be applied to solve the problem of prioritisation and ranking. It discusses the importance of the problem and presents state of the art taking into account concepts from three closely related scientific sub-disciplines, namely Multi Criteria Decision Aiding (MCDA), Preference Learning (PL) and Machine Learning (ML).

Prioritisation and ranking taking into account context awareness and preference information is not a trivial task. It is shown in the report that this task requires the application of theory and a customisable operational framework to fulfil expectations of the end-users preferably in highly functional and visually attractive way. The important aspect of the theoretical part is the ability to deal with ambiguities and uncertainties in the data received from the sensors and introduced by the user during preference gathering. In the cybersecurity domain, it is often the case that not all the knowledge that drive decision maker's action is available in the system.

Thus uncertainties and ambiguities, from the perspective of the system, are inherent to the processes that the PROTECTIVE aims to support. Rough sets theory, dominance based principles, and intuitive preference gathering methodology appear to be the most reasonable approaches. The methodology will be further adjusted to the needs of particular users during the pilot phases.

The first part of the document focus on the provision of the theory of MCDA, PL and the concept of rough sets. It aims at providing solid ground for the discussions around implementations for the PROTECTIVE project. The second part discusses technical details that must be taken into account during integration, implementation, and development of the PROTECTIVE system.

Horizon 2020 Programme
Instrument: Innovation Action

The research leading to these results has received funding from the European Union's Horizon 2020 Research and Innovation Programme, under grant agreement no 700071.

Abbreviations List

AHP	Analytic Hierarchical Process
CAUSE	Criteria, Alternatives, Uncertainties, Stakeholders, Environment factors and constraints
CBA	Cost-Benefit Analysis
CR	Consistency Ratio
DM	Decision Maker
DRSA	Dominance-Based Rough Set Approach
ENISA	European Network and Information Security Agency
IDEA	Intrusion Detection Extensible Alert
IR	Information Retrieval
LHS	Left Hand Side
MA	Meta-Alert
MAUT	Multi-Attribute Utility Theory
MCDA	Multiple Criteria Decision Aiding
ML	Machine Learning
MLR	Machine-Learned Ranking
NDCG	Normalized Discounted Cumulative Gain
NFS	Net Flow Score
PCT	Pairwise Comparison Table (PCT)
PL	Preference Learning
PRanking	Perceptron-based Ranking
RHS	Right Hand Side
SAW	Simple Additive Weighting
SMART	Simple Multi-Attribute Rating Technique
SVM	Support Vector Machine
TOPSIS	Technique of Order Preference by Similarity of Ideal Solution
VC-DRSA	Variable-Consistency DRSA model
WSM	Weighted Sum Method

1 Introduction

IT security systems generate an extensive number of events and alerts that must be handled by a limited number of security officers on duty. Alerts are usually related to some incidents in IT system or network thus the security officer is, in fact, an incident handler. The meta-alert concept is the way to reduce the amount of data units to be handled. The reduction is done by application of correlation and aggregation techniques on sets of alerts and events (cf. D6.1). Prioritisation and ranking aim to support handling process by ordering the meta-alerts in the way coherent with expectations of the incident handler - the most important meta-alerts should be on top of the list. The expectations of incident handler can be modelled using preference learning techniques. Preferences of the handler are influenced by multiple factors. Those factors can be perceived as sets of criteria influencing decision-making process. Ranking building or the ordering problem, and sorting (assigning alternatives to graded classes) are the common tasks considered in Multiple Criteria Decision Analysis (MCDA).

This deliverable contains an overview of methodology that solves the problem of prioritisation and ranking. We discuss the importance of the problem and presents state of the art taking into account concepts from three closely related scientific sub-disciplines, namely MCDA, Preference Learning (PL) and Machine Learning (ML). The first part of the document focus on the provision of the theory of MCDA, PL and the concept of rough sets. It aims at providing solid ground for the discussions around implementations for the PROTECTIVE project. The second part discusses technical details that must be taken into account during integration, implementation, and development of the PROTECTIVE system.

The document is structured as follows. Section 1 serves as a comprehensive introduction to the problem of prioritisation and ranking. Section 2 gives an overview of the key concepts from the field of ranking and prioritisation. It shows the importance of the problem, provides the most important (from PROTECTIVE project perspective) excerpts from MCDA, Rough Sets, PL and ML theories. Section 3 describes the technical aspects of the implementation of ranking and prioritisation, shows how the criteria can be defined, discusses ways of direct preference gathering and proposes an approach for preference discovery, provides insights into storage requirements and data structures, discusses possible visualisations and identifies specific low-level requirements. Section 4 focuses shortly on the development. Finally, Section 5 concludes this document.

2 Prioritisation and ranking

Prioritisation is, in fact, a sorting process that assigns meta-alerts to defined graded classes (e.g. critical, important, and unimportant). A good example of prioritisation technique could be a MoSCoW (IIBA, 2009), methods commonly used in the analysis of business processes and project management e.g. specification of requirements.

The ranking is the solution to the ordering problem. The ordering problem by definition (Figueira, 2005) aims at "ranking the alternatives by decreasing order of preference. The prescription may be given in terms of a partial or a complete order." It is important to note that the order could be partial, it doesn't have to be complete in the mathematical sense. The ordering problem, known also as ranking problem, is one of a common problems considered in MCDA.

2.1 Importance of Prioritisation and Ranking

Need for prioritisation is an intrinsic part of the workplace with an extensive level of workload often not equally distributed in time and within the team. Time is among the most pervasive sources of pressure in the workplace (MindTools, 2017).

Prioritisation according to Eisenhower Principle (MindTools, 2017) allows focusing on the most important among the most urgent tasks. The ranking process allows to discover relations between the tasks and order them accordingly. The tasks or threats that are the most urgent do not have to be the most important from the business or operational perspective.

Increased number of security monitoring tools and devices produce vast amounts of data that need to be processed in a timely manner by a limited number of people and in accordance with internal policies. Prioritisation in many cases can be done manually, especially in the case of project management and prioritisation of requirements. However in the case of a flood of security alerts related to ongoing attacks or attack attempts, it must be automated to the largest possible extent. One of the aims is to reduce stress conditions influencing the reaction effectiveness. Security officer can't spend hours finding out what needs to be handled immediately.

The European Union Agency for Network and Information Security (ENISA) in "Good Practice Guide for Incident Management" (ENISA, 2010) recognized prioritisation as an important part of incident triage within incident handling process. Prioritisation is crucial also in escalation process where "severity and urgency can and should be used to decide what to escalate and to where." It is important to note, that escalation of too many false alarms will cause loss of credibility.

A few important points from ENISA (2010) guide:

1. Setting priorities is a good idea right from the start (of the incident management).
2. Reaction expected from a constituent may vary with the type of incident and its priority or importance.
3. The profile of the team influences prioritisation:
   a) If you are a company CERT, then you are likely to be most responsive and provide services of the highest level for those company resources defined as critical.
   b) If you are a governmental CERT, your mission is to protect your country .gov domain.
   c) If you are any other CERT with commercial contracts for an incident handling service, your goal is to deliver the best service to a paying customer.
4. Another factor to take into account in prioritisation is the severity of an incident you are handling.

ENISA gave two examples of basic prioritisation, one that takes into account severity of attacks (Table 1) and one that takes into account type of affected constituency member (Table 2).

Table 1: Basic prioritisation of incidents by severity of attacks (ENISA, 2010; provided in the original form)

Group	Severity	Examples
RED	Very High	DDoS, phishing site
YELLOW	High	Trojan distribution, unauthorised modification of information
ORANGE	Normal	Spam, copyright issue

Table 2: Basic prioritisation of incidents by type of constituency member (ENISA, 2010; provided in the original form)

PRIORITY	.GOV ORGANISATION	SLA CUSTOMER	OTHERS
RED	1	1	2
YELLOW	2	1	3
ORANGE	3	2	3

Additionally, in "Alerts, Warnings, Announcements" (ENISA, 2013) best practice guide ENISA identified "source importance rating" as one of the factors to be considered during prioritisation of incoming information. Within the document several measures usable in prioritisation process are presented. Among the most extensively discussed is TARANIS risk assessment model that uses chance matrix (probability measure) and impact matrix (damage potential) to compute risk measure.

PROTECTIVE will categorise and rank critical alerts based on the potential damage the attack can inflict on the threatened assets and hence on the organisations business as it was stated in the proposal. However, the assessment of potential damage is not based on a single factor - it is rather a combination of multiple factors, including host criticality, vulnerability exposure, and level of trust to the alert sources. It is worth to note, the damage itself could be quantified in multitude ways for various perspectives e.g. operational (number of damaged devices, a decrease of efficiency, downtime) or business (lost revenue, loss of customer trust). However, the incident handler making decisions what to handle has only partial knowledge about the importance of the systems and the possible influence of the attacks on business and operations.

Within PROTECTIVE to prioritise meta-alerts we can use information provided by Mission Impact Assessment module, Asset State Management module (cf. D4.1), and scores provided by TI Trust component (cf. D5.1). Those measures are the way of expressing risks related to the criticality of the assets, damage potential, and trust to the sources. The approach is coherent with ENISA suggestions. Those measures can be used to provide composite prioritisation score used to provide basic grouping of meta-alerts.

Ranking methods that we propose as addition to simple prioritisation can be applied in three ways:

1. To rank meta-alerts within priority groups, after assigning discrete priorities to them (sorting meta-alerts into priority groups);
2. To create priority groups based on the ranking of the values of multiple measures (creation of composite measure);
3. To provide ranking of all meta-alerts without sorting them into priority groups. Thus, ranking methods complement prioritisation techniques and can be seen as next level of handling support.

By application of appropriately crafted prioritisation and ranking methods, the following gains can be achieved from a business perspective: reduction of operational costs, more timely response, reduction of losses caused by security breaches.

2.2 Multicriteria Decision Aiding

In decision making problems, MCDA is the assessment of a few individual alternatives by explicitly considering the subjective preferences of a decision maker for the purpose of decision support (Geldermann & Treitz, 2008; Munda, 2003). MCDA supports decision makers in selecting a solution to decision and planning problems having multiple criteria. The solution may correspond to "best" (i.e. decision-maker may choose its most preferred alternative) alternative, a small set of good alternatives, grouping alternatives into different preference sets or all "efficient" or "non-dominated" alternatives. In the case of PROTECTIVE the alternatives corresponds to the meta-alerts (or more generally incidents) that must be served and MCDA should point out which one should be picked as the first one.

In MCDA, a decision is divided into small and more understandable parts. Each part of this decision is analyzed before integration to produce a meaningful solution. MCDA is now a vast field of research in various disciplines like (Ishizaka & Nemery, 2013): computer science, mathematics, management, informatics, psychology, social science, economic etc. In nutshell, MCDA can be used in any discipline where there is need to solve a problem which demands significant decision needs to be made. Every MCDA problem has five components (Addor & Smutko, 2011), (Belton & Pictet, 1997): a goal, a decision maker or a group of decision makers with their preferences, a set of decision alternatives, a set of evaluation criteria, and outcomes or consequences associated with alternative/interest combination.

2.2.1 Overview of MCDA Problems

In this section, the type and characteristics of MCDA problems are discussed as follows.

Types of MCDA Problems

According to (Kadziński et. al, 2016), (Roy, 1981), there are four main types of decision problems in MCDA (as shown in figure): choice, ranking (or ordering), sorting and description. These problems are explained as follows:

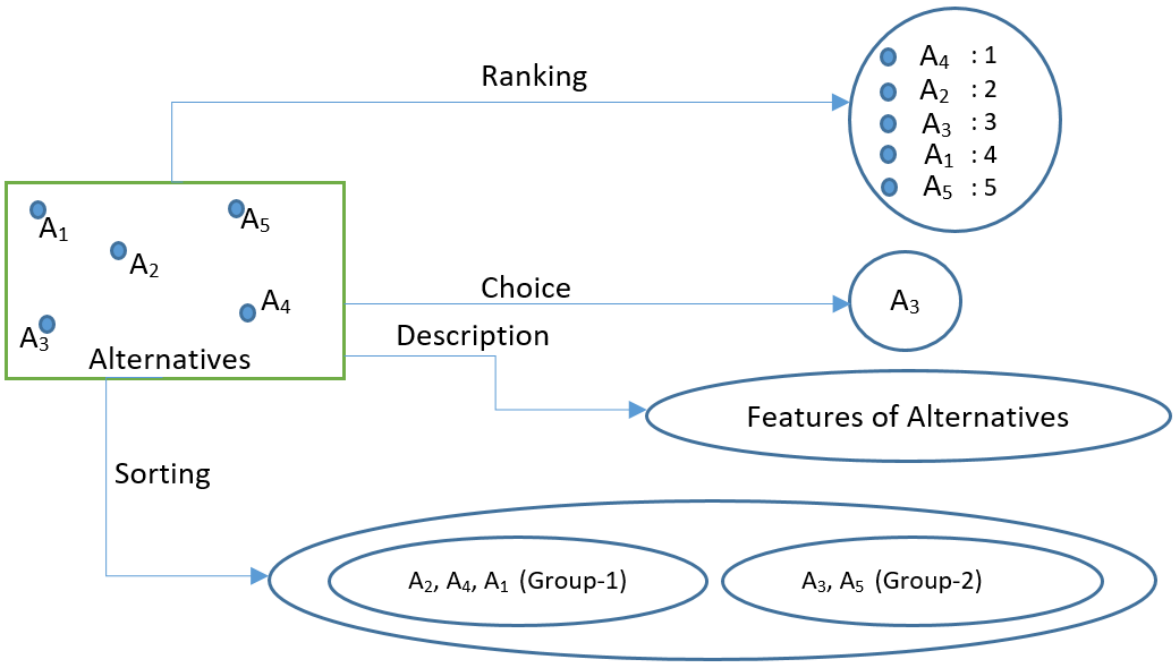


Figure 1: MCDA Problems

i. *The choice Problems*: A small subset of potentially best fitted alternatives is selected in a way that single best may be chosen in the end. Here, aim is to select one or a subset of the most satisfying optimal solutions. This subset contains alternatives which are non-comparable between each other. For example, a project manager selects the expert person for a project. MCDA methods that can be applied to solve choice problems include (Ishizaka & Nemery, 2013): AHP, ANP, MAUT/UTA, MACBETH, PROMETHEE, ELECTRE I, TOPSIS, Goal Programming, DEA etc.

ii. *The Ranking Problems:* Placing a set of alternatives in a preference order (from best to worst) by means of scores, pairwise comparisons etc. After ordering, top ranked are selected. For example, a student may rank the universities based on his area of interest, location, placement records, career opportunities etc. Thereafter, he may choose the top ranked universities to apply to. MCDA methods that can be applied to solve Ranking problems include (Ishizaka & Nemery, 2013): AHP, ANP, MAUT/UTA, MACBETH, PROMETHEE, ELECTRE III, TOPSIS, DEA etc.

iii. *The Sorting Problems*: Alternatives are grouped in a set of predefined categories, each one including a collection of non-distinctive alternatives. For example, cars can be evaluated for classification into different categories such as 'Mini', 'Small', 'Midsize', 'Large' and 'Very large'. Based on these classifications, cars from different brands can be put into these categories and necessary measures can be taken. As it can be easily analyzed that these methods reduce the number of alternatives thus helpful in initial screening. MCDA methods that can be applied to solve Sorting problems include (Ishizaka & Nemery, 2013): AHPSort, UTADIS, FlowSort, ELECTRE-Tri etc.

iv. *The Description Problems*: Here, alternatives and their consequences are described initially before making a decision. This helps in understanding the characteristics of the decision problems for making accurate decision. MCDA methods that can be applied to solve Description problems include (Ishizaka & Nemery, 2013): GAIA, FS-Gaia etc.

In addition to four main types of MCDA problems, three other types are also discussed in the MCDA community. These types are discussed as follows (Roy, 1981):

v. *The Elimination Problem:* The alternatives are eliminated one by one if they do not meet the specific goal or criteria. Elimination problems are usually considered as a branch of sorting problems.

vi. *The Design Problem*: A new action is identified or created for meeting the goals and aspirations of the decision maker.

vii. *The Elicitation Problem*: This elicits the preference parameter for a specific MCDA method.

For PROTECTIVE, the most important problems are sorting and ranking. The sorting problem is in fact what is broadly called prioritisation in the security community.

Common denominators of MCDA Problems: criteria, attributes, and objectives

In MCDA problems, criterion, attribute and objective are preferred terms used from decision making context. According to (Goodwin, 2003), (Xiaoqian, 2012), criterion, attribute and objective can be distinguished as follows:

- *Criterion:* a "criterion" is defined as "mean or standard of judging" by which performance of one particular choice is measured when evaluating an alternative. MCDA problems emerge as having multiple, usually conflicting attributes or objectives.
- *Attribute:* It is an inherent feature of an alternative. An attribute helps in measuring the performance in relation to an objective. However, a proxy attribute is not considered to be directly related to the objective.
- *Objective*: It is an indication of preferred direction of movement and operating synthesis in view of achieving a goal. The synthesis process may lead to change in desired direction. Further, an attribute with a direction is considered as an objective.

For example, level of comfort is a criterion while selecting a luxury car. Seating plan, fuel efficiency and noise are attributes which may be used to measure the level of comfort. Maximization of seating plan and fuel efficiency, and minimization of noise are objectives in the luxury car selection process.

According to (Couder, 2015), MCDA problems shares the following characteristics:

- *Multiple criteria*: MCDA approaches consider multiple criteria rather a single criterion in modeling a problem.
- *Conflict among criteria:* Each problem has multiple, usually conflicting objectives. It is assumed usually that no one option or alternative will be best in achieving all objectives. For example, more beneficial options or alternatives in selecting a luxury car may be safety and/or cost.
- *Contrasting units:* Each objective has a different unit of measurement. For example, benefit and cost are two objectives in selecting a luxury car. A high performance level is better for benefit measurement whereas, a low performance level is better for cost measurement.

2.2.2 Phases of MCDA Process

As shown in figure, Belton and Stewart (2002) generalized the MCDA process into three phases: (i) problem identification and structuring, (ii) model building and use, and (iii) the development of action plans.

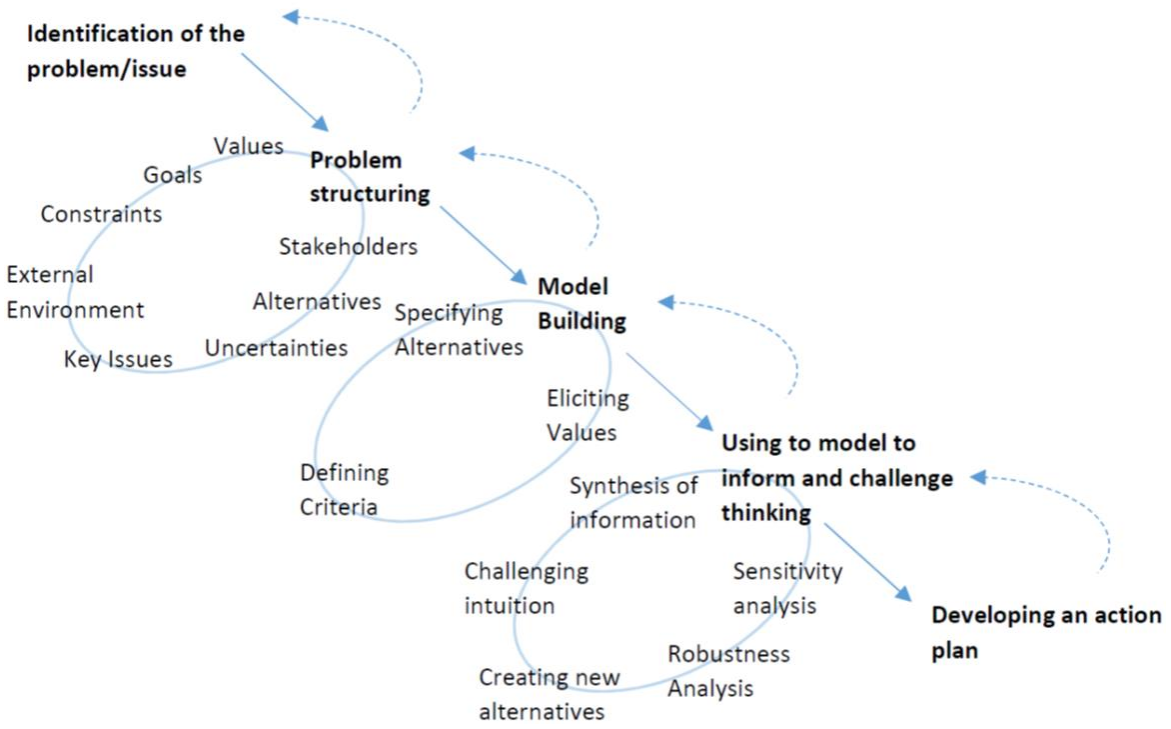


Figure 2: MCDA Process (Belton & Stewart 2002)

These phases are explained as follows:

- *Problem identification and structuring*: - In this phase, a problem is analysed broadly in terms of the CAUSE checklist (C: Criteria, A: Alternatives, U: Uncertainties, S: Stakeholders, E: Environment factors and constraints). This phase deals with understanding the problem, collecting the alternative thinking, study the complexity of the problem and follow the decision maker's process in moving forward. In this phase, a common understanding is established by taking the views of various stakeholders, including facilitators and technical analysts. This understanding is established on decisions that have to be made and their judgement and evaluation processes.
- *Model building and use*: - This phase extracts the essence of the issue from its complexity. This helps detailed and precise evaluation of selecting a roadmap in moving forward. Further, a formal model of decision making is developed such that alternative policies or actions under consideration can be compared with each other in a systematic and transparent manner.
- *Development of action plans*: - This phase translates the analysis process into specific plans of action which are required to solve the decision problem. Thus, MCDA process does not merely consider technical modelling and analytical features but also support and insight given to implementation.

2.2.3 Classical Approaches

For the purpose of this document, as classical methods are considered approaches that use outranking methods and multiattribute utility theory without extensions addressing uncertainties. Non-classical are those dealing with uncertainties, especially external uncertainties (cf. (Figueira et al., 2005)) related to imperfect knowledge concerning consequences of actions.

Weighted Sum Method (WSM) – Simple Additive Weighting (SAW)

In this method (Fishburn, 1967), values of alternatives on multiple criteria are combined into one overall value. Single overall value is computed by multiplying the value score on each criterion by the

weight of that criterion and then compute the sum of all those weighted scores together (Communities UK, 2009). The score of an alternative A_i is calculated as:

$$S_i = \sum_{j=1}^{n} w_j x_{ij} \;\; i = 1,2, \dots, m$$

Where,

S_i is the score of alternative A_i

w_j is the relative weight of importance of criteria C_i

m and n are the number of alternatives and criteria respectively

x_{ij} is the performance value of alternative A_i in terms of criterion C_i

In this computation, it is usually assumed that $\sum_{j=1}^{n} w_j = 1$. After computing the score of each alternative, sorting is performed for screening the alternative. One having the maximum score is selected as a best alternative.

Advantages of SAW are: (i) it is simple and efficient, and (ii) generates strong non-dominated solution that can be used as input for other techniques (Palesi, 2006).

Disadvantages of SAW are: (i) In order to compute the alternative scores, all the criterion values are required to be normalized and comparable which demands quantification and normalization methods to be integrated. These methods may influence the final result, (ii) if alternative are added or removed then it may affect the overall ranking, and (iii) this method is only applicable if multiple criteria are mutually independent.

Simple Multi-Attribute Rating Technique (SMART)

This technique is based on linear additive model. In this method, the ranking value ($R(A_i)$) of alternative Ai is computed as the weighted algebraic mean of the utility values associated with it i.e.

$$R(A_i) = \frac{\sum_{j=1}^{n} k_j v_j(A_i)}{\sum_{j=1}^{n} k_j} \;\; i = 1,2, \dots, m$$

Where,

k_j is the criteria (attribute) weight

$v_j(A_i)$ is the performance level of alternative A_i on the criterion or attribute $g_j(A_i)$

According to Edwards (1977) and Olson (1996), SMART is a simple way to implement the principles of multi-attribute utility theory (MAUT). In contrast to other Logical Decision and MAUT methods, SMART does not require any judgements of preference or indifference among hypothetical alternatives. The ten steps proposed for SMART technique includes [17]:

1. Identify the person or organization whose utilities are to be maximized.
2. Identify the issue or issues: Utility depends on the context and purpose of the decision.
3. Identify the alternatives to be evaluated.
4. Identify the relevant dimensions of value for evaluation of the alternatives.
5. Rank the dimensions in order of importance.
6. Rate dimensions in importance, preserving ratios.

7. Sum the importance weights, and divide each by the sum.
8. Measure the location of each alternative being evaluated on each dimension.
9. Calculate utilities for entities i.e.

$$U_i = \sum_j w_j u_{ij}$$

Here, $\sum_j w_j = 100$, U_i is the aggregated utility for the i-th entity, w_j is the normalized importance weight of the j-th dimension of value and u_{ij} is the rescaled position of the i-th entity on the j-th dimension.

10. Perform analysis and decide. If a single alternative (i-th) is to be selected, then U_i should be maximum elseif a subset of i is to be selected then $\sum_i U_i$ should be maximum.

*Advantages of SMART technique (*InforHarvest, 2017; Barfod & Leleur, 2014): (i) In this method, rating of alternatives is not relative i.e. changing the number of alternatives considered will not change the decision score of the original alternatives, (ii) SMART method is similar to that of well-accepted traditional Cost-Benefit Analysis (CBA). This characteristic makes it easier to grasp for the decision maker, and (iii) this method is preferred if in-depth analysis in input data is to be analyzed. Here, value functions need to be accesses for each of the lowest level attributes and weights should be assigned as trade-off.

Disadvantages of SMART technique (Barfod & Leleur, 2014; Hobbs and Meier, 2000): (i) According to Hobbs and Meier (2000), SMART oversimplify the problem if used as screening method, and (ii) It is recommended to use rank sum (RS) method over SMART if problems involves large numbers of criteria because SMART become progressively complex as the number of criteria increases.

Analytic Hierarchy Process (AHP)

In 1977, Saaty proposed the analytic hierarchy process (AHP) for transport planning problems in Sudan (Saaty, 1977). Nurse and Sinclair (2012) applied the AHP method to the field of security to support companies in negotiating and reconciling high-level security needs.

In AHP method, pairwise comparison was introduced which allows users to perform quantitative comparison for selecting a preferred alternative. This pairwise comparison of alternatives is performed based on their relative performance against the criteria. The basis of pairwise comparison is the human adaptability in recognizing relative judgements if quantitative ratings are unavailable.

AHP develop hierarchy of decision criteria by decomposing a decision-making problem into a system of hierarchies. A tree-structure is often constructed for criteria. For example: hierarchy tree for selecting a luxury car is shown in Figure 3 (Demir, 2014). The performance at a given level is multiplied with its weight and the weighted performances are summed to get the score at higher level. This process is repeated again and again in an upward direction until it reaches the top of hierarchy. Finally, an alternative with the maximum score is selected as best alternative.

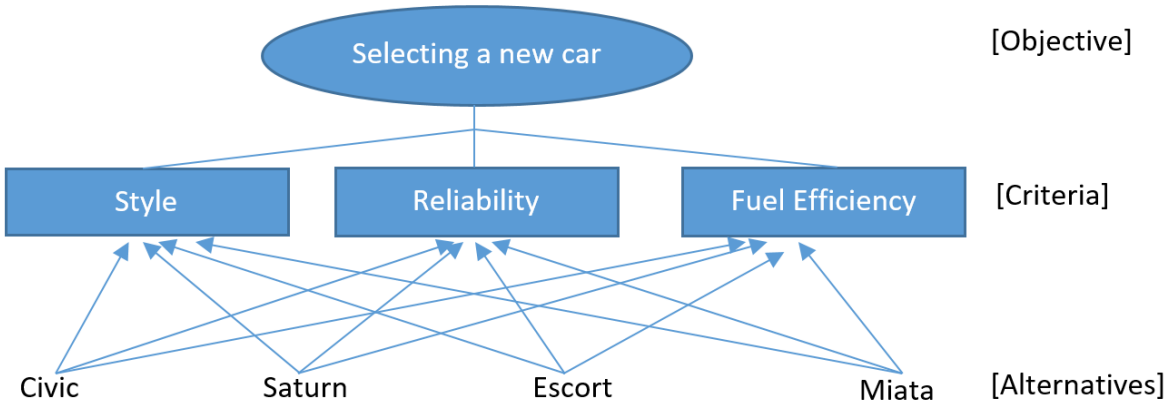


Figure 3: Hierarchy Tree for AHP analysis

In AHP, pairwise comparisons are made using a nine-point grades ranging from 1 to 9. The ranking scale for criteria, its definition and explanation is shown in Table 3.

Table 3: AHP ranking scale for criteria and alternatives

Intensity of Importance	Definition	Explanation
1	Equal importance	Two factors contribute equally to the objective
3	Somewhat more important	Experience and judgement slightly favour one over the other
5	Much more	Experience and judgement strongly favour one over the
7	Very much more	Experience and judgement very strongly favour one over
9	Absolutely more	The evidence favouring one over the other is of the highest
2,4,6,8	Intermediate	When compromise is needed

The following steps are used in AHP procedure:

Develop a pairwise comparison matrices: In matrix, If criteria 'C_x' is absolutely more important than criteria 'C_y' and is rate at 'R' then 'C_y' is considered to be absolutely less important than 'C_x' and is graded as 1/R. In nutshell, for each comparative score assigned, a reciprocal is awarded to the opposite relation. Now, if there are m-alternatives and n-criteria then n-(m x m) matrices are developed. Table 4 shows an example of one such matrix in which three ranking criteria (the style, reliability and fuel efficiency) are selected in selecting a new car.

Table 4: Ranking of criteria in AHP

	Style	Reliability	Fuel Economy
Style	1	1/2	3
Reliability	2	1	4
Fuel Economy	1/3	1/4	1

2. Calculate "priority vector" (i.e. normalized weight) for each criterion: This step consists of following two steps:

a) Compute normalize column sums
b) Take the overall row averages

For example:

$$\begin{pmatrix} 1 & 0.5 & 3 \\ 2 & 1 & 4 \\ 0.33 & 0.25 & 1 \end{pmatrix} \rightarrow \begin{pmatrix} 1/3.33 & 0.5/1.75 & 3/8 \\ 2/3.33 & 1/1.75 & 4/8 \\ 0.33/3.33 & 0.25/1.75 & 1/8 \end{pmatrix} = \begin{pmatrix} 0.30 & 0.29 & 0.38 \\ 0.60 & 0.57 & 0.50 \\ 0.10 & 0.14 & 0.13 \end{pmatrix} \rightarrow \begin{pmatrix} 0.30 \\ 0.60 \\ 0.10 \end{pmatrix}$$

In hierarchy tree structure, this can be represented as shown in Figure 4.

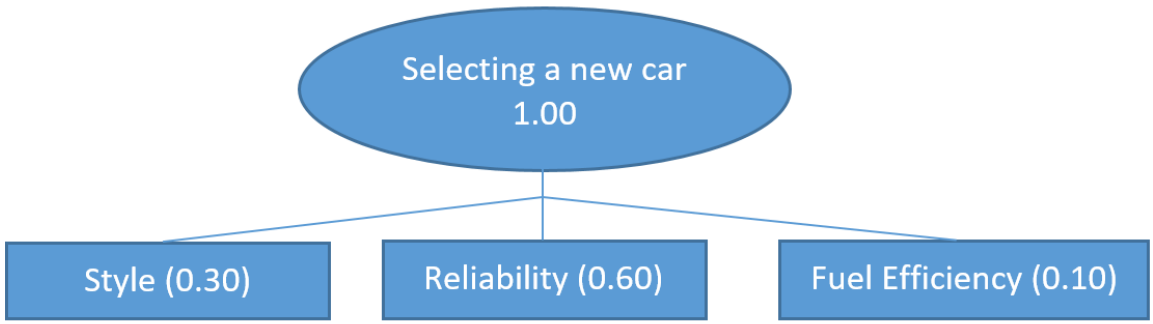


Figure 4: Hierarchy Tree for priority vector

In this evaluation, a method having the highest total score is considered as preferred alternative.

3. *Calculate the consistency index and consistency ratio (CR):*

$$\begin{pmatrix} 1 & 0.5 & 3 \\ 2 & 1 & 4 \\ 0.33 & 0.25 & 1 \end{pmatrix} \begin{pmatrix} 0.30 \\ 0.60 \\ 0.10 \end{pmatrix} = \begin{pmatrix} 0.90 \\ 1.60 \\ 0.35 \end{pmatrix} = \mu_{max} \begin{pmatrix} 0.30 \\ 0.60 \\ 0.10 \end{pmatrix}$$

$$\mu_{max} = average\left(\frac{0.90}{0.30}, \frac{1.60}{0.6}, \frac{0.10}{0.10}\right) = 3.06$$

$$\text{Consistency index, CI} = (\mu_{max} - n)/(n-1) = (3.06 - 3)/(3 - 1) = 0.03$$

For 3 criteria, consistency for random judgements is 0.58. Thus, CR (in this case) = CI/0.58 =0.05. Since 0.05 < 0.1, so the evaluations are consistent.

Advantages of AHP (Couder, 2015): (i) both qualitative and quantitative evaluations are possible, (ii) AHP calculations are simple and applicable when it is difficult to formulate criteria evaluations, (iii) AHP is applicable in group decision making environments also.

Disadvantages of AHP (Couder, 2015)*:* (i) difficult to use if the number of criteria or alternative is high (>7), (ii) difficult to add new criterion or alternative and (iii) difficult to exclude an existing criterion or alternative, since it may affect the best alternative, and (iv) decision maker may find it difficult to distinguish and use of 9-point scale.

Technique of Order Preference by Similarity of Ideal Solution (TOPSIS)

In 1981, Hwang and Yoon (1981) proposed the TOPSIS method. This method is based on two types of hypothetical solutions: positive-ideal alternative solution and negative-ideal alternative solution. The positive-ideal solution is one which has the best attributes values. The negative-ideal alternative solution is one which has the worst attribute values. In TOPSIS method, an alternative solution is selected which is closest (i.e. shortest (Euclidean) distance) to the positive-ideal alternative solution, and farthest from negative-ideal alternative solution. In TOPSIS method, decision matrix and relative weights are required as input parameters and follows the following steps for ordering preferences:

1. Standardise/Normalise the decision matrix.
2. Calculate weighted standardize/normalize decision matrix. This is constructed by multiplying attributed weight to each rating.
3. Determine positive ideal solution and the negative ideal solution.
4. Determine the distance of each alternative to positive-ideal solution and negative ideal solution.
5. Determine the relative closeness of each alternative to ideal solution.
6. Rank the alternatives based on value of their relative closeness.

Advantages of TOPSIS (Roszkowska, 2011): simplicity, good computational efficiency and ability to measure the relative performance for each alternative in a simple mathematical form.

Disadvantages of TOPSIS (Xu et al, 2015): correlations between criteria, uncertainty in obtaining the weights only by objective methods or subjective methods, and possibility of alternative closed to ideal point are among major disadvantages of TOPSIS method.

2.2.4 Dealing with Uncertainties – Rough Sets vs. Fuzzy Sets

In MCDA as non-classical approaches are usually considered those that are able to deal with uncertainties. The uncertainties in case of alerts processing could be, in particular, a result of external factors not included in the system that influence the decision making process (e.g. phone call, news about new campaign). Within non-classical approaches, very popular are the concepts that use extensions of the classical set theory. In particular the rough sets by Pawlak (1982) and fuzzy sets by Zadeh (1965).

Brief and succinct comparison of the rough sets versus fuzzy sets can be found in the Foreword to Pawlak's book (Pawlak, 1991):

Rough sets have often been compared to fuzzy sets, sometimes with a view to introduce them as competing models of imperfect knowledge. Such a comparison is misfounded. Indiscernibility and vagueness are distinct facets of imperfect knowledge. Indiscernibility refers to the granularity of knowledge, that affects

the definition of universes of discourse. Vagueness is due to the fact that categories of natural language are often gradual notions, and refer to sets with smooth boundaries.

Borrowing an example from image processing, rough set theory is about the size of the pixels, fuzzy set theory is about the existence of more than two levels of grey. Fuzzy set theory relies on ordering relations that express intensity of membership. Rough set theory is based on equivalence relations describing partitions made of classes of indiscernible objects. So one theory is quite distinct from the other, and they display a natural complementarity.

2.2.5 Rough Sets

Rough set theory has been introduced by Zdzislaw Pawlak in 1980's (Pawlak 1991, Komorowski 2011). The main goal of the rough set analysis is to synthesize approximation of concepts from the acquired data.

From a mathematical point of view, it is a very simple theory, since it requires only finite sets, equivalence relations, and cardinalities. The idea is that the universe on which we collect data is uselessly too refined for a clear expression of knowledge, i.e. the universe of concepts is coarser, and some object in our data set may become indiscernible (Pawlak 1991).

A finite set of objects creates *the universe*. Any subset of *the universe* is called *concept* or a *category*, and any *family of concepts* in the universe is referred as *abstract knowledge* (or in short *knowledge*) about the universe. The binary *equivalence relations* allow to describe partitions made of classes of indiscernible objects, so they define knowledge base.
For each subset (X), new sets being its *upper* and *lower approximation* can be defined in accordance with provided binary equivalence relation. The upper approximation contains all the object of X and additionally other objects from universe that are, according to equivalence relation, identical to some object from X. The intersection of upper and lower approximations are called *borderline*.

Categories are items of information which can be expressed by available knowledge. In other words, categories are subsets of objects having the same properties expressible in terms of our knowledge. In general, not all subsets of objects form categories in a given knowledge base, i.e. concepts which can be expressed by the knowledge; therefore, such subsets may be regarded as rough categories (i.e. imprecise or approximate categories) which can be only roughly defined employing our knowledge - by using two exact categories - the lower and the upper approximation. Hence the concept of approximation allows us to speak precisely about imprecise notions. (Pawlak 1991)

2.2.6 Dominance-Based Approach

One of the methods that can deal with uncertainties by combining rough set concept to dominance relations is Dominance-based Rough Set Approach. Let us introduce the general principles of the approach.

The basic aim of DRSA is to provide a set of decision rules that are coherent with preferences given by DM for a finite set of objects described by a finite set of criteria. Let's denote:

U - a finite set of objects

C - a finite set of criteria

Outranking relation

Let $\succcurlyeq_q$ be a weak preference relation on U (often called outranking) representing a preference on the set of objects with respect to criterion $q \in \{C\}$. Now, $x_q \succcurlyeq y_q$ means that x_q is at least as good as y_q with respect to criterion q. Assuming, without loss of generality, that the domains of the criteria are numerical (i.e. $x_q \in R$ for any $q \in \{C\}$) and that they are ordered so that the preference increases with the value, we can say that $x_q \succcurlyeq y_q$ is equivalent to $x_q \geq y_q$ on criterion q (Słowiński et al., 2012).

Dominance relation

We say that x dominates y with respect to $P \subseteq C$ (shortly, x P-dominates y) denoted by xD_Py, if $x_q \succcurlyeq y_q$ for all $q \in P$(x is better than y on every criterion from $P \subseteq C$). Observe that for each $x \in X_q$, xD_Px, i.e. P-dominance is reflexive. What's more, it is also transitive.

Given $P \subseteq C$ and $x \in U$, let $D_P^+(x)$ be a set of objects y dominating x, called the *P-dominating set*, defined as $D_P^+(x) = \{y \in U: yD_Px\}$.

Given $P \subseteq C$ and $x \in U$, let $D_P^-(x)$ be a set of objects y dominated by x, called the *P-dominated set*, defined as $D_P^-(x) = \{y \in U: xD_Py\}$. (Słowiński et al, 2012)

Multigraded dominance

Given subset $P \subset C$ ($P \neq \emptyset$)of criteria and pairs of objects $(x,y),(w,z) \in A \times A$, the pair (x,y) is said to P-dominate the pair (w,z) (denotation $(x,y)D_P(w,z)$), if $P_i[g_i(x),g_i(y)] > P_i[g_i(w),g_i(z)]$ for all $g_i \in P$i.e. if $x$is preferred to y at least as strongly as w is preferred to z with respect to each criterion $g_i \in P$ (cf. Figure 5).

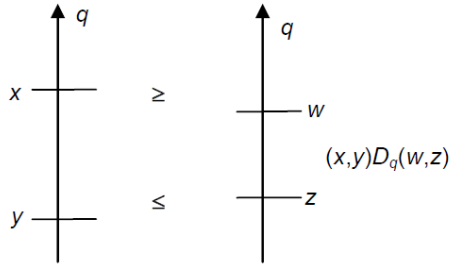


Figure 5: Multigraded dominance example

Given $P \subseteq C$ and $x,y \in U$, let $D_P^+(x,y)$ be a set of pairs (w,z) dominating pair (x,y), called the *P-dominating set*, defined as $D_P^+(x,y) = \{(w,z) \in U: (w,z)D_P(x,y)\}$.

Given $P \subseteq C$ and $x,y \in U$, let $D_P^-(x,y)$ be a set of pairs (w,z) dominated by pair (x,y), called the *P-dominated set*, defined as $D_P^-(x,y) = \{(w,z) \in U: (x,y)D_P(w,z)\}$. (Słowiński et al., 2012)

Rough approximations

The considered objects are evaluated by criteria from set C from one side, and by the comprehensive decision d from the other side. Using the dominance relation with respect to d, we can define unions of classes relative to a particular dominated or dominating class – these unions of classes are called upward and downward unions of classes. The upward union $Cl_t^{\geq}$ is a class containing all the same or better classes than Cl_t. Analogously, $Cl_t^{\leq}$ contains the same or worse classes.

Object $x \in U$ can create an inconsistency with the dominance principle with respect to the upward union of classes $Cl_t^{\geq}$, $t = 2,3,\ldots,n$, if one of the following conditions holds:

- $x$belongs to class Cl_t or better, but it is P-dominated by an object y belonging to a class worse than Cl_t
- x belongs to a worse class than Cl_t, but it P-dominates an object y belonging to class Cl_tor better

If such an inconsistency exists, we say that x belongs to $Cl_t^{\geq}$ with some ambiguity. Thus, x belongs to $Cl_t^{\geq}$ without any ambiguity, if $x \in Cl_t^{\geq}$ and there is no inconsistency with the dominance principle.This means that all objects P-dominating x belong to $Cl_t^{\geq}$, i.e. $D_P^+(x) \subseteq Cl_t^{\geq}$.

x possibly belongs to $Cl_t^{\geq}$ if according to decision d attribute belongs to $Cl_t^{\geq}$, or it is inconsistent in the sense of the dominance principle with an object y belonging to $Cl_t^{\geq}$.

The P-lower approximation of an upward union $Cl_t^{\geq}$, $\underline{P}(Cl_t^{\geq})$ is composed of all objects x from the universe such that all objects y having at least the same evaluations on all the considered criteria from P also belong to class Cl_tor better. The P-upward approximation of an upward union $Cl_t^{\geq}$, $\underline{P}(Cl_t^{\geq})$ is composed of all objects x from the universe such that all objects y having at least the same evaluations on all the considered criteria from P also belong to class Cl_tor better. Analogously, the P-lower and the P-upper approximation of downward union can be created. The set difference between upper and lower approximation, the P-boundaries, is composed of objects whose assignment to the considered upward union $Cl_t^{\geq}$ or downward union $Cl_t^{\leq}$ is ambiguous, that is inconsistent, with the dominance principle.

$$Bn_P(Cl_t^{\geq}) = \underline{P}(Cl_t^{\geq}) - \underline{P}(Cl_t^{\geq})$$

$$Bn_P(Cl_t^{\leq}) = \underline{P}(Cl_t^{\leq}) - \underline{P}(Cl_t^{\leq})$$

Variable-Consistency Dominance-Based Rough Set Approach

The definitions of rough approximations introduced are based on a strict application of the dominance principle. However, when defining nonambiguous objects, it is reasonable to accept a limited proportion of negative examples, particularly for large data tables. Such extended version of DRSA is called Variable-Consistency DRSA model (VC-DRSA).

For any $P \subseteq C$, we say that $x \in U$ belongs to $Cl_t^{\geq}$ without any ambiguity at consistency level $l \in (0,1]$, if $x \in Cl_t^{\geq}$ and at least $l \cdot 100\%$ of all objects $y \in U$ dominating x with respect to P also belong to $Cl_t^{\geq}$ i.e.

$$\frac{\left|D_P^+(x) \cap Cl_t^{\geq}\right|}{\left|D_P^+(x)\right|} \geq l$$

Analogously, for any $P \subseteq C$, we say that $x \in U$ belongs to $Cl_t^{\leq}$ without any ambiguity at consistency level $l \in (0,1]$, if $x \in Cl_t^{\leq}$ and at least $l \cdot 100\%$ of all objects $y \in U$ dominating x with respect to P also belong to $Cl_t^{\leq}$ i.e.

$$\frac{\left|D_P^-(x) \cap Cl_t^{\leq}\right|}{\left|D_P^-(x)\right|} \geq l$$

The level is called consistency level because it controls the degree of consistency between objects qualified as belonging to $Cl_t^{\leq}$(or $Cl_t^{\geq}$) without any ambiguity. The concept of non-ambiguous objects at some consistency level l leads naturally to the modified definition of P-lower and P-upper

approximations. The variable consistency model of the dominance-based rough set approach provides some degree of flexibility in assigning objects to lower and upper approximations of the unions of decision classes.

Decision rules

The end result of DRSA is representation of the information contained in the considered data table in terms of simple "if..., then..." decision rules. If fact, the decision rules are not induced directly from the data table but from lower and upper approximations of upward and downward unions of decision classes. For a given upward or downward unions of classes, the decision rules induced under a hypothesis that objects belonging to $\underline{P}(Cl_t^{\leq})$ or $\underline{\underline{P}}(Cl_t^{\geq})$ are positive (i.e. must be covered by the induced rules) and all the others negative (i.e. must not be covered by the induced rules). There are five types of decision rules:

1. certain $D_{\geq}$-decision rules, providing lower profile descriptions for objects belonging to union $\underline{Cl_t^{\geq}}$ without ambiguity
2. possible $D_{\geq}$-decision rules, providing lower profile descriptions for objects belonging to union $\underline{Cl_t^{\geq}}$ with or without ambiguity
3. certain $D_{\leq}$-decision rules, providing upper profile descriptions for objects belonging to union $\underline{Cl_t^{\leq}}$ without ambiguity
4. possible $D_{\leq}$-decision rules, providing upper profile descriptions for objects belonging to union $\underline{Cl_t^{\leq}}$ with or without ambiguity
5. approximate$D_{\geq\leq}$-decision rules, providing simultaneously lower and upper profile descriptions for objects belonging to classes $\underline{Cl_s \cup Cl_{s+1} \cup \ldots \cup Cl_t}$ without possibility of discerning to which class

The rules of type 1) and 3) are exact, if they do not cover negative examples, and they are probabilistic otherwise. Probabilistic rules are concordant with the VC-DRSA model resented above. Since a decision rule is a kind of implication, by a minimal rule we understand such an implication that there is no other implication with the antecedent (the LHS of the rule) of at least the same weakness (in other words, a rule using a subset of its elementary conditions and/or weaker elementary conditions) and the consequent (the RHS of the rule) of at least the same strength. A set of decision rules is complete if it is able to cover all objects from the data table in such a way that consistent objects are re-classified to their original classes and inconsistent objects are classified to clusters of classes referring to this inconsistency.

2.3 Preference Learning

Preference learning is defined as the task of learning to predict an order relation on a collection of objects (alternatives) (Seel (ed.), 2012). Generally speaking, it is a process of inducing predictive preference models from empirical data (Fürnkranz & Hüllermeier, 2010). As shown in Figure 6, preference learning is subfield of machine learning, and preference modelling and decision analysis. It plays an important role in automated decision making especially in fields such as machine learning, information retrieval, applied mathematics and decision sciences as shown in Figure 7 (Doyle, 2004). In conclusion, it can be said that methods for learning and constructing preference models, from explicit or implicit preference information, are among the contemporary topic of research trends in disciplines such as machine learning, knowledge discovery, information retrieval, statistics, social choice theory, multiple criteria decision making, operations research etc.

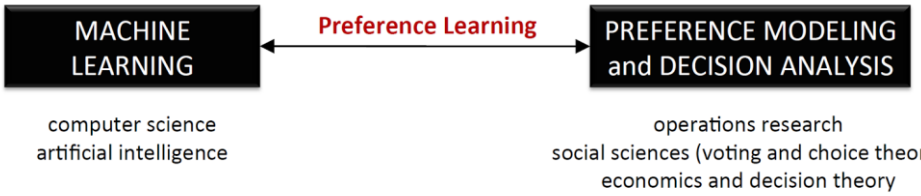


Figure 6: Relationship between Machine Learning, Preference Learning and Preference Modelling and Decision Analysis (Fürnkranz & Hüllermeier, 2011)

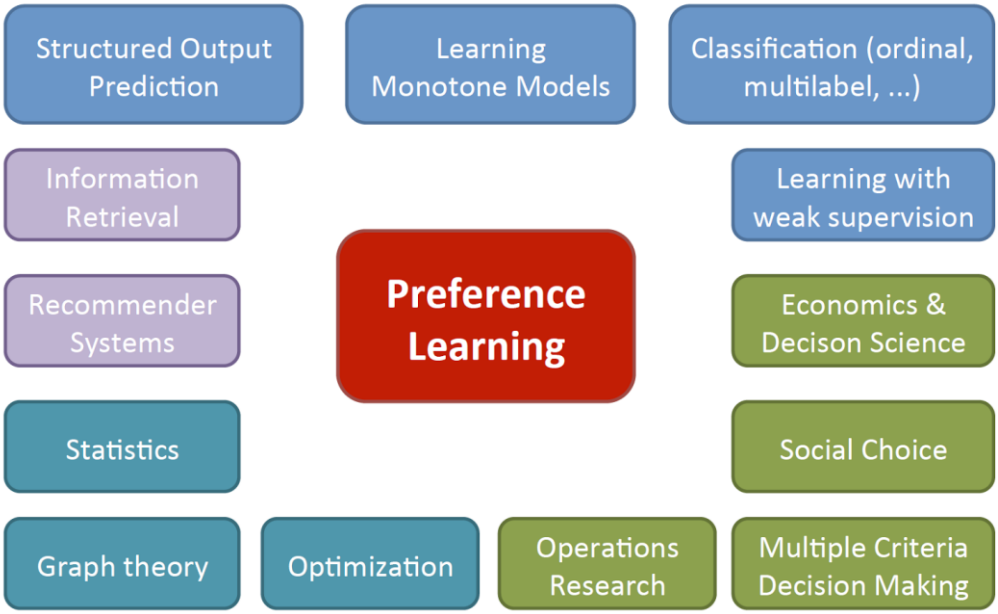


Figure 7: Preference and learning and related research areas within machine learning (blue), information retrieval (purple), applied mathematics (turquoise), and the decision sciences (green) (Fürnkranz et al., 2014).

2.3.1 Choice of preference learning strategy for PROTECTIVE

Preference learning can be assessed in various ways as shown in Figure 8. Broadly, there are two ways: absolute and relative. These two classes are explained as follows:

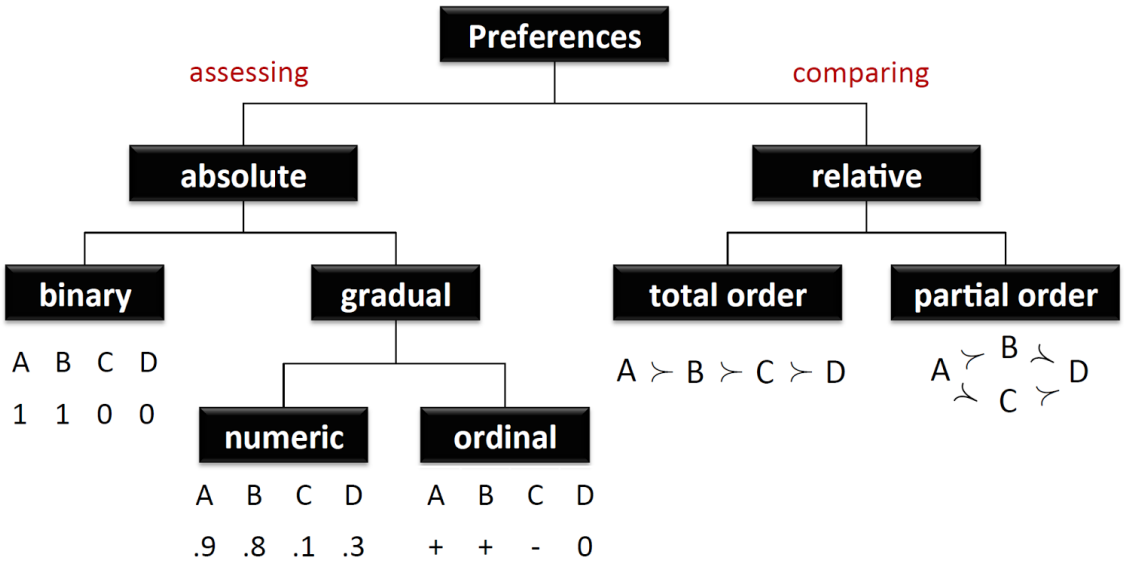


Figure 8: Ways of Preference Learning (Fürnkranz & Hüllermeier, 2011)

- *Absolute preference learning* is selecting a particular alternative or a set of alternatives after ranking, labelling or other mathematical/symbol representation. This can be classified as: binary or gradual preference learning.
    - In binary preference learning, binary bits are assigned to data and inferred function is used to extract a ranking from preference relation. For example: Table 5 shows a binary inferred function F(i,j) to infer the rank of two objects (Y1 and Y2). As F(1,0)=1 and F(0,1)=0, so it can be inferred that Y2 is having higher preference than Y1 (i.e. Y2Y1).

Table 5: Binary Preference Learning

F(i,j)	Y1	Y2
Y1		0
Y2	1	

    - In gradual preference learning, two types of preference learning are possible: numeric and ordinal. In numeric preference learning, a numeric scale is used to rank the preferences. For example: a numeric scale (0.1 to 0.9) is used to rank four objects (A, B, C and D) in Figure 8. Thus, an order inference can be drawn as: ABDC. In ordinal preference learning, mixed scales (numeric, symbol, binary) are used. Here, ranking of symbols decide the first preference order and other preference models should be used for ranking same symbol objects.
- *Relative preference learning* compares one object with another to generate the order. This preference learning method can be classified into two ways: total order and partial order. In total order preference learning, all objects are ranked on one scale and no two object lies at same point. In partial order preference learning, one scale is used to rank objects and two objects may lie at same point as shown in Figure 8.

MCDA vs Preference Learning

Table 6 shows the comparative analysis of PL and MCDA.

Table 6: PL vs MCDA (Hüllermeier, 2014)

	Preference Learning	**Multi Criteria Decision Analysis**
Problem Focus	Predictions	User/decision maker
User Interaction	Typically not, yet possible in active learning	Constructive, feedback with user in the loop
Learning domain	Population (generalize across individuals)	Single user

Representation of alternatives	Features-based, but also structured, often many (generic) features	Monotone, well-engineered criteria, decision space versus criteria space
Representation of users	Feature-based	No features of the DM used
Preference information	Global/holistic, example-based	Local and/or global, rich specifications
Nature of the data	Noisy/probabilistic	Consistent, possibly corrected
Models and model assumptions	Possibly weak assumptions (compensated by massive data)	Stronger assumptions, axiomatic foundation
Model interpretation, usage, and expectations	Mainly predictive, accurate prediction of decision maker's behaviour	Mainly constructive or normative, convincing explanations of decisions
Data availability	Data sets massively available (but not always accessible)	Limited, user-generated data, no benchmark data
Data volume	Possibly very large ("big data")	Typically small
Validation, success criteria	Accuracy metrics, internal validation on data	User satisfaction (difficult to measure)
Computational aspects	Scalability is critical	Less critical (but short response time required)
Application domains	Broad but typically not safety-critical (e-commerce, etc.), automated decisions	Broad, possibly safety critical, one-shot decisions

Final choice of preference learning strategy

Due to the nature of meta-alerts containing multiple, sometimes conflicting criteria, the best choice seems to be *relative* way of gathering preferences leading to *partial order.* It is worth to note, that such strategy is easy to understand for end-user and it is relatively easy to deal with changing number of criteria over time as well as conflicting results of comparisons. The conflicts in comparisons of meta-alerts may lead to uncertainties in MCDA process.

2.3.2 Gathering Preferences

Manual preference modelling is cumbersome task. Thus, preference learning is automated acquisition of preference model from data i.e. data from which preference information can be deduced in a direct or indirect way (Seel (ed.), (2012)). For example: when Internet user searches something in a search engine then results are displayed in a specific order. Ideally, this order is from most interesting to least interesting. Google search engine uses PageRank algorithm developed by Google founders Sergery Brin and Larry Page (White, 2006). In PageRank method, ranking is based on how pages are linked. In a constantly changing web-world, Google calculates ranking every month. Hence, google displays the web pages in order of assessed month's ranking preference.

According to Greco et al. (1999) the preference model can be also built using a preferential information provided by the decision maker (DM). This information concerns a set of, so called, *reference actions*, with respect to which the DM is willing to express his/her attitude through pairwise comparisons. The pairwise comparisons are considered as exemplary. Two kinds of these decisions can be considered: historical, and simulated. Historical examples represent actual decisions taken by the DM in the past. Simulated examples represent decisions taken by the DM on fictitious or real reference actions with the aim of using them for preference modelling.

In order to represent preferential information provided by the DM in form of a pairwise comparison of reference actions, it is recommended to use a *pairwise comparison table* (PCT) introduced by Greco et al. (1995).

Criteria, actions, scenarios and decision makers based preference aggregation approaches

Identified alternatives may be compared using criteria, scenarios and decision makers for gathering preferences and solving a problem. Ideally, if an alternative outperforms all other alternatives for all criteria, scenarios and decision makers then it is considered as the most preferred alternative. A decision problem may consist of multiple criteria, actions, decision makers and scenarios. Examples of preference aggregation are as follows (Beroggi, 2013):

- preferences aggregation across *criteria* include: linear weight function, linear scaling model, or multiplicative mode.
- preferences aggregation across *actions* include: non-linear and linear model.
- preference aggregation across *scenarios* include: expected value.
- preference aggregation across *decision makers* include: through voting.

Interactive Model based complete strong preference ordering approaches

In an analytical process (Beroggi, 2013), model based preference aggregation approaches can be classified into three categories: structural, formal and resolution. These approaches are explained as follows:

- **Structural Model**: This is the simplest way of gathering preferences as shown in Figure 9. In this model, two alternatives are picked at a time from a given set of alternatives. A pairwise comparison between these two alternative is performed and a transitive conclusion is drawn until the ranking is completed. Now, if a decision maker may choose two alternatives which has been already inferred by transitivity then system ask the user to keep the transitivity relationship or derive new relation. If user choose to reject transitivity relation then system deletes all direct and inferred assessments which are related to selected alternatives.

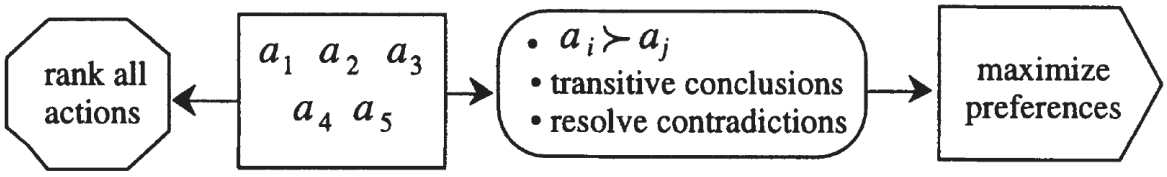


Figure 9: Structural model of interactive complete strong preference ordering (Beroggi, 2013)

- **Formal Model:** This model is based on assumption that decision maker can determine strong preference between any two alternatives from a given set of alternatives. In this formal process, a pair of alternatives are selected for determining the preference until all elements are completely ranked from most to least preference order.
- **Resolution Model:** The preference gathering process in this model is explained in figure. If the decision maker picks a pair whose preference has already been transitively inferred, s/he can either accept the preference relation between the two elements or reject it.
  - If it is accepted, the assessment is confirmed and a new, not yet assessed, pair can be chosen for assessment.
  - If the inferred preference order $a_i \succ a_j$ is rejected, then all pairs of S (set for direct strong preference Assessment) and I (set for inferred preferences) with at least one element being related (directly or indirectly) to a_i or a_j are deleted. The new pair $a_j \succ a_i$ is put into S. The related elements are found by going through the sets S and building up iteratively a list, called Error. Eventually, only disjoint preferences remain after the resolution of a contradiction.

For example, if a decision maker assesses $a_2 \succ a_5$, $a_1 \succ a_3$, $a_3 \succ a_4$, the model concludes, $a_1 \succ a_4$. If, in the next direct assessment, the decision maker decides, $a_4 \succ a_1$, which is a contradiction, the model retains , $a_4 \succ a_1$ and , $a_2 \succ a_5$.

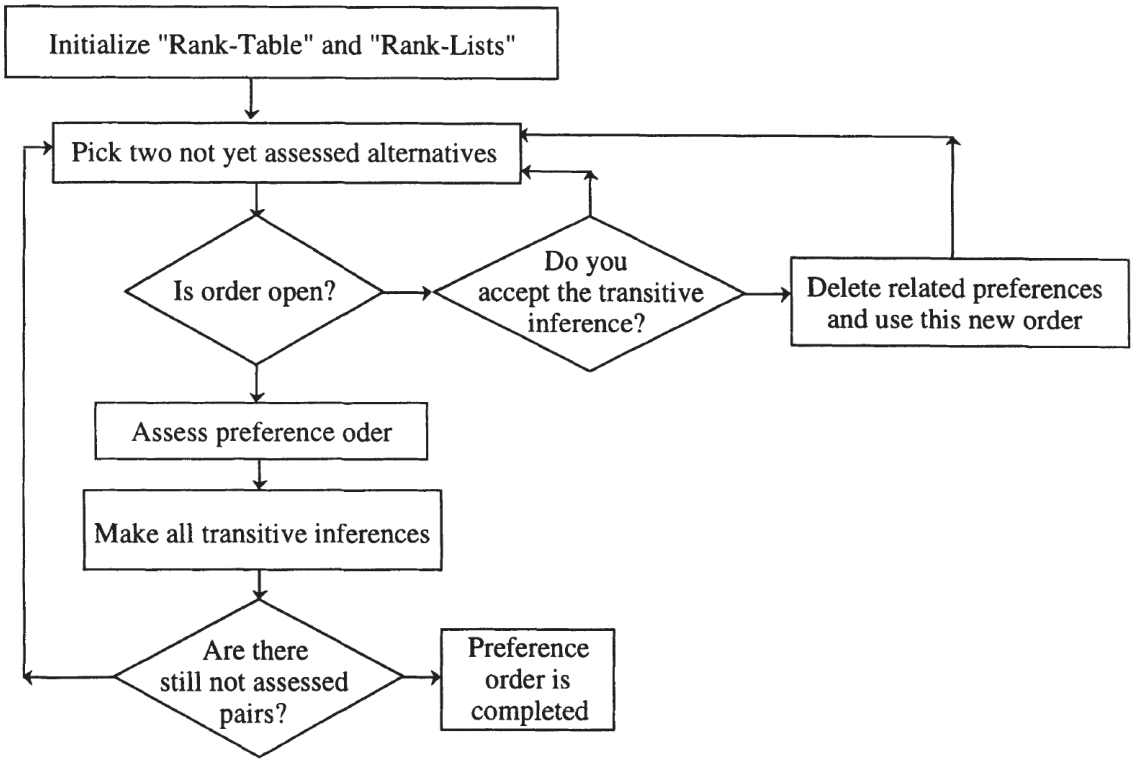


Figure 10: Resolution model for complete strong preference ordering (Beroggi, 2013)

2.3.3 Dominance Relation vs. Preference Relation

Dominance relation is usually too weak to compare all the alternatives. For example, on Figure 2.3.3, comparison between x and y (and the rest of blue points) is not possible using only dominance relation. They are not dominated by any of the alternatives. That is why to create complete comparison, preference relation is required.

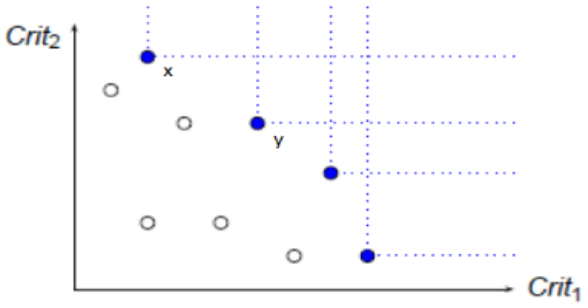


Figure 11: Graph showing relations between objects, criterion 1 and 2 are gain-type.

Dominance Principle: an object *x* dominating object *y* on all considered criteria (i.e. *x* having evaluations at least as good as *y* on all considered criteria) should also dominate *y* on the decision (i.e., *x* should be assigned to at least as good decision class as *y*) (Dembczynski et al., 2010).

Greco et al. (1995) addressed the dominance principle's inconsistency using classical rough set approach with taking into account preference orders and monotonic relationships between evaluations on criteria and assignment to decision classes. This generalized approach is named as Dominance based Rough Set Approach (DRSA). Dominance-based rough set approach is expansion of Pawlak's rough set approach. Pawlak's rough set model is based on indiscernibility relation which are used to deal with the regular attributed such as symptoms, colors and textural features. The major disadvantage of indiscernibility relation based rough set approach is that it is not able to discover inconsistencies coming from the consideration of criteria i.e. attributed with preference-ordered domains (scales), such as product quality, market share, and debt ratio. Examples of Dominance based Rough Set Approaches are (Yang & Yang, 2012): Expanded Dominance-based Rough Set in Incomplete Information System with "*" unknown Values (For example, Valued Dominance-based Fuzzy Rough Set Approach, ↑ and ↓ Descriptors and Certain Rules, Limited Dominance-based Rough Set Approach etc.) and Expanded Dominance-based Rough Set in Incomplete Information System with "?" unknown Values (For example, Similarity Dominance-based Rough Set and Approximate Distribution Reducts, Similarity Dominance-based Rough Sets in Fuzzy Decision System etc.)

2.3.4 Probabilistic Decision Rules

Probabilistic rules are concordant with the VC-DRSA. Each rule is characterized by a confidence ratio, representing the probability that an object matching the left hand side (LHS) of the rule matches also its right hand side (RHS). Probabilistic decision rule semantics: "*if (LHS), then (RHS) in (X)% of cases*"

Let us remark that the probabilistic decision rules are very useful when large data tables are considered. In large data tables, an ambiguity typically exists and prevents finding some very strong patterns because the certain decision rules are contradicted by the ambiguous examples. Probabilistic decision rules, permitting a limited number of counterexamples, may represent these strong patterns.

2.3.5 Preference Graphs

The preference of each training sample can be presented in the form of a directed graph, known as a preference graph (Dekel et al., 2004; Aiolli & Sperduti, 2005). The nodes in preference graphs are

labeled as: relevant, related or irrelevant. A node is considered *relevant* if it is important or pertinent to the context for graph. It is considered to be *related* if it is connected partially (more than acceptable limit) with the situation or context for graph. It is considered as *irrelevant* if it is not pertinent with context for graph. Now, different types of preference graphs are shown in figure (Dekel et al., 2004). Based on labels, preference graphs can be classified into four categories:

- *Standard Multiclass single-label categorization*: In this categorization, a directed edge points from the (single) relevant label to each of the *k-1* irrelevant labels as shown in Figure 12(a).
- *Standard Multiclass multilabel categorization*: In directed bipartite graph, relevant labels constitute one side of the graph and the irrelevant labels the other side. There is a directed edge from each of the relevant label to each irrelevant label as shown in Figure 12(b).
- *A multi-layer graph with hierarchical multiclass settings:* In this category, a multi-layer graph is presented in which there are three levels of label "goodness", useful for instance in hierarchical multiclass setting.
- *General (possibly cyclic) preference graph with no predefined structure:* In this category, there is no clear bifurcation of relevant and irrelevant labels.

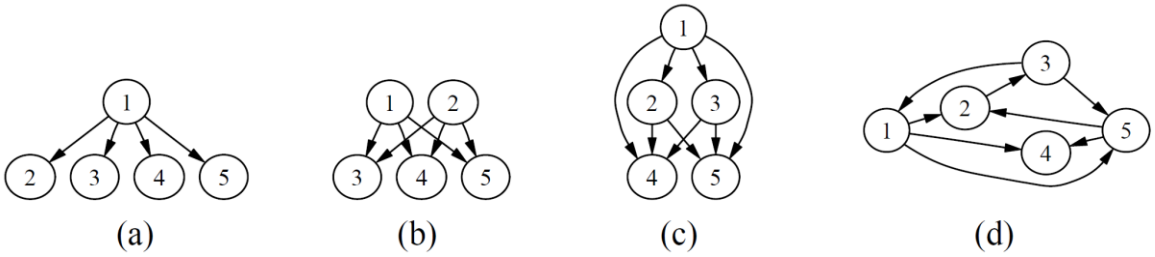


Figure 12: Types of preference Graphs: (a) multiclass single-label categorization where 1 is the correct label, (b) multiclass multilabel categorization where {1, 2} is the set of correct labels, (c) A multi-layer graph that encodes three levels of label "goodness", useful for instance in hierarchical multiclass settings, (d) general (possibly cyclic) preference graph with no predefined structure (Dekel et al., 2004)

Examples of preference graphs with no predefined structure

In this subsection, two preference graphs (oriented preference graph and network or weighted preference graph) with no predefined structure, and types of preference order are explained as follows.

Oriented Preference Graphs

An oriented preference graph visualizes the preferences between elements (such as alternatives, criteria and decision makers). A graph consists of two sets: (i) set of all elements (nodes or vertices), and (ii) set of all preferences between these elements (links or edges).

Let A={a_j | j = 1,2,…, n} represents a set of alternatives and L={l_{jk} | $a_j \succ a_k$ (strong) or $a_j \succcurlyeq a_k$ (weak)} represents set of all preference relations. A preference graph can be represented by an adjacency matrix using following rules: $l_{jk} = 1$ and $l_{kj} = 0$ if $a_j \succ a_k$, and $l_{jk} = " - "$ and $l_{kj} = " - "$ if the relation between the two alternatives is not defined. Let L = { $a_1 \succ a_5$, $a_1 \succ a_6$, $a_2 \succ a_5$, $a_3 \succ a_4$, $a_3 \succ a_5$, $a_4 \succ a_2$, $a_6 \succ a_3$} be the set of strong preference relations of the six alternatives. The adjacency matrix and the corresponding strong preference graph if shown as follows:

$$\begin{bmatrix} - & - & - & - & 1 & 1 \\ - & - & - & 0 & 1 & - \\ - & - & - & 1 & 1 & 0 \\ - & 1 & 0 & - & - & - \\ 0 & 0 & 0 & - & - & - \\ 0 & - & 1 & - & - & - \end{bmatrix}$$

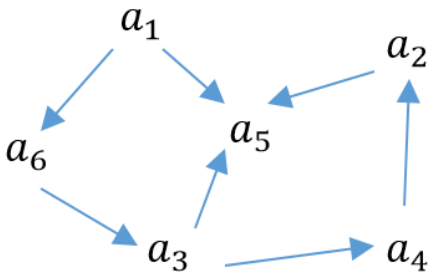


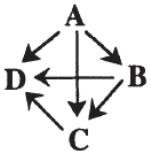
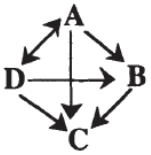
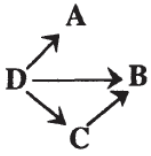
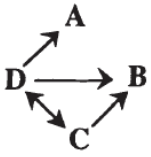
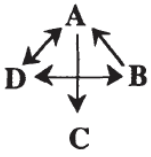
Figure 13: Adjacency Matrix and corresponding preference graph (Beroggi, 2013)

Preference Order in Preference Graphs

There are two types of preference graphs: *complete* and *partial*. In *complete preference graph*, every alternative must be connected with every other alternative. Whereas, in partial preference graph, every alternative may or may not be connected with other alternatives. In both types of preference graphs, following three types of preference orders are possible (as shown in Table 7 also):

- *Strong preference order*: It is possible to rank all elements from most preferred to least preferred order without ties.
- *Weak preference order*: It is possible to rank all elements from most preferred to least preferred order with ties.
- *Pseudo order*: It is not possible to rank elements.

Table 7: Preference orders in preference graphs with examples (Beroggi, 2013)

	$\succ$: strong preference order	$\succsim$: weak preference order	pseudo order
complete	 rank: A$\succ$B$\succ$C$\succ$D	 rank: (A$\sim$D)$\succ$B$\succ$C	 rank: unclear
partial	 ranks: D$\succ$A, and D$\succ$C$\succ$B	 ranks: (D$\sim$C)$\succ$B, and D$\succ$A	 rank: unclear

Network or Weighted Preference Graphs

Network or Weighted preference graph is a type of intensity measure graph in which vertices are assigned weights. Figure shows an example of weighted adjacency matrix and the corresponding preference graph for the following preference relations among six alternatives: L={$\pi(a_1) = 6\pi(a_3)$, $\pi(a_1) = 24\pi(a_5)$, $\pi(a_1) = 2\pi(a_6)$, $\pi(a_2) = 2\pi(a_5)$, $\pi(a_3) = \pi(a_4)$, $\pi(a_3) = 4\pi(a_5)$, $\pi(a_4) = 2\pi(a_2)$, $\pi(a_4) = 4\pi(a_5)$, $\pi(a_6) = 3\pi(a_3)$, $\pi(a_6) = 12\pi(a_5)$}

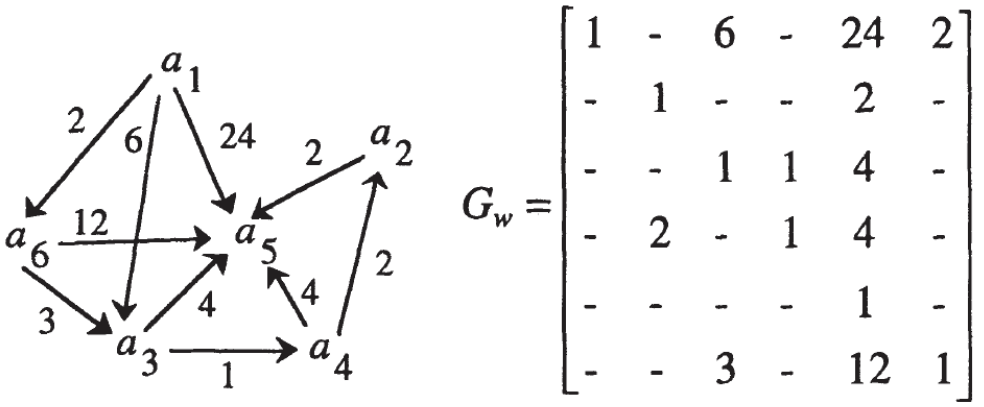


Figure 14: Weighted preference graph and corresponding adjacency matrix (Beroggi, 2013)

2.4 Ranking Methods

Ranking problems are within the interest of two distinct scientific communities: PL and MCDA oriented. PL community focuses attention on the methods reconstructing preferences from existing rankings in other words in learning to rank, usually without a human in the learning loop. MCDA focuses on taking into account DM preferences, that can be collected interactively. However, the general aim is the same - to provide meaningful ordering of the considered items (object, alternatives, data samples).

2.4.1 Preference Learning Perspective on Ranking Problems

Figure shows an example of document retrieval system to describe the purpose of ranking methods. A set of documents D = {d_1, d_2, … , d_N} are stored in this system and a query 'q' is placed to retrieve these documents. After executing the query, documents are received in a specific order {$d_{q,1}$, $d_{q,2}$, … ,d_{q,n_q}}.

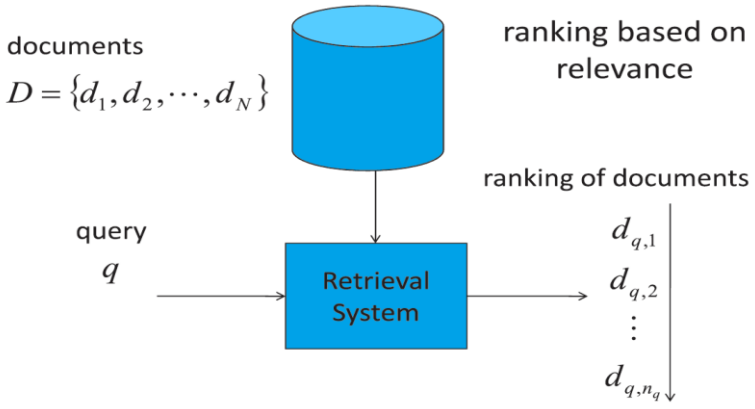


Figure 15: Ranking based document retrieval system (Li, 2011)

In literature, various frameworks for learning to rank from preferences are proposed (Fürnkranz & Hüllermeier, 2010; 2010a;2010b) Aioli & Sperduti, 2005; Aioli et al. 2007). For example, Aiolli and Sperduti (Aioli & Sperduti, 2005; Aioli et al. 2007) proposed the framework as unifying framework to model and solve a large class of multiclass problems in a large margin perspective. Fürnkranz, & Hüllermeier (2010) proposed another widely accepted framework in which ranking problems are categorized into object ranking, instance ranking and label ranking. These ranking problems are

explained as follows (Fürnkranz & Hüllermeier, 2010):

- *Instance Ranking:* In this PL problem, an absolute preferences of a set of data samples (Objects) are made over an ordinal scale. Figure 16 shows an example of instance ranking rated on an ordinal scale such as {bad, medium, good}. The instance ranking problem (Fürnkranz & Hüllermeier, 2010) can be formalized as follows:

  Given:

  - A set of training instances $\{x_l \mid l=1,2, \dots, n\} \subseteq X$ (each instance typically though not necessarily represented by a feature vector)
  - A set of label $Y = \{y_1, y_2, \dots, y_k\}$ endowed with an order $y_1 < y_2 < \dots, < y_k\}$
  - For each training instance x_l an associated label y_l

  Find:

  - A ranking function that allows one to order a new set of instances $\{x_j Y_{j=1}$ according to their (unknown) preference degrees

  Performance Measures:

  - The area under the ROC-curve (AUC) in the dichotomous case (k=2)
  - Generalizations such as the C-index in the polychotomous case (k>2)

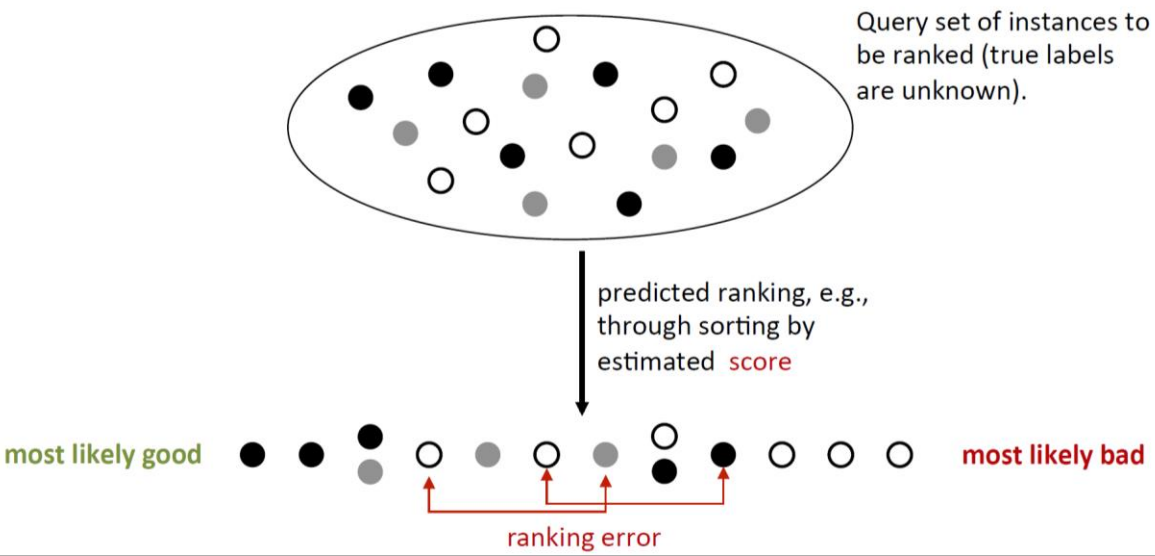


Figure 16: Instance Ranking Example (Fürnkranz & Hüllermeier, 2011)

- *Object Ranking:* This is another type of PL problem in which a set of rankings or pairwise comparisons (partial orders) are specified over a set of data samples (Objects) [12]. In object ranking, objects are characterized by properties in terms of an attribute-value representation before ordering. Here, aim is to identify a learning function that accepts a set of objects as input, predicted the order among them and produces ordered objects as output. For example: given a set of books as input, preference learning function may order the books (objects) in accordance with most purchased or read. The object ranking problem (Fürnkranz & Hüllermeier, 2010) can be formalised as follows:

  Given:

- A (potentially infinite) reference set of objects Z (each object typically though not necessarily represented by a feature vector)
- A finite set of pairwise preferences $x_i \succ x_j$, $(x_i, x_j) \in Z \times Z$

Find:

A ranking function f(.) that assumes as input a set of objects and returns a permutation (ranking) of this set. Typically, an order O_k induces a preference relation among set of objects as follows:

$$\forall x_i, x_j \in x_k \{(x_i \succ x_j) \in O_k, or\ (x_i \prec x_j) \in O_k, or\ (x_i \equiv x_j) \in O_k$$

Where, $x_i \succ x_j$ means that x_i has higher preference ranking as compared to x_j and $x_i \equiv x_j$ means that both $x_i\ and\ x_j$ have same preference ranking.

- *Label Ranking* (Fürnkranz & Hüllermeier, 2010): In label ranking, data objects typically consists of an instance $X=\{x_i\}$ and an associated finite set of class labels $Y= \{y_i\}$ i=1,2, …, k}. The goal is to learn a label ranker in the form of X à S_y mapping, where the output space S_y is given by the set of all total orders (permutations) of the set of label Y. For example, a set of books is an input to label ranker function and output will be an order of books according to user preferences. The label ranking problem can be formalised as follows:

  Given:

  - A set of training instances $\{x_l \mid l=1,2, \dots ,n\} \subseteq X$ (each instance typically though not necessarily represented by a feature vector)
  - A set of label $Y = \{y_i \mid i = 1,2, \dots, k\}$
  - For each training instance x_l: a set of pairwise preferences of the form $y_i \succ_{x_l} y_j$

  Find:

  - A ranking function that maps any $x \in X$ to a ranking $\succ_x$ of Y (permutation $\pi_x \in S_k$)

  Performance Measures:

  - Ranking error (e.g., based on rank correlation measures) comparing predicted ranking with target ranking
  - Position error comparing predicted ranking with a target label

2.4.2 DRSA Approach For Ranking Problem

The application of DRSA to the choice or ranking problems proceeds as follows. First, the DM gives some examples of pairwise comparisons with respect to some reference objects, for example a complete ranking from the best to the worst of a limited number of objects – well known to the DM. From that, Pairwise Comparison Table is made.

Let us suppose that the DM expresses her preferences with respect to pairs (x, y) from $E \subseteq BxB, |E| = m$, where B is a set of reference objects $B \subseteq A$. These preferences are represented in an $m\ x\ (n + 1)$ Pairwise Comparison Table (Figure 17). The m rows correspond to

the pairs from E. For each $(x, y) \in E$ in the corresponding row, the first n columns include information about preferences $P_i[g_i(x), g_i(y)]$ on particular criteria from set C, while the last column represents the comprehensive preference $P(x, y)$.

Pair(x, y) of warehouses	$P_1[g_1(x), g_1(y)]$	$P_2[g_2(x), g_2(y)]$	$P_3[g_3(x), g_3(y)]$	S or S^c
(1,2)	0	1	0	S
(4,7)	-2	0	-1	S^c
(6,3)	-1	-1	0	S^c
...	...	...	...	...

Figure 17: Pairwise Comparison Table (PCT) (Figueira et al.(eds.), 2005)

Then, calculation of lower and upper approximations of outranking relation S and non-outranking relation S^c is made. Approximations should be calculated according to DRSA (see Figure 18 for details).

$$\underline{P}(S) = \left\{(x,y) \in B : D_P^+(x,y) \subseteq S\right\}$$
$$\overline{P}(S) = \bigcup_{(x,y)\in S} D_P^+(x,y)$$
$$Bn_P(S) = \overline{P}(S) - \underline{P}(S)$$
$$\underline{P}(S) \subseteq S \subseteq \overline{P}(S)$$

$$\underline{P}(S^c) = \left\{(x,y) \in B : D_P^-(x,y) \subseteq S^c\right\}$$
$$\overline{P}(S^c) = \bigcup_{(x,y)\in S^c} D_P^-(x,y)$$
$$Bn_P(S^c) = \overline{P}(S^c) - \underline{P}(S^c)$$
$$\underline{P}(S^c) \subseteq S^c \subseteq \overline{P}(S^c)$$

Figure 18: Upper and lower approximations of outranking and non-outranking relations

In the next step, a preference model in terms of "if ..., then..." decision rules (cf. Section 2.3.4) is induced. These rules are applied to a larger set of objects and preference graph can be build. A proper exploitation of the results so obtained (see Section 2.4.2) gives a final recommendation for the decision problem at hand.

2.4.3 Ranking Cost Functions - Exploitation of Preference Graphs

To exploit the preference structure in order to obtain a final ranking, ranking cost function has to be chosen. One of the most popular function used is Net Flow Score (NFS).

In fuzzy-sets, to acquire NFS, positive and negative flows have to be produced. The positive (or leaving) flow measures the average degree to which an action is preferred to the other ones. Actions with a larger leaving positive flow value should be ranked first. The negative (or entering) flow measures the average degree to which the other actions are preferred to that action. Thus actions with a smaller negative flow value should be ranked first. Usually both preference flows lead to somewhat different rankings as in a multi-criteria context there is usually no ranking completely consistent with all the pairwise comparisons results. The net flow is the balance between the positive and the negative flows. It can be used to rank all the actions from the largest positive values to the most negative ones.

In DRSA, Net Flow Score can be calculated in following way:

$$S_{nf}(x) = S^{++}(x) - S^{+-}(x) + S^{-+}(x) - S^{--}(x),$$

where:

$S^{++}(x)$ = card($\{y \in A$: there is at least one decision rule which affirms $xSy\}$),

$S^{+-}(x)$ = card($\{y \in A$: there is at least one decision rule which affirms $ySx\}$),

$S^{-+}(x)$ = card($\{y \in A$: there is at least one decision rule which affirms $yS^cx\}$),

$S^{--}(x)$ = card($\{y \in A$: there is at least one decision rule which affirms $xS^c y\}$)

The recommendation in ranking problems consists of the total preorder determined by $S_{nf}(x)$ on A. (Figueira et al., 2005)

The "Repeated Net Flow Rule" (RNFR, RNFS) ranks in first position the elements with the highest Net Flow score in A. These elements are then removed from A and the Net Flow scores of the remaining alternatives are computed in the reduced set. Alternatives with the highest Net Flow score in the reduced set are then ranked in second position and so on. (Bouyssou, 1997)

In fact, NFS and RNFS can be modified in various ways. Preprocessing the preference graph can be made, either by calculating transitive closure of S and S^c relations or by first taking the asymmetric part of S and S^c, and only then calculating the transitive closure of both relations. In case of relation S, calculation of the asymmetric part boils down to the following rule:

$$if\ (x,y)\ \in S\ and\ (y,x) \in S\ \ then\ remove\ both\ arcs\ from\ the\ graph$$

In case of relation S^c, calculation of the asymmetric part boils down to the following rule:

$$if\ (x,y)\ \in S^c\ and\ (y,x) \in S^c\ \ then\ remove\ both\ arcs\ from\ the\ graph$$

The actions are then ranked from the highest (best) score down to the lowest.

2.5 Machine Learning in Ranking Problems

Machine learning for the rankings problem is usually applied if the sufficient amount of reference rankings that can be used in the learning process is present. The aim is to derive the ranking functions based on the content of the training set. The methodology is called learning-to-rank or machine-learned ranking.

2.5.1 Introduction to MLR

Learning-to-rank or Machine-Learned Ranking (MLR) is a type of supervised or semi-supervised machine learning problem. In this problem, ranking model is automatically constructed from training data (Pedersen, 2008). Training data contains lists of items. Each of these lists have some partial order specified between items. An ordinary, numerical or binary judgement can be induced for partial order.

2.5.2 Key Components of Machine Learning Framework

The key components of Machine Learning are explained as follows (Mitchell, 1997):

- input space: contains objects under investigation.
- output space: contains learning target with respect to the input objects.
- hypothesis space: defines the class of functions mapping the input space to the output space.
- Loss function: measures to what degree the prediction generated by the hypothesis is in accordance with the ground truth label.

The relationship between these four key components is pictorially represented in Figure 19. A detailed description can be found here (Mitchell, 1997).

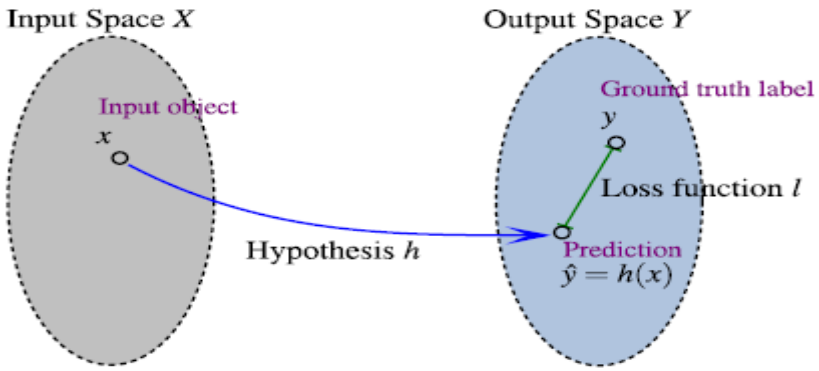


Figure 19: Key components of Machine Learning Framework (Mitchell, 1997)

2.5.3 MLR Framework

Figure 20 shows Learning-to-rank or machine-learned ranking framework. Since, MLR is a supervised learning mechanism thus a training set is required. A typical training set consists of training queries (i.e. q_i , i=1,2…n), associated documents represented by feature vectors ($x^{(i)} = \{x_j^{(i)}\}_{j=1}^{m^{(i)}}$ (where, $m^{(i)}$ is the number of documents associated with query q_i) and the corresponding relevance judgements.The learning system apply the learning algorithm to learn the ranking such that the output of the ranking model can predict the ground truth in the training set as accurately as possible. The accuracy is measured using loss function. A high accuracy is expected when there is minimum loss. In Testing phase, a new query comes in and the model learned in the training phase is applied to sort the documents and return the corresponding ranked list to the user in response to his/her query.

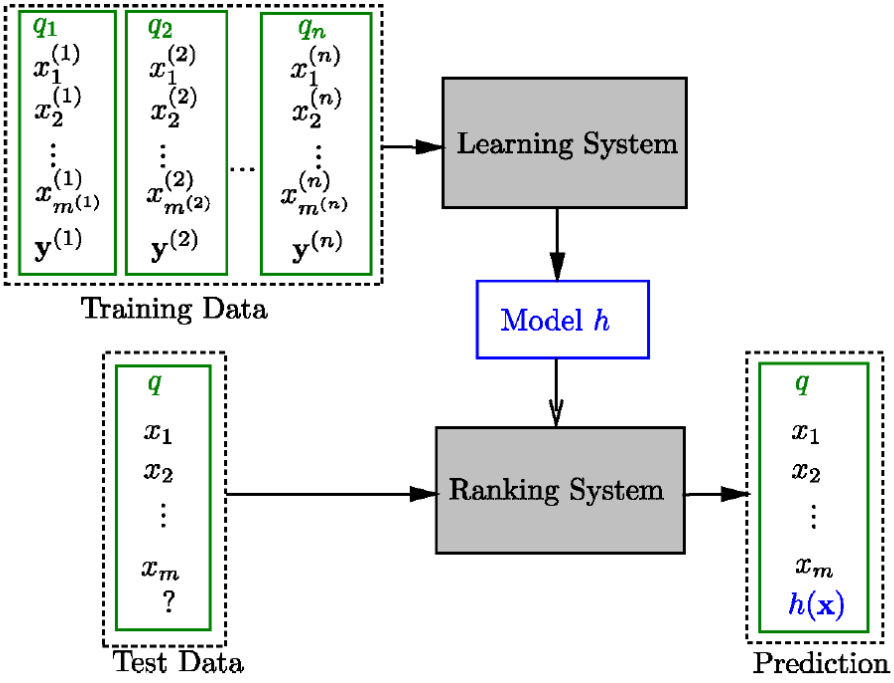


Figure 20: MLR Framework (Liu, 2009)

2.5.4 Overview of MLR Approaches

Different learning to rank approaches uses different input, output, hypotheses or loss functions but are mainly classified on these four pillars of machine learning. According to (Liu, 2009), MLR methods can be classified into three categories:

- **Pointwise Approach:** In this approach, single document is processed at a time in the loss function. This single document is taken as input to classifier for predicting how much relevant it is for the current query. The result list of this document scores helps in predicting the final ranking. In this approach, the score for each document is independent of others. Pointwise approaches are classified into following three categories:
  - Regression-based algorithms: example: subset ranking with regression
  - classification-based algorithms: example: binary classification for ranking, multi-class classification for ranking etc.
  - ordinal regression-based algorithms:example: perceptron-based ranking (PRanking), ranking with large margin principles, ordinal regression with threshold-based loss functions etc.
- **Pairwise Approach**: In this approach, a pair of documents are processed at a time in the loss function. Here, learning a binary classifier tells which document is better in a given pair. The goal of ranker is to minimize the number of inversions in ranking. Pairwise approaches are preferred over pointwise approaches because predicting relative order is closer to the nature of ranking than predicting class label or relevance score. Examples of Pairwise approaches include:
  - Ordering with preference function, SortNet:Neural network-based sorting algorithm, RankNet: Learning to rank with gradient descent, FRank: Ranking with a fidelity loss, Rankboost, Ranking SVM, GBRank

- **Listwise Approach:** In this approach, a set of documents is taken as input and produces the optimal ordering for it as output. There are two main categories of listwise learning approach: (i) Direct optimization of information retrieval (IR) measures such as Normalized Discounted Cumulative Gain (NDCG). For example, SoftRank (Taylor et al., 2008) and AdaRank (Xu & Liu, 2007), (ii) Minimize a loss function (such as ListNet (Cao et al., 2007) and ListMLE (Lan et al., 2014)). Minimization of loss function is further classified into following categories:
  - Minimization of measure-specific loss (measure approximation, bound optimization, non-smooth optimization)
  - Minimization of Non-measure-specific loss (ListNet, ListMLE, Ranking using comulative distribution networks, BoltzRank)

2.5.5 Comparative Analysis of MLR Approaches

A comparative analysis of MLR approaches using four pillars of machine learning (input space, output space, hypothesis space and loss function) is shown in Figure 21.

Category	Pointwise		
	Regression	Classification	Ordinal regression
Input space	Single document x_j		
Output space	Real value y_j	Non-ordered category y_j	Ordered category y_j
Hypothesis space	$f(x_j)$	Classifier on $f(x_j)$	$f(x_j)$ + thresholding
Loss function	$L(f; x_j, y_j)$		

Category	Pairwise	Listwise	
	–	Non-measure-specific	Measure-specific
Input space	Document pair (x_u, x_v)	Set of documents $\mathbf{x} = \{x_j\}_{j=1}^{m}$	
Output space	Preference $y_{u,v}$	Ranked list π_y	
Hypothesis space	$2 \cdot I_{\{f(x_u) > f(x_v)\}} - 1$	$sort \circ f(\mathbf{x})$	
Loss function	$L(f; x_u, x_v, y_{u,v})$	$L(f; \mathbf{x}, \pi_y)$	

Figure 21: Comparative analysis of Pointwise, Pairwise and Listwise approaches (Liu, 2009)

3 Meta-Alerts Prioritisation in PROTECTIVE

The key factors for prioritisation and ranking of meta-alerts are as follows:

1. Identification of direct and derived criteria that can be extracted from meta-alerts;
2. Provisioning of mechanisms for selection and derivation of criteria;
3. Construction of information table from sets of meta-alerts on the basis of selected criteria;
4. Preference learning, derivation of decision rules;
5. Generation of ranking.

3.1 Meta-Alert Definition

Meta-alert (MA), by intuition, is a structure that contains information from multiple alerts (but most probably only a relevant subset of that information). More formally, following Kruegel et al. (2005), the meta-alert is a result of the merging of two or more related alerts, merged as a part of the alert correlation process. MAs are objects consisting of all common attributes of correlated alerts and MAs, a list of data sources and links to original alerts and meta-alerts, as well as additional data derived, in the enrichment process, as a function of the attributes of the merged alerts (cf. Figure 22). It is worth to note that from technical point of view attributes and their values corresponds to keys and values in IDEA format. The general purpose of meta-alerts is to aggregate information of related attacks and at the end present a single alert instance that summarizes all the relevant information to a human analyst.

Attribute (key)	Value
ID	unique identifier of MA
AssetID	identifier of the local asset involved
AssetCriticality	Asset Criticality Score provided by AIT
TrustScores	table of Trust Scores for data sources of combined alerts (partially provided by TUDA)
SourceAlerts	References to alerts combined within MA (actual method of implementation to be determined)
Common attribute A_1	V_1 [values determined/extracted in correlation process]
…	…
Common attribute A_n	V_n [values determined/extracted in correlation process]

Figure 22: General concept of meta-alert (MA)

The crucial matter is the representation of meta-alert concept within the PROTECTIVE. Taking into account already made technological choices it is natural to build the representation on top of IDEA format.

For the representation of the source alerts the following key-values pairs already defined in IDEA Schema (https://idea.cesnet.cz/en/definition) can be used:

- **CorrelID**: array of ID
  Identifiers of messages, which are information sources for creation of this message in case the message has been created based on correlation/analysis/deduction of other messages.

- **AggrID**: array of ID
  Identifiers of messages, which are aggregated into more concise form by this message. Should be sent mostly by intermediary nodes, which detect duplicates, or aggregate events, spanning multiple detection windows, into one longer.
- **PredID**: array of ID
  Identifiers of messages, which are obsoleted and information in them is replaced by this message. Should be sent only by detection nodes to incorporate further data about ongoing event.
- **RelID**: array of ID
  Otherwise related messages.

The data collected through the enrichment processes can be incorporated into the IDEA as custom defined extensions – an additional attributes e.g.:

- **AssetCriticality** – for example the criticality of the host, from the perspective of business processes, depicted on an ordered categorical scale such as "Small, Medium, Large" or an ordinal scale from e.g. 1 to 5 (cf. D4.1).
- **AttackSourceIPReputation** – some numerical value
- **TrustScore** – trust to the source of information, for example in percent.

More data supporting prioritisation, rankings, and handling of meta-alerts can be stored in separate objects or computed on the fly, please see the next section (Section 3.2) for more details. It is not wise to place into meta-alerts data that frequently changes over time when the decision should be made on the basis of current values, not the one from the creation time of meta-alert.

3.2 Identification and Selection of Decision Criteria

Let us recall the definition of criteria as the starting point of further considerations. Criterion is an attribute with values **ordered** according to a scale of preference introduced (by a decision maker) as a part of domain knowledge.

Raw alerts, represented in IDEA, contains only limited number of attributes (key-value pairs) that can be directly treated as criteria. Most of the valuable information that can serve as criteria is gathered during enrichment phase of the processing. However, some statistical data collected during aggregation and correlation process are also suitable for consideration.

The ranking and prioritisation process does not have to use all available criteria; it is assumed that the system will allow choosing a set of relevant criteria for MCDA procedures.

Following keys (from IDEA key-value pairs) and derived data can serve as the criteria:

- Time-related keys: DetectTime, EventTime, CeaseTime and derived values e.g. period of time from DetectTime or time remaining to due-date computed as sum of CreateTime and requested (by internal or external policies) reaction time,
- Category through assignment to ordered classes of importance or severity (such as e.g. port scan ->low importance, brute-force ->medium importance, DDoS ->high importance),
- data derived from the IP4/IP6 address of attack source e.g. reputation of host or ASN,
- data derived from the IP4/IP6 address of attack target e.g. asset's criticality (AssetCriticality)

- the confidence level of the reporting Node,
- the trust level of the reporting Node (TrustScore),
- volumetric data such as: ConnCount, FlowCount, PacketCount, ByteCount,
- number of unique reporting Nodes in the case of Meta-alerts composed of multiple alerts,
- number of correlated alerts (cardinality of CorrelID or AggrID),
- more criteria may be identified later on, during pilot phases.

To each criterion, DM should be able to define and assign a scale of preference and type of criterion (cost or gain). The way of defining of the scales of preference depends on a nature of criteria: cardinal (with values expressed on some interval or ratio scale) and ordinal (with values expressed on some ordinal scale). For the sake of clarity, let us mention that the remaining attributes are called regular attributes (with values on some nominal scale) and are not used to build preference model. The criteria will be grouped into criteria sets in order to relate them with particular preference model build for particular decision maker.

The criteria extracted from meta-alerts serve as the basis for construction of information table that is required to conduct MCDA and build preference based decision model being used to generate a ranking. An information table (see Table 8) is a two-dimensional array consisting of columns tagged by criteria names and ID. The number and kind of criteria is variable, thus it is reasonable to store information tables in NoSQL database (e.g. MongoDB).

Table 8. Example of information table

ID	Time to due-date	AssetID	AssetCriticality	TrustScore	Category Priority
MA-ID1	10 min.	AS-ID100	...	...	High
MA-ID2	5 min.	AS-ID200	...	...	Medium

A representation of criterion should include the following data:

- **Name**: string
  Name of criteria usually corresponds to key name in meta-alert. However, there could be derived criteria, e.g. CategoryPriority
- **RelatedKeyInMetaAlert:** string
- **Derived**: Boolean
  If the criterion is not derived then 0, otherwise 1. The derived criteria have values assigned according to mapping table that transforms values from domain of attributes values into domain of criterion
- **Type**: gain|cost
  Definition of type in case of meta-alerts is a little bit tricky. The gain type means that the higher is the value of criterion then the higher urgency of the meta-alert. The cost type means that the higher is the value of criterion then the lower urgency of meta-alert. Thus, time to due-date will be the cost type (the lowest value, the higher position in ranking). AssetCriticality should be the gain type (the higher value, the higher position in ranking)
- **ScaleType**: cardinal | ordinal

- **Scale**: object
  Within the scale object the data type is provided (integer, real, string, time), the range of values (e.g. <0-1>) or list of values for ordinal criteria
- **Units**: string
  Optional definition of units that should be used for criterion values e.g. percent, minutes etc.
- **MapForDerivedCriterion**: object | function.
  The map providing transformation of values from meta-alerts attributes domain into criterion domain. Multiple values from meta-alert attributes could be mapped into the same value in criterion domain. The mapping should allow to map the ranges of values into single value in criterion space and also to provide default value for the values that are not explicitly given. In some cases it is reasonable to provide mapping as a function, e.g. to compute distance of attribute value from a cut-off or time dependent value - good example here is calculation of time remaining to due date where current time and handling policy must be taken into consideration.

3.3 Approaches for Gathering Preferences – Technical Perspective

To apply DRSA, or other MCDA or PL technique for ranking, one needs to provide or collect results of multiple pairwise comparisons in each case depicting which of two meta-alerts is more important for particular DM. Meta-alerts, in that case, are the rows from information table (cf. Section 3.2), in other words, the vectors of attributes with explicitly indicated criteria. The results of comparisons are analysed per each DM separately. Thus the system must be able to identify DMs uniquely.

For example, the fact that MA-ID2 is more important from MA-ID1 for decision maker DM1 can be expressed in form of the following tuple: (MA-ID2,MA-ID1,DM1). For greater clarity we may add additional position denoting relation ('+' if first item outperforms (outranks) the second one and '-' in the other case, additionally we can extend the notation for the case of non-outranking - let's denote that case as '0'): e.g. (MA-ID1,MA-ID2,-,DM1) or (MA-ID2,MA-ID1,+,DM1). Non-outranking can take place when meta-alerts are incomparable for the DM. Equality relation can be represented as two tuples without necessity of providing additional markings.

The ultimate question is how to get the relevant results of comparisons without inflicting too much additional work on the DM, in our case operator of the system (cybersecurity officer, incident handler). Two possible solutions arise. The first one is to make use of meta-alerts (corresponding to incidents) handling history and deduce preference data for the selected pairs of meta-alerts. The second is to generate sets of representative vectors of attributes and ask DM to compare and order them at least partially.

3.3.1 Deriving Preference Data From Meta-Alerts (Incidents) Handling History

The process of deducing preferences, in form of pairwise comparisons, from the history of alerts handling is the multi-step procedure. In general, the following steps can be distinguished:

1. Filter meta-alerts handled by particular DM.
2. From the result of step 1, select sets of alerts that appeared in the system nearly at the same time (period between their arrival times (CreateTime) is below some threshold), preferably in the moment when the DM was not logged in. The last condition aims is increasing probability that the identified alerts were exposed to DM at the same moment of time.
3. Pair the alerts within each set from step 2 (e.g., generate all possible pairs).
4. Assume that the preference corresponds to the handling order.

To conduct the procedure the following data must be registered in the system:

- login and logout times of DM,
- information who started handling the alert,
- time when the handling process started,
- time of arrival of the alert,
- information about statuses of the alerts.

Data about handling history, alert statuses and user (DM) activities could be stored in the data structures independent from meta-alerts e.g. in relations under control of RDBMS.

3.3.2 Interactive Learning by Pairwise Comparisons

To provide the functionality of interactive gathering of preferences a few problems have to be solved: 1. How to select or generate pairs of feature vectors (vectors of attributes/rows for information table) to be presented to DM? The feature vectors consist only of criteria selected by DM as relevant for him or her. 2. How to present those vectors to DM? 3. How to collect the DMs preference for presented vectors?

The first problem can be solved in multiple ways. For example, when the historical data are available, then the sets can be generated by choosing a representative set randomly from historical data. It can be done quite easily. However, it would be wise to take into account distribution of feature vectors available in historical data. To achieve that goal the clustering techniques can be applied and feature vectors selected in the way covering all or at least most of the clusters. In the case, when there is no historical data, the sets of vectors for learning can be generated randomly and clustered. The clusters representatives should serve as learning vectors. Each value in the generated vector must be consistent with provided scales of components' attributes and if possible mimic their natural distribution.

The vectors should be presented to DM in limited number at one time not to overwhelm him or her. In the basic setup, it could be just pairs of vectors. However, presentation of larger sets is also worth to consider. The vectors of feature could be, for example, presented in the form of circles with values of criteria (cf. Figure 23) or as tables. Let's focus on the first form as visually more attractive.

The preferences could be collected by allowing DM to provide partial order between presented objects (see Figures 23 and 24) – in that case, pairwise comparisons can be derived from preference graph and transformer to PCT, or by collecting data about pairwise comparisons directly (Figure 25 and 26) - four types of relations can be collected that way, and lastly asking DM to rank the vectors manually (Figure 27).

The process should be conducted iteratively by showing the DM different sets of items in each iteration. Of course, sets shown in consecutive iterations can have items in common to allow construction of a larger preference graph.

Figure 23: Meta-alerts provided to DM for marking preferences.

Figure 24: Preference graph defined by DM.

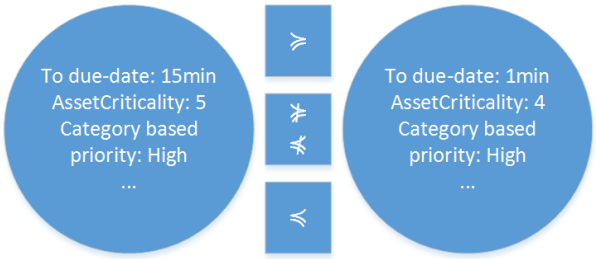


Figure 25: Meta-alerts provided to DM for pairwise comparison. DM has three options to choose from: left precede right one, right precede left one, or left and right are incomparable for DM. In the case of equality DM can select first and third options simultaneously.

Figure 26: Four possible responses of DM.

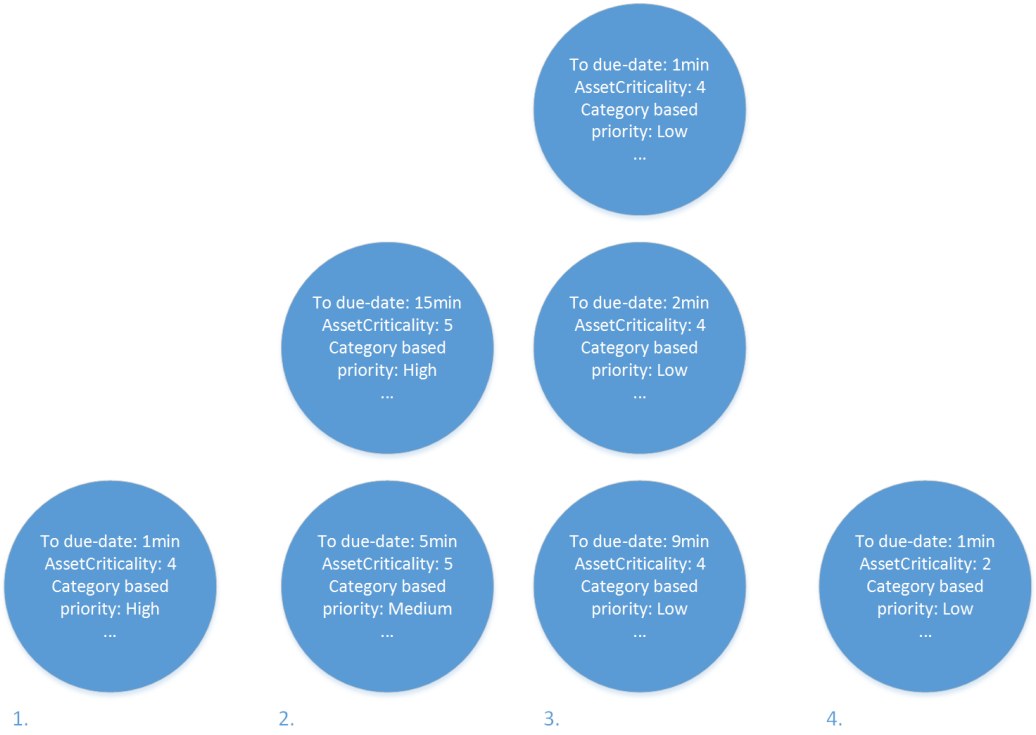


Figure 27: Example ranking provided by DM in process of gathering preferences.

3.4 Representation and Storage of Preference Model

During the process of building preference model, several kinds of data have to be collected and stored. Namely: results of pairwise comparisons, decision rules and optionally preference graphs.

First of all, results of pairwise comparisons must be handled. That kind of data could be fitted, for example, into the following tuple: (CmpId, MA-ID1, MA-ID2, Precedence, DM, IDofCriteriaSet, PLSessionId). CmpId is the autogenerated primary key. MA-ID1 and 2 are referencing meta-alert objects. Precedence encodes four cases: if MA-ID1 precede MA-ID2 it takes value '1', if MA-ID2 precede MA-ID1 it takes value '2', if there are equal then '3', if the precedence cannot be justified then '0'. DM is an identifier of decision maker – user of the system. IDofCriteriaSet allows identifying the set of criteria used during the comparison. PLSessionId is a reference to data describing preference learning session. It is worth to note that such data can be represented either in the relational form or in the key-value form and stored as document in NoSQL database. Key-value representation may be desired from an implementation and development point of view. In addition to aforementioned data it may be also useful to store differences of criteria values, in other words, the PCT table. However it is easily doable only if it will be implemented in NoSQL database. Although it is not necessary in order to achieve the basic functionality of the system.

Decision rules are generated as result of running DRSA algorithm with the information table and list of results of pairwise comparisons as inputs. For the first version of the system jRank software will be used. Thus the system must be able to handle decision rules in the way compatible with jRank. Each rule can be represented as a string in the format used in jRank and in if-then notation e.g.:

- '1: {PAIR(Evaluations_on_CategoryPriority) D (Medium, High) } => (Comprehensive_preference_grade >= 0.0) |CERTAIN, AT_LEAST, 0 . 0 |'.
- if CategoryPriority(x) $\geq$Medium and CategoryPriority(y) $\leq$ High then xSy (where S means that x is weakly preferred over y)

Preference model will consist of:

- set of decision rules,
- set of criteria (see Section 3.2 for the specification of criterion),
- reference to pairwise comparisons used to generate the model,
- information about decision maker.

The information from preference model in conjunction with data extracted from meta-alerts is sufficient for generating temporal information tables needed for generation of rankings on the subsets of meta-alerts.

Optionally preference graph can be built for the set of pairwise comparisons used to generate model. The storage of preference graphs is discussed in the following section.

3.5 Representation of Preference Graphs

During the process of rankings generation, the system stores the preference graphs in the form of a list of arcs, namely, as the pairwise comparisons. The implementation of DRSA will also be able to generate as output the PCT table (it is also form of list of arcs). Thus, at first glance, it seems that storing a list of arcs is a natural choice for storing preference graphs. Although, it may be not very efficient for more sophisticated analysis and efficient visualization without appropriate support from DBMS engine. The efficiency of the graph processing tasks can be improved by using graph databases

or triple stores (also known as RDF databases). The examples of databases supporting graph operations are Neo4j, GraphDB (by Ontotext), MongoDB (since version 3.4).

3.6 Selection of Methods Generating Rankings

The dominance-based rough set approach (cf. Section 2.4.2) seems to be a reasonable choice for building rankings in the presence of only partial preference information. The principles of the methods are easy to understand for the user of the system. The only requirement for the criteria is their monotonicity, so the method is flexible and can be without hassle accommodated to changing requirements. The method has potential to deal with large datasets due to reasonable complexity.

For the generation of ranking, from the preference graph acquired through application of decision rules on the set of meta-alerts, the NFS and RNFS scores (cf. Section 2.4.3) will be available as the choice options.

The initial version of the system will generate rankings for the given sets in batch mode. However it is considered to provide functionality allowing for usage of decision rules to update once generated ranking in order to include new observations and fit them into existing ranking structure.

Ranking can be stored in form of lists (or lists of lists) where each element of the list references the original meta-alert object and vector of criteria values.

3.7 Visualization

The aim of this section is not to provide definitive answers on the design patterns of visualizations but to point out general possibilities and goals of rankings and preference graphs presentation. Moreover, within this section, the support of interactive learning tasks through appropriate visualization is mentioned.

3.7.1 Rankings

Alerts and incident reports in incident handling or ticketing system appear usually in a tabular form. Operators of the system are accustomed to such representation. Thus the PROTECTIVE system should support such way of ranking presentation. However, there could be multiple points of entry to such view placed in various dashboards customized to the needs of the individual user. For example, the dashboard with statistics can lead to rankings of particular categories of the alerts. It is worth to note that presentation layer could be coupled with data filtering capabilities of the system.

The tabular view can be enriched by incorporation of additional visual forms, e.g., representation of criteria values as bar charts instead of visually poor numbers, or by using a colour scale or colour intensities to visually distinguish variety of values.

Additionally, the ranking can be coupled with the presentation of preference graphs allowing more complex analysis of order of alerts, for example, to identify potential errors in preference model influencing final ranks.

3.7.2 Preference Graphs

The visualisation of preference graph is not a trivial task. However, multiple frameworks supporting the visually attractive presentation of graphs in 3D and 2D space exists on the market (e.g., D3.js, alchemy.js, three.js, sigma.js). It is important to choose appropriate layout for the graph to make it informative. For example, in case of presenting preference graph having additional information about ranks of the meta-alerts (represented as vertices), it is possible to use layered layout and present vertices with the same ranks on the same layer. Another way to represent ranks is to use circles with a variable diameter (the largest diameter – the largest rank).

The aim of visualisation is to allow conducting visual inspection of the preference graph used to derive preference model (in particular decision rules) in the first place. Secondly it can be used to allow analysis of preference graphs derived from PCT tables computed on the basis of rules from preference model. The data from PCT is the ground truth used for generation of final rankings. Based on the visual inspection the DM can identified possible problems in the preference model and for example correct it by adding relevant pairs of examples to the pairwise comparisons used for model generation.

3.7.3 Support for Interactive Learning

Visualisation for interactive learning should allow presenting in an attractive way examples of vectors of attributes and allow to point preference relations between them (cf. Section 3.3.2). It can be done either in a tabular form or a graph representation. In the latter case, the user should have possibilities to change (by hand) arrangements of vertices and draw arcs showing his or her preference. The user should be able also to navigate through the sets of vectors used for preference gathering. Finally, the knowledge gathered during the learning process should be visualize as preference graph. It is worth to note, that for graph handling with support of interactions the frameworks mentioned above (in Section 3.7.2) can be also used.

3.8 Requirements for PROTECTIVE System

During the investigation of prioritisation and ranking methods for PROTECTIVE several specific requirements have been identified. The requirements defined below are complementary to those described in D2.1.

Successful preference gathering requires the availability of data about users, interfaces for collecting preferences – preferably user-friendly graphical interface, a storage facility allowing flexible data processing and facilitating in particular storage of preference graphs and optionally decision rules– preferably NoSQL database.

The system should be capable of storing multiple preference models for given DM as well as linking single preference model to multiple DMs (for example to support the case when multiple operators work according to the same well defined procedures).

Successful preference discovery requires the availability of user activity logs including login and logout times, meta-alerts handling history along with definitions of the processing statuses and times of transitions between those statuses (especially creation of the meta-alert and start of the handling), and users involved in the processing.

User should be able to define criteria and related them to attributes of meta-alerts. The criteria definition mechanism should allow for mapping of attributes domains into criteria domains (e.g. mapping Categories to priorities from the following set {High, Medium, Low}, cf. Section 3.3.1).

The system should have capabilities of filtering meta-alerts according to definable patterns, e.g. using criteria values, attributes values or both.

4 Meta-Alerts Prioritisation Development

4.1 Prioritisation module

The following functional submodules can be identified as important for prioritisation and ranking process: filtering module; criteria handling module; preference gathering module; preference discovery module; ranking generation.

Implementation of internalities of filtering module is highly coupled with the choice of backend storage. The filtering possibilities depends from available query mechanisms and possible parametrization of the queries.

Criteria handling module is a set of methods allowing extraction of criteria vectors from meta-alert structures. The module is critical for the development of prioritisation and further ranking generation. The functionalities that will be implemented within the module include:

- mechanisms allowing defining of criteria by selection of attributes and mapping them into criteria space using maps or functions;
- mechanisms allowing to fetch meta-alert from database or stream and transform it into an item of information table (vector of criteria and attributes).

Preference gathering module implements methods for storing results of pairwise comparisons, importing and exporting pairwise comparisons from and to files, acquiring preference information through the graphical interface as pairwise comparisons, preference graphs and reference rankings.

Preference discovery module depends on user activity logs and availability of history of meta-alerts handling process. Its implementation is also partially dependent on capabilities of backend storage. The module will partially reuse functionalities from preference gathering module.

The first version of the ranking generation module will use DRSA approach and will be based on jRank software.

4.2 Graphical User Interface

Graphical user interface for the Prioritisation and Ranking module will be part of the general user interface of the PROTECTIVE system. Thus it will use the same frameworks and design patterns like the other modules. The graphical user interface will allow configuring all crucial mechanisms of the module, namely definitions of criteria, selection of criteria, filtering patterns, gathering preferences, configuring of the preference discovery systems.

The functionalities will be delivered gradually during consecutive development cycles. In early stages of development, the most functions will be configured using configuration files. On top of configuration files, the interactive dialogs will be designed and delivered. Please refer to D4.5 for more information on user interfaces.

5 Summary

Prioritisation and ranking taking into account context awareness and preference information provided by the user or derived from the history of users activities concerning incident handling coupled with particular meta-alerts is not a trivial task. As it was shown within the report, the task requires the application of solid theory and customizable operational framework to meet expectations of the end-users preferably in highly functional and visually attractive way. The important aspect of the theoretical part is the ability to deal with ambiguities and uncertainties in the data received from the sensors and introduced by the user during preference gathering. In the cybersecurity domain, it is often the case that not all the knowledge that drive decision maker's action is available in the system.

Thus uncertainties and ambiguities, from the perspective of the system, are immanent part of the processes that the PROTECTIVE aims to support. Choice of the technique based on the rough sets theory, dominance based principles, and intuitive preference gathering methodology seems to be reasonable. The methodology will be further adjusted to the needs of particular users during the pilot phases. It is worth to note that implementation of prioritisation module without support for ranking is a much simpler task and can be based on outputs of context awareness modules implemented within the PROTECTIVE project.

6 References

Aiolli, F. and Sperduti, A., (2005). Learning preferences for multiclass problems. In Advances in neural information processing systems (pp. 17-24).

Aiolli, F., Sebastiani, F. and Sperduti, A., (2007). Preference learning for category-ranking based interactive text categorization. In Neural Networks, 2007. IJCNN 2007. International Joint Conference on (pp. 2034-2039). IEEE.

Addor,M.L. & Smutko, S. (2011). Multi-Criteria Decision Analysis. Retrieved 08 22, 2017 from: https://www.ncsu.edu/nrli/decision-making/MCDA.php

Barfod, M.B. and Leleur, S., 2014. Multi-criteria decision analysis for use in transport decision making. the, 2, pp.5-75.

Belton, V., & Pictet, J. (1997). A framework for group decision using a MCDA model: sharing, aggregating or comparing individual information? Journal of decision systems, 6(3), 283-303.

Belton, V. & Stewart, T. (2002). Multiple criteria decision analysis: an integrated approach. Springer Science & Business Media.

Beroggi, G. (2013). Decision modeling in policy management: an introduction to the analytic concepts. Springer Science & Business Media.

Bouyssou, D., Vincke, P. (1997). Ranking Alternatives on the Basis of Preference Relations: A Progress Report with Special Emphasis on Outranking Relations. Journal of Multi-Criteria Decision Analysis, vol. 6, pp. 77-85

Cao, Z., Qin, T., Liu,T.-Y., Tsai, M.-F., Li., H. (2007). Learning to rank: from pairwise approach to listwise approach. In ICML 2007, pages 129-136, 2007. Extended version available also at: https://www.microsoft.com/en-us/research/publication/learning-to-rank-from-pairwise-approach-to-listwise-approach/

Communities UK (2009), Multi-criteria analysis: a manual, Department for Communities and Local Government, London, January 2009. Retrieved 08 22, 2017 from: http://eprints.lse.ac.uk/12761/1/Multi-criteria_Analysis.pdf

Couder, J. (2015) Review of Research Efforts and Approaches to Energy Efficiency Barriers. Deliverable D3.1 of Heron Project. Retrieved 08 22, 2017: http://heron-project.eu/images/Deliverables/HERON_D3.1_Review-of-Approaches.pdf

Dekel, O., Singer, Y. and Manning, C.D. (2004). Log-linear models for label ranking. In Advances in neural information processing systems (pp. 497-504).

Dembczyński, K., Kotłowski, W., Słowiński, R. and Szeląg, M., 2010. Learning of rule ensembles for multiple attribute ranking problems. In Preference Learning (pp. 217-247). Springer Berlin Heidelberg.

Demir, I. (2014). Analytical Hierarchical Process (AHP), Retrieved 08 28, 2017 from: http://content.lms.sabis.sakarya.edu.tr/Uploads/49858/27314/week-10-ahp-analytical_hierarchy_process.pdf

Doyle, J. (2004). Prospects for preferences. Computational Intelligence, 20(2), 111-136.

Edwards, W. (1977). How to use multiattribute utility measurement for social decisionmaking. IEEE Transactions on Systems, Man, and Cybernetics, 7(5), pp.326-340.

ENISA. (2010). Good Practice Guide for Incident Management. Retrieved from:https://www.enisa.europa.eu/publications/good-practice-guide-for-incident-management

ENISA. (2013). Alerts, Warnings, Announcements - best practice guide. Retrieved from: https://www.enisa.europa.eu/publications/awa

Figueira, J., Greco, S., & Ehrgott, M. (Eds.). (2005). Multiple Criteria Decision Analysis: State of the Art Surveys (Vol. 78). Springer Science & Business Media. DOI:10.1007/b100605.

Fishburn P. C. (1967). Additive Utilities with Incomplete Product Set: Application to Priorities and assignments, Operational Research Society of America (ORSA), Baltimore.

Fürnkranz, J. & Hüllermeier, E. (2010). Preference learning: An introduction. In Preference learning (pp. 1-17). Springer Berlin Heidelberg

Fürnkranz, J., & Hüllermeier, E. (2010a). Preference learning and ranking by pairwise comparison. In J. Fürnkranz & E. Hüllermeier (Eds.), Preference Learning (pp. 63-80). Springer.

Fürnkranz, J., & Hüllermeier, E. (Eds.). (2010b). Preference learning. Springer

Fürnkranz, J. & Hüllermeier, E. (2011). Preference Learning: A Tutorial Introduction. Retrieved 08 22, 2017 from: http://www.preference-learning.org/PL-Tutorial-DS-11.pdf

Fürnkranz, J., Hüllermeier, E., Rudin, C., Sanner, S., Słowiński, R. (2014). Preference Learning. Report from Dagstuhl Seminar 14101. Retrieved 08 22,2017: http://drops.dagstuhl.de/opus/volltexte/2014/4550/pdf/dagrep_v004_i003_p001_s14101.pdf

Geldermann, J., & Treitz, M. (2008). Quantifying eco-efficiency with multi-criteria analysis. Schwerpunkt Unternehmensführung, Wirtschaftswiss. Fak..

Goodwin (2003), Decision Involving Multiple Objective: SMART Simple Multi-attribute Rating Technique, Powerpoint presentation.

Greco S., Matarazzo, B., Slowinski, R. (1995). Rough set approach to multi-attribute choice and ranking problems. ICS Research Report 38/95, Warsaw University of Technology.

Greco, S., Matarazzo, B., Slowinski, R. (1999) Rough approximation of a preference relation by dominance relations. European Journal of Operational Research, 117(1),63-83. DOI: 10.1016/S0377-2217(98)00127-1.

Hobbs, B. F. and P. Meier (2000). Energy decisions and the environment – a guide to the use of multicriteria methods. Kluwer Academic Publishers.

Hüllermeier, E. (2014). Preference Learning: Machine Learning Meets MCDA. DA2PL 2014. Retrieved 08 21, 2017 from: http://www.lgi.ecp.fr/~mousseau/DA2PL-2014/uploads/Main/Eyke.pdf

Hwang C. L. and Yoon K. (1981), Multiple Attributes Decision Making Methods and Applications, Springer, Berlin, Heidelberg, 1981.

InfoHarvest (2017) Answers to Frequently Asked Questions about decision analysis. Retrieved 08 22, 2017 from: http://www.infoharvest.com/ihroot/infoharv/infoharvestfaq.asp

Ishizaka, A., & Nemery, P. (2013). Multi-criteria decision analysis: methods and software. John Wiley & Sons.

IIBA. (2009). A Guide to the Business Analysis Body of Knowledge. International Institute of Business Analysis.

Kadziński, M., Słowiński, R., & Szeląg, M. (2016). Dominance-based rough set approach to multiple criteria ranking with sorting-specific preference information. In Challenges in Computational Statistics and Data Mining (pp. 155-171). Springer International Publishing.

Komorowski, J., Polkowski, L., Skowron, A. (1999). Rough Sets: A Tutorial Retrieved 08 22, 2017 from: http://eecs.ceas.uc.edu/~mazlack/dbm.w2011/Komorowski.RoughSets.tutor.pdf

Kruegel, Ch., Valeur, F., Vigna, G. (2005) Intrusion Detection and Correlation: Challenges and Solutions, Advances in Information Security, Vol. 14, 2005, Springer 2005, pp. 31

Lan, Y., Zhu Y., Guo, J., Niu, S. Cheng, X. (2014). Position-Aware ListMLE: A Sequential Learning Process for Ranking. In UAI 2014, 164. Url: http://auai.org/uai2014/proceedings/individuals/164.pdf

Li, H. (2011). A Short Introduction to Learning to Rank. IEICE Transactions on Information and Systems, E94.D(10), 1854-1862. DOI: 10.1587/transinf.E94.D.1854

Liu, T. (2009). Learning to rank for information retrieval, volume 3 of Foundations and Trends in Information Retrieval.

MindTools. (2017). Eisenhower's Urgent/Important Principle. Retrieved 08 22, 2017 from: https://www.mindtools.com/pages/article/newHTE_91.htm.

Mitchell, T. (1997). Machine Learning. McGraw-Hill, New York

Munda, G. (2003). Multicriteria assessment. International Society for Ecological Economics.

Nurse, J. R., & Sinclair, J. E. (2012, June). Towards a model to support the reconciliation of security actions across enterprises. In Socio-Technical Aspects in Security and Trust (STAST), 2012 Workshop on (pp. 11-18). IEEE.

Olson, D.L. (1996). Decision aids for selection problems. Springer Science & Business Media.

Palesi M. (2006), A Brief Introduction to Multiobjective Optimization Techniques, Universita di Catania.

Pawlak, Z. (1982). Rough sets. International Journal of Parallel Programming, 11(5), 341-356.

Pawlak, Z. (1991). Rough Sets: Theoretical Aspects of Reasoning about Data (Vol. 9). Springer Science & Business Media.

Pedersen, J. (2008). The Machine Learned Ranking Story. Retrieved 08 21, 2017 from: http://jopedersen.com/Presentations/The_MLR_Story.pdf

Roszkowska, E. (2011). Multi-criteria decision making models by applying the TOPSIS method to crisp and interval data. Multiple Criteria Decision Making/University of Economics in Katowice, 6, 200-230. Retrieved 08 22, 2017 from: http://www.mcdm.ue.katowice.pl/files/papers/mcdm11(6)_11.pdf

Roy, B. (1981). The optimisation problem formulation: Criticism and overstepping. Journal of the Operational Research Society, 32(6), pp. 427-436.

Saaty, T.L. (1977). Scenarios and priorities in transport planning: Application to the Sudan. Transportation Research, 11(5), pp.343-350.

Seel,N. M. (ed.), (2012). Encyclopedia of the Sciences of Learning, Springer. DOI: 10.1007/978-1-4419-1428-6

Slowinski,R.,Greco,S., Matarazzo B. (2012). Rough Set and Rule-Base Multicriteria Decision Aiding. *Pesquisa Operacional*, *32*(2), 213-270. DOI: 10.1590/S0101-74382012000200001

Taylor, M., Guiver, J., Robertson, S., Minka, T. (2008). SoftRank: Optimising Non-Smooth Rank Metrics, In WSDM 2008, https://www.microsoft.com/en-us/research/publication/softrank-optimising-non-smooth-rank-metrics/

White, B. (2006). Math 51 lecture notes: How Google ranks web pages. Retrieved 08 21, 2017 from: http://math.stanford.edu/~brumfiel/math_51-06/PageRank.pdf

Xiaoqian S. (2012), Multiple Criteria Decision Analysis Techniques in Aircraft Design and Evaluation Processes, Ph. D. thesis, Technischen Universitat Hamburg-Harburg.

Xu, Q., Zhang, Y. B., Zhang, J., & Lv, X. G. (2015). Improved TOPSIS model and its application in the evaluation of NCAA basketball coaches. *Modern Applied Science*, *9*(2), 259. Retrieved 08 22, 2017 from: http://ccsenet.org/journal/index.php/mas/article/view/41201

Xu, J. & Li.,H. (2007). Adarank: a boosting algorithm for information retrieval. In SIGIR 2007, 391-398.

Yang, X. B., & Yang, J. Y. (2012). Incomplete information system and rough set theory (pp. 195-222). Berlin, Germany: Springer.

Zadeh, L. A. (1965). Fuzzy sets. Information and control, 8(3), 338-353.