Interpretable AI-Enhanced Multi-Criteria Decision-Making for Time-Critical Emergency Applications
With a Practical Application to Dynamic Alternate Airport Selection in Aviation

DISSERTATION

zur Erlangung des akademischen Grades

Doktoringenieur (Dr.-Ing.)

angenommen durch die Fakultät für Informatik
der Otto-von-Guericke-Universität Magdeburg

von M. Sc. Boris Djartov

geb. am 12.01.1995 in Skopje

Gutachterinnen/Gutachter

Prof. Dr.-Ing. habil. Sanaz Mostaghim
Assist. Prof. Dr. Tome Eftimov
Prof. Dr. Manuel López-Ibáñez

Magdeburg, den 20.04.2026

Boris Djartov

Interpretable AI-Enhanced Multi-Criteria Decision-Making for Time-Critical Emergency Applications

With a Practical Application to Dynamic Alternate Airport Selection in Aviation

December 2025

Ehrenerklärung

Ich versichere hiermit, dass ich die vorliegende Arbeit ohne unzulässige Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe; verwendete fremde und eigene Quellen sind als solche kenntlich gemacht. Insbesondere habe ich nicht die Hilfe eines kommerziellen Promotionsberaters in Anspruch genommen. Dritte haben von mir weder unmittelbar noch mittelbar geldwerte Leistungen für Arbeiten erhalten, die im Zusammenhang mit dem Inhalt der vorgelegten Dissertation stehen.

Ich habe insbesondere nicht wissentlich

- Ergebnisse erfunden oder widersprüchliche Ergebnisse verschwiegen,
- statistische Verfahren absichtlich missbraucht, um Daten in ungerechtfertigter Weise zu interpretieren,
- fremde Ergebnisse oder Veröffentlichungen plagiiert,
- remde Forschungsergebnisse verzerrt wiedergegeben.

Mir ist bekannt, dass Verstöße gegen das Urheberrecht Unterlassungs- und Schadensersatzansprüche des Urhebers sowie eine strafrechtliche Ahndung durch die Strafverfolgungsbehörden begründen kann. Die Arbeit wurde bisher weder im Inland noch im Ausland in gleicher oder ähnlicher Form als Dissertation eingereicht und ist als Ganzes auch noch nicht veröffentlicht..

Magdeburg, den 09.12.2025

Boris Djartov

Abstract

Time-critical decision making in emergency situations poses significant challenges for both humans and traditional multi-criteria decision-making methods. Under such demanding conditions, supporting human decision makers with tools that structure information and reduce cognitive burden is both beneficial and prudent. This thesis addresses this challenge by introducing the ExACT-MCDM methodology, a framework that integrates artificial intelligence with multi-criteria decision-making concepts to support high-stakes decisions in environments where human operators remain responsible and actively involved in the emergency scenario.

The proposed methodology reformulates emergency, time-critical multi-criteria decision-making problems as supervised ranking tasks, enabling the use of machine learning models that preserve the structure of options and attributes while providing rapid, context-aware recommendations. To address the scarcity of real emergency data, a synthetic scenario generation and labeling process was developed, incorporating expert knowledge through structured scoring functions with hard and soft constraints. Robustness was achieved by simulating realistic variability in contextual factors and maintaining their interdependencies, resulting in datasets that reflect the uncertainty inherent in real emergency operations.

The methodology was applied to the dynamic alternate-airport selection problem, a representative aviation emergency scenario in which pilots must select a diversion airport under dynamic and unfamiliar conditions. Experiments were conducted using both classification and ranking formulations across a range of models. The results indicated that ranking was better suited to the structure of the problem and more consistent with real decision-making processes, where the goal is to order feasible alternatives rather than output a single preferred choice. Interpretability and explainability were incorporated throughout the process using inherently interpretable models, structured feature contributions, and model-agnostic tools such as SHAP, with the aim of helping stakeholders maintain oversight of why recommendations are made.

The results demonstrate the potential that artificial intelligence has to enhance multi-criteria decision making and provide fast, context-sensitive, and transparent decision support suitable for emergency situations while preserving the responsibility and authority of human decision makers. Aside from the ExACT-MCDM methodology, this thesis contributes a scalable benchmark that advances understanding of the dynamic alternate-airport selection problem and provides guidance for integrating such models into operational systems. Together, these contributions demonstrate how artificial intelligence can meaningfully augment decision making in complex, unpredictable, and high-stakes environments, including cockpit decision-support tools such as the Intelligent Pilot Advisory System under development at the German Aerospace Center.

Zusammenfassung

Die zeitkritische Entscheidungsfindung in Notfallsituationen stellt sowohl für Menschen als auch für traditionelle Methoden der multikriteriellen Entscheidungsfindung eine erhebliche Herausforderung dar. Unter solchen anspruchsvollen Bedingungen ist es sinnvoll und vorteilhaft, menschliche Entscheider durch Werkzeuge zu unterstützen, die Informationen strukturieren und die kognitive Belastung reduzieren. Diese Dissertation begegnet dieser Herausforderung durch die Einführung der ExACT-MCDM-Methodik, eines Rahmens, der künstliche Intelligenz mit Konzepten der multikriteriellen Entscheidungsfindung verbindet, um Entscheidungen mit hohen Risiken in Situationen zu unterstützen, in denen menschliche Akteure weiterhin verantwortlich sind und aktiv im Notfallszenario eingebunden bleiben.

Die vorgeschlagene Methodik reformuliert atypische, zeitkritische multikriterielle Entscheidungsprobleme als überwachtes Ranking-Problem und ermöglicht so den Einsatz von Modellen des maschinellen Lernens, die die Struktur von Optionen und Attributen bewahren und gleichzeitig schnelle, kontextabhängige Empfehlungen liefern. Um dem Mangel an realen Notfalldaten zu begegnen, wurde ein Verfahren zur synthetischen Szenariogenerierung und -kennzeichnung entwickelt, das Expertenwissen über strukturierte Bewertungsfunktionen mit harten und weichen Nebenbedingungen integriert. Die Robustheit der Daten wurde durch die Simulation realistischer Variabilität kontextueller Faktoren und die Berücksichtigung ihrer gegenseitigen Abhängigkeiten erreicht, sodass Datensätze entstehen, die die Unsicherheiten realer Notfalloperationen widerspiegeln.

Die Methodik wurde auf das Problem der dynamischen Auswahl eines Ausweichflughafens angewendet – ein repräsentatives Notfallszenario in der Luftfahrt, bei dem Piloten unter dynamischen und potenziell ungewohnten Bedingungen einen geeigneten Ausweichflugplatz bestimmen müssen. Experimente, in denen Klassifikations- und Ranking-Formulierungen über verschiedene Modelle hinweg untersucht wurden, zeigten, dass der Ranking-Ansatz besser zur Struktur des Problems passt und näher an realen Entscheidungsprozessen liegt, bei denen mehrere machbare Alternativen geordnet werden müssen, anstatt ausschließlich eine einzige Option vorherzusagen. Interpretierbarkeit und Erklärbarkeit wurden durch den Einsatz inhärent transparenter Modelle, strukturierter Beitragsanalysen und modellunabhängiger Werkzeuge wie SHAP über den gesamten Prozess hinweg integriert, um Beteiligten ein Verständnis darüber zu ermöglichen, weshalb Empfehlungen ausgesprochen werden.

Die Ergebnisse verdeutlichen das Potenzial künstlicher Intelligenz, die multikriterielle Entscheidungsfindung zu unterstützen und schnelle, kontextsensitive und transparente Entscheidungsunterstützung für Notfallsituationen bereitzustellen, während die Verantwortung und Autorität menschlicher Entscheidungsträger gewahrt bleiben. Neben der ExACT-MCDM-Methodik liefert diese

Dissertation einen skalierbaren Benchmark, der das Verständnis des Problems der dynamischen Ausweichflugplatzwahl vertieft, und bietet praktische Hinweise zur Integration solcher Modelle in operationale Systeme. Zusammen zeigen diese Beiträge, wie künstliche Intelligenz die Entscheidungsfindung in komplexen, unvorhersehbaren und sicherheitskritischen Umgebungen sinnvoll erweitern kann – einschließlich cockpitnaher Entscheidungsunterstützungssysteme wie dem Intelligent Pilot Advisory System, das derzeit am Deutschen Zentrum für Luft- und Raumfahrt entwickelt wird.

Contents

1 Introduction

1.1 Motivation

1.1.1 Challenges of Time-Critical MCDM and the Role of AI

Decision-making is a fundamental aspect of human existence, from routine daily choices to complex strategic determinations that shape our personal and professional lives. Every day, we navigate countless decisions, weighing options and considering consequences based on our experience, knowledge, and preferences. In our everyday lives, many decisions are made almost instinctively, drawing on what psychologists call "System 1" thinking, which is a fast, automatic, and emotionally driven cognitive process [135]. This intuitive decision-making process relies on mental shortcuts known as heuristics and on pattern recognition developed through years of experience [92]. While these cognitive shortcuts serve us well in familiar situations, they can also lead to systematic biases and errors in judgment [105]. As the complexity of a decision increases, however, the need for a more structured approach also grows. This engages what is known as our "System 2" thinking, which is slower, more deliberate, and more analytical [135]. Structured thinking allows people to break down problems into clear elements, evaluate trade-offs, and weigh multiple factors more consistently. This is precisely the kind of reasoning that Multi-Criteria Decision Making (MCDM) methods aim to support. By providing a formal framework for comparing alternatives across different dimensions, MCDM helps decision-makers reach choices that balance competing objectives in a systematic and transparent way.

The complexity of MCDM increases significantly when decisions must be made under uncertainty, where the decision-maker must consider not only known factors but also potential future scenarios and their implications. Traditional MCDM approaches typically rely on methods such as pairwise comparisons, trade-off matrices, and weight assignments to capture decision-maker preferences and help to make a decision. While these techniques have their uses in certain decision-making scenarios, they become increasingly impractical in time-constrained situations. The cognitive demands of systematically comparing alternatives, establishing consistent weights, and maintaining coherent preference structures are particularly challenging under time pressure. Moreover, these conventional methods often assume that decision-makers have sufficient time to gather information, analyze alternatives, and carefully weigh their options. However, in many real-world situations, particularly during emergencies or crises, decision-makers face severe time constraints while operating under heightened stress levels. These conditions can significantly impact cognitive processing, potentially leading to suboptimal decisions precisely when the stakes are highest.

The challenge becomes even more pronounced when dealing with emergency situations, which are scenarios that fall outside the scope of the decision-maker's routine operational experience. In such cases, the decision-maker must not only process multiple criteria under time pressure but also adapt their decision-making approach to less frequently encountered or contextually unfamiliar circumstances while managing the psychological stress of the situation. This combination of time pressure, multiple criteria, and atypical circumstances can create a particularly challenging decision-making environment.

The limitations of traditional MCDM methods raise the question of how such decisions could be supported more effectively. To address these limitations, this thesis explores methods from the field of Artificial Intelligence (AI), since they can process complex inputs quickly, recognize patterns across many varying circumstances, and produce recommendations even when the available information is incomplete or noisy. In this way, an AI agent could evaluate a complex set of circumstances and options and provide the human decision-maker with a suitable recommendation in a timely manner.

1.1.2 Dynamic Alternate-Airport Selection: A Practical Use Case

A particularly relevant example of an emergency MCDM problem under a time constraint is mid-flight dynamic alternate-airport selection. Namely, during an event, such as an engine fire or medical emergency, commercial pilots may be faced with diverting from their original airport to a new alternate airport mid-flight. In such a predicament, decision-making in the cockpit of an airliner remains one of the most challenging and workload-intensive tasks for the flight crew. Pilots are required to gather and assess a wide range of information to support their decision-making. This may include details about the nature of a technical malfunction and its operational consequences, current weather conditions, nearby air traffic, and the infrastructure available at potential alternate airports.

Traditional alternate-airport selection, typically defined during pre-flight planning, relies on fixed alternates chosen according to static criteria. However, this approach does not reflect the dynamic nature of in-flight operations, where conditions may change rapidly [236, 225]. Dynamic alternate-airport selection acknowledges the fact that, in such cases, a diversion airport may change during the flight due to evolving weather patterns, airport status changes, airspace constraints, or updates to the aircraft's fuel state and position [298, 100]. Flight crews must therefore continuously update their mental models of potential alternates, integrating real-time information about multiple options while simultaneously managing their primary flight responsibilities. It is precisely the presence of these characteristics that makes a dynamic alternate-airport selection a suitable use case for an MCDM problem, in which decision-makers must balance competing objectives such as safety, fuel considerations, operational constraints, and passenger impact under significant time pressure.

Although diversions do not happen so readily, they account for approximately 0.3% of all flights in European airspace according to EUROCONTROL, which still is 28,000 diversion annually [89].

In the United States, according to the U.S. Bureau of Transportation Statistics, only 0.32% of flights were diverted in June 2024 [46]. The number of diversions, as well as the their percentage from the total flights, based on the data from the U.S. Bureau of Transportation Statistics are shown in figure 1.1.

Based on the available data, EUROCONTROL estimates that diversion cost varies significantly depending on the type of flight, with regional flights costing between €1,000 and €7,000, continental flights between €1,400 and €10,500, and intercontinental flights between €7,000 and €77,200 [89]. Beyond the direct costs, diversions can trigger cascading delays throughout the airline's network, affecting subsequent flights and potentially impacting thousands of passengers.

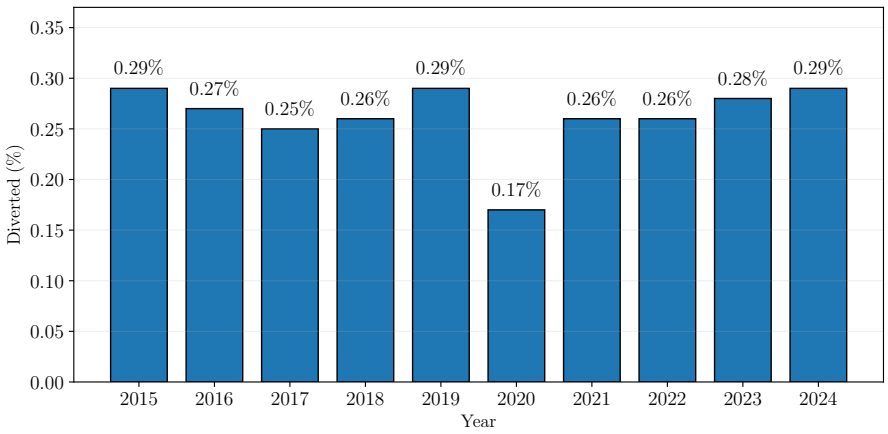

Figure 1.1: Percentage of Diverted Flights (2015-2024), U.S. Bureau of Transportation Statistics [46]

The dynamic and time-critical nature of these decisions makes them a relevant environment for exploring AI-enhanced MCDM. This work therefore uses dynamic alternate-airport selection as the primary application context for investigating such an approach.

These considerations align closely with ongoing work at the German Aerospace Center (Deutsches Zentrum für Luft- und Raumfahrt) (DLR), where current efforts are being undertaken to develop an Intelligent Pilot Advisory System (IPAS) as a research platform in order to investigate the potential of artificial intelligence in future commercial aircraft cockpits and examine the integration of human and AI systems in complex operational settings. The system was first introduced in [279] and has since been in active development [280, 281]. A specific functionality currently being developed for the IPAS system is exactly decision support capabilities for alternate-airport selection. A proposed system model for the IPAS system can be seen in Figure 1.2.

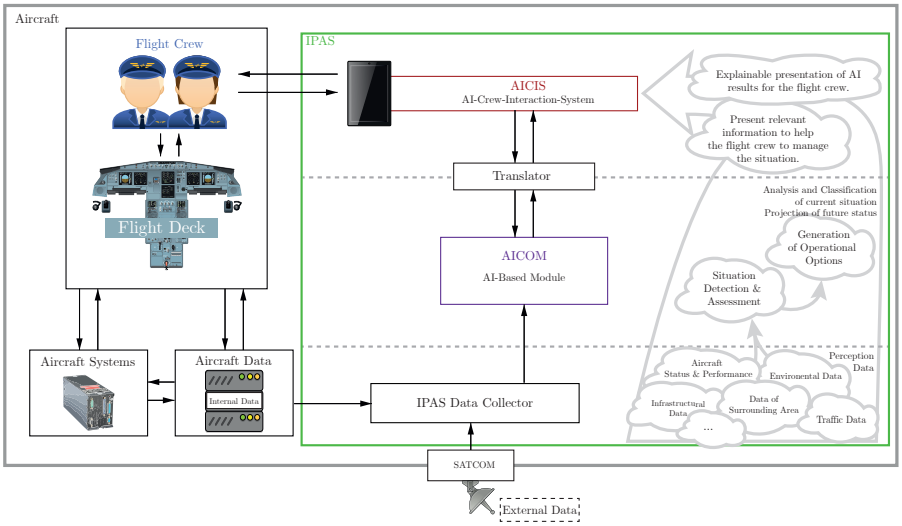

Figure 1.2: IPAS System Model, taken from authors publication [279]

The figure illustrates the main components of the system. It includes the AI Core Module (AICOM), which contains the models that support decision-making, and the AI-Crew Interaction System (AICIS), which presents these recommendations to the decision-makers (pilots). It also includes the data-collector module and the translation module, which enable communication between the AICOM and the AICIS. Within the context of this thesis, the IPAS in general and the AICOM in particular provide a practical environment for examining how AI-enhanced MCDM concepts can be implemented and evaluated as part of an intelligent Decision-Support System (DSS).

1.2 Research Goals and Questions

1.2.1 Research Goals

The main goal of this thesis is to develop a methodological approach on how to incorporate ideas from the field of AI to extend MCDM methods and enable the creation of concepts for intelligent DSS capable of providing support to decision-makers in time-critical emergency situations. To this end, this thesis addresses the following research goals:

G1: Develop a methodology for addressing atypical, time-critical MCDM problems using artificial intelligence to complement and extend traditional MCDM approaches.

> While decision-support systems for MCDM problems are well established in the literature, this research focuses on creating an approach that can help human decision-makers under uncertainty, especially when time pressure, unfamiliar situations, or stress make conventional methods impractical or insufficient.

G2: Demonstrate and evaluate this approach through the dynamic alternate-airport selection problem, contributing practical insights for further development of the IPAS system.

> Dynamic alternate-airport selection serves as a realistic and demanding test case that reflects the challenges of high-stakes aviation decisions involving multiple, often conflicting, criteria under rapidly changing conditions. By applying the developed approach to this use case, the research both tests its feasibility in practice and provides valuable knowledge for enhancing the development of the IPAS.

1.2.2 Research Questions

Building on these two goals, this thesis formulates the following research questions, each addressing a specific aspect of the methodological and practical challenges:

RQ1: In what ways can artificial intelligence techniques be integrated with MCDM concepts to improve decision support in emergency, time-critical situations where the main decision-makers and responsibility holders are embedded in the emergency scenario itself?

> The use of AI for supporting MCDM problems is already present in the literature. Various approaches exist in which AI models learn the weights or preferences for decision factors not through elaborate pairwise comparisons or direct expert elicitation, but by leveraging historical data relevant to the problem. Once these weights are estimated, standard MCDM methodologies can then be applied. In other instances, AI has been combined with MCDM methods to learn how to score each option by treating the problem as a regression task, allowing the model to approximate scores and rankings more efficiently. However, these examples typically address routine operations, such as supplier selection, where there is no severe time pressure or life-threatening consequences. In such contexts, AI-based approaches help automate processes and enhance decision-making, benefiting from relatively stable conditions and sufficient time to analyze information.
>
> Taking the use of AI further, this thesis explores how AI can be employed within MCDM problems where decisions must be made under significant uncertainty, high stakes, and severe time pressure. Crucially, it focuses on scenarios in which the ultimate decision-makers are not only managing the emergency but are embedded in it and will directly bear the consequences of their choices.

RQ2: Due to the diverse nature of emergencies, how can the AI model be trained in such a way that it may be able to provide decision-support adapted to the situation at hand?

> Decision-making is rarely performed in a vacuum. Decisions are heavily influenced by the context and conditions in which they are made. In fact, one could argue that the use of weight vectors, pairwise comparisons, and preference point setting is one of the most straightforward ways to not only express preferences but also embed information about the circumstances surrounding the decision. Therefore, it is important that the model can take into account the full context in which the decision is being made and, through this lens, provide appropriate suggestions to the decision-makers.

RQ3: What data requirements, limitations, and design considerations need to be addressed to develop and train an AI model that can reliably support time-critical, high-stakes MCDM scenarios?

> Given that emergencies are, by nature, exceptional events and do not fall within the scope of routine day-to-day operations, the availability of a suitable dataset for training an AI model is unlikely. Even if some data do exist, the question remains whether its distribution is adequate for training since emergencies can take many different and unpredictable forms. An aptly representative distribution in the training dataset is essential to ensure that the model can handle a wide range of scenarios rather than becoming specialized in just one type of emergency and failing to perform in others. For this reason, it makes sense to explore how data augmentation and synthetic data generation techniques can be used to produce sufficiently diverse and robust input for model training.

RQ4: How can interpretability and explainability be integrated so that stakeholders can have clear oversight regarding the decision-making process?

> In time-critical and high-stakes decision-making scenarios, such as dynamic alternate-airport selection, it is essential that stakeholders maintain clear oversight of how decisions are reached. Stakeholders in this context include not only the pilots making the decisions themselves but also airlines, investors, developers of the DSS, and governing bodies responsible for safety and certification. For all of these actors, integrating interpretability and explainability into the decision-support framework is a prudent and necessary step.
>
> By adopting a methodological approach in which the outputs of AI models are transparent, including how different factors are weighted and why specific options are prioritized, decision-makers can better understand, trust, and verify the system's recommendations. Techniques such as inherently interpretable models, clearly structured feature-importance measures, and visual explanations help translate complex model behavior into information that is accessible and meaningful for human operators. This ensures that stakeholders remain in control of the final decisions, are able to question or override system suggestions when necessary, and ultimately feel confident in the system's ability to support them under uncertain and dynamic conditions.

1.3 Thesis Structure

This thesis is organized into three distinct parts and nine chapters in total, following a logical progression from theoretical foundations to a showcase of a practical implementation.

Chapter two establishes the theoretical background, covering fundamental concepts in MCDM, as well as a showcase and classification of the various types and methods available. Afterwards, robust decision-making is explored, along

with its variants, which incorporate multiple scenarios and multi-objective concepts. What follows is a short primer on machine learning, covering concepts from regression, classification, and multi-class classification. Additionally, reinforcement learning is also covered, along with a brief explanation of the Markov Decision Process and a greater focus on bandit algorithms, with the notable inclusion of contextual multi-armed bandits. Following this, the Learning Classifier Systems paradigm is introduced, along with the multiplexer problem that is commonly associated with them. The chapter concludes with another primer on explainable and interpretable AI.

Chapter three reviews related work, focusing on the role of MCDM in emergency management and identifying key research gaps in existing approaches. It also surveys the use of AI in emergency contexts, discusses decision-making mental models, and presents aviation decision-support systems as a relevant application environment. The chapter ends with an overview of existing combined AI–MCDM approaches. Together with Chapter two, it helps contextualize the challenges underlying RQ1 and RQ2.

Chapter four outlines the theoretical contributions of this thesis and situates them within the author's publication record developed during the research period. Chapter five then introduces the ExACT-MCDM methodology, a structured, context-aware approach developed iteratively through successive refinements to address the previously identified challenges. These two chapters directly address RQ1 and RQ2, explaining how context can be represented and how the methodology emerged.

Chapter six addresses RQ3 by presenting the data-generation pipeline used in the experiments. It describes the design of synthetic scenarios, perturbation techniques, and the integration of contextual information required to obtain a diverse and representative dataset for training and evaluation.

Chapter seven outlines the experimental design in detail. It describes the implementation of the ExACT-MCDM methodology, the simulation setup, the dataset characteristics, and the models selected for evaluation. It also discusses aspects of interpretability and explainability within the models and the supporting Explainable Artifical Intelligence techniques, thereby partially addressing RQ4.

Chapter eight presents the experimental results and provides a comparative analysis of the models. Performance, robustness, feature-importance patterns, and interpretability characteristics are examined, offering empirical insights relevant to RQ1 and additional evidence for answering RQ4.

The thesis concludes with Chapter nine, summarizing key findings and outlining directions for future research.

Part I

Fundamentals and Related Work

2 Scientific Fundamentals

This chapter provides a systematic overview of the fundamental concepts upon which this thesis builds. It begins with a general introduction to Multi-Criteria Decision-Making and its subfields, followed by a closer examination of key methods from the area of Multi-Attribute Decision-Making. Next, the core ideas of Robust Decision-Making are introduced. The chapter then outlines essential concepts in machine learning, covering its main branches, such as supervised learning, unsupervised learning, and reinforcement learning. This is complemented by a brief discussion of Learning to Rank and weak supervision. Building on this foundation, concise descriptions of Gradient Boosting for decision trees, Artificial Neural Networks, and Learning Classifier Systems are provided. The chapter then explores explainable and interpretable AI, which frames the importance of transparency in decision support systems.

2.1 Multi-criteria Decision-making

Multi-Criteria Decision Making (MCDM) emerged as a distinct field in the mid-20th century, building on the foundations laid by von Neumann and Morgenstern in their 1947 work Theory of Games and Economic Behavior [199]. Their groundbreaking contribution to game theory and economic behavior provided a mathematical framework for understanding strategic decision-making, which eventually influenced the development of MCDM and Multi-Attribute Decision Making (MADM) [258].

The field gained further traction in the 1960s with seminal works such as Charnes and Cooper's goal programming [55], which introduced systematic approaches to handling multiple, often conflicting, objectives in decision problems. The recognition that real-world decisions typically involve balancing various criteria, rather than optimizing a single objective, was a key insight that drove the growth of MCDM methods in both theory and practice.

The fundamental formulation of a MCDM problem can be expressed in its most basic form as follows [128]:

Given:

$$A = \{A_1, A_2, ..., A_m\} \quad \text{(set of alternatives)}$$

$$C = \{C_1, C_2, ..., C_n\} \quad \text{(set of criteria)}$$

The decision matrix X represents the performance of each alternative against

each criterion:

$$X = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{m1} & x_{m2} & \cdots & x_{mn} \end{bmatrix}$$

The objective is to find:

$$A^* = \text{opt}_i f(x_{i1}, x_{i2}, ..., x_{in})$$

Where:

- x_{ij} represents the performance value of alternative i under criterion j.
- A^* is the optimal alternative.
- f is the decision function that maps performance values to a final decision.

This basic formulation serves as the foundation for more complex MCDM methods [32, 258].

MCDM naturally evolved into two main subfields: Multi-Objective Decision Making (MODM) and MADM [128]. MODM deals with continuous decision spaces and focuses on problems where multiple objective functions need to be optimized simultaneously, typically in design or planning scenarios where decision variables can take any value within defined constraints. In contrast, MADM addresses problems with discrete decision spaces, where decision-makers must choose from a finite set of predetermined alternatives, each characterized by multiple attributes [32]. This fundamental difference in problem structure—continuous versus discrete decision spaces-leads to distinct mathematical approaches and solution methodologies, with MODM employing techniques from mathematical programming and optimization theory, while MADM utilizes methods based on preference modeling and alternative comparison [140].

In [32], the authors challenge the conventional academic treatment of Multi-Criteria Decision Analysis (MCDA) that often begins with well-defined alternatives and criteria, arguing instead that real-world decision problems emerge as complex, unstructured challenges. Based on [32], the MCDA process unfolds in three distinct but interconnected phases:

1. Problem identification and structuring, focuses on fully understanding the complexity of the situation. During this phase, stakeholders and analysts work together to capture all relevant aspects of the problem, creating a foundation for moving forward. This initial exploration is crucial for ensuring that no critical elements are overlooked.

2. Model building. Here, the complexity captured in the first phase is distilled into its essential elements, creating a model that can support detailed evaluation of potential solutions. While the resulting model may appear simple, it should emerge from a deep understanding of the underlying complexity. This simplification follows the principle of "Through complexity to simplicity"—not denying the complexity but rather capturing its essence in a workable form.

3. Development of action plans, where the insights gained from the model are translated into concrete steps. These actions might include implementing specific actions, making recommendations, or establishing monitoring procedures. The process is inherently iterative, with the possibility of moving back and forth between phases as new insights emerge or circumstances change.

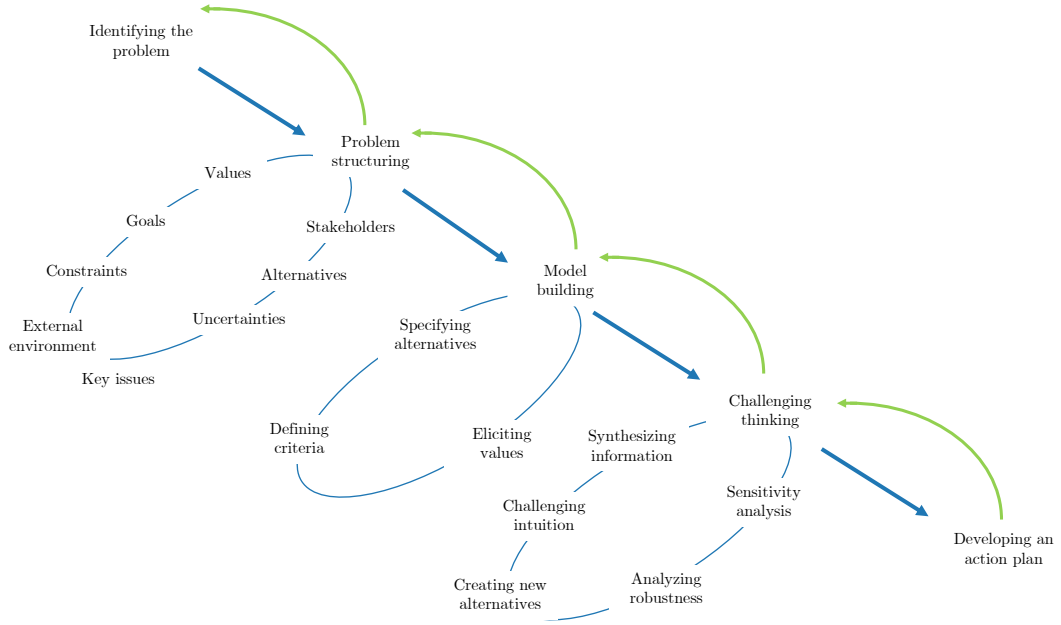

Figure 2.1: MCDM Process

Throughout all phases, multiple stakeholders play crucial roles—from decision-makers and clients to analysts and other affected parties. Their involvement and interactions shape the process, while various internal and external factors influence each phase. Different MCDA approaches primarily diverge in how they handle the model-building and use phase, particularly in their methods for information elicitation, specification, and synthesis. This understanding of MCDA as a comprehensive process rather than just an evaluation tool provides a more realistic and effective framework for addressing complex decision problems. It emphasizes the importance of thorough problem structuring before rushing to evaluate alternatives, ensuring that the final solution addresses the real needs of the situation [32].

The field has seen significant applications across diverse sectors. In urban planning, MCDM methods help evaluate transportation infrastructure projects, considering factors such as cost, environmental impact, and social benefits [178]. In environmental management, these techniques assist in renewable energy planning, waste management strategies, and climate change adaptation policies [269]. The healthcare sector employs MCDM for hospital location decisions, medical technology assessment, and treatment protocol selection [182].

2.1.1 Multi-Objective Decision Making and Multi-objective Optimization

Multi-Objective Optimization (MOO) is very near to multi-criteria decision making MCDM, as both the MODM subfield of MCDM and MOO aim to generate new solutions that approximate the best possible trade-offs among objectives. The primary goal in both MOO and MODM is to find or approximate the Pareto front, which represents solutions where no objective can be improved without degrading another. This approach allows decision-makers to explore a range of trade-offs and make informed choices based on their preferences and priorities.

A multi-objective optimization problem with n decision variables and m objectives, can be mathematically formulated as:

$$\text{Minimize} \quad \mathbf{F}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_m(\mathbf{x})) \, \text{Subject to} \quad \mathbf{x} \in \Omega \tag{2.1}$$

where: $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ is the vector of decision variables, $\mathbf{F}(\mathbf{x})$ is the vector

of objective functions, and Ω is the feasible set defined by constraints.

In comparing solutions, the concept of domination is crucial. A solution $\mathbf{x}^* \in \Omega$ is Pareto optimal if no other solution $\mathbf{x} \in \Omega$ exists such that:

$$f_i(\mathbf{x}) \leq f_i(\mathbf{x}^*) \quad \forall i \in \{1, 2, \ldots, m\} \tag{2.2}$$

and

$$f_j(\mathbf{x}) < f_j(\mathbf{x}^*) \quad \text{for at least one } j. \tag{2.3}$$

This implies that no objective can be improved without worsening at least one other objective. The Pareto front is the set of all Pareto optimal solutions, meaning it contains the solutions with the only the very best trade-offs that the decision-makers should consider [69].

Evolutionary Multi-Objective Optimization (EMO) makes use of Evolutionary Algorithms (EA) to handle multiple conflicting objectives. EMO algorithms exploit the population-based nature of EAs to explore and approximate the Pareto front. By maintaining a diverse set of solutions, EMO provides a set of possible solutions and enables decision-makers to consider their various trade-offs.[69, 61].

EAs, foundational to EMO, are inspired by natural evolution, with roots in the 1960s through the work of pioneers like John Holland [231], Lawrence Fogel [102], and Ingo Rechenberg [216]. These algorithms are particularly well-suited to complex optimization problems due to their ability to explore large search spaces effectively.

The typical steps of an evolutionary algorithm are as follows:

1. Initialization: The process begins with the random generation of an initial population of potential solutions [25]. Each solution, often referred to as an individual, is represented by a chromosome. This chromosome encodes the decision variables of the problem, which are the parameters that need to be optimized. Decision variables can be continuous, discrete, or a combination of both, depending on the nature of the problem. The size of the population is a crucial parameter, as it affects the algorithm's ability to explore the search space. A larger population can provide more diversity, but it also requires more computational resources. The initial population serves as the starting point for the evolutionary process.

2. Evaluation: Each individual in the population is assessed using a fitness function, which quantifies how well the solution satisfies the objectives of the optimization problem [106]. The fitness function is problem-specific and plays a critical role in guiding the search process. It translates the multi-objective problem into a scalar value that can be used to compare solutions. In multi-objective optimization, techniques such as Pareto ranking or weighted-sum approaches may be used to evaluate fitness. The evaluation step is essential for identifying the most promising solutions in the population.

3. Selection: This step involves choosing individuals from the current population to serve as parents for the next generation. Selection is based on the fitness of the individuals, with fitter individuals having a higher probability of being selected. Two common selection methods are as follows:

    - Roulette Wheel Selection: Also known as fitness proportionate selection, this method assigns a probability of selection to each individual based on its fitness relative to the total fitness of the population [231]. The selection process is analogous to spinning a roulette wheel, where each individual occupies a segment proportional to its fitness. This method ensures that individuals with higher fitness

have a greater chance of being selected, but it can be less effective when fitness values are very close.

- Tournament Selection: In this method, a subset of individuals is chosen randomly from the population, and the fittest among them is selected as a parent [107]. The tournament size, which is the number of individuals in each subset, is a key parameter. Tournament selection is robust and easy to implement, and it allows for fine-tuning of selection pressure by adjusting the tournament size. Larger tournaments increase selection pressure, favoring fitter individuals more strongly.

The goal of selection is to favor better solutions while maintaining diversity within the population.

4. Crossover (Recombination): In this step, pairs of selected individuals (parents) are combined to produce offspring [231]. Crossover involves exchanging segments of the parents' chromosomes to create new individuals that inherit characteristics from both parents. This process introduces new genetic material into the population and allows for the exploration of new areas of the search space. Various crossover techniques exist, such as single-point, multi-point, and uniform crossover, each with its own advantages and trade-offs. Crossover is a key mechanism for generating diversity and innovation in the population.

5. Mutation: Mutation introduces random changes to the chromosomes of some individuals in the population [25]. This step is crucial for maintaining genetic diversity and preventing premature convergence to suboptimal solutions. Mutation can involve altering one or more genes in a chromosome, and the mutation rate determines the likelihood of these changes occurring. A carefully chosen mutation rate helps balance exploration and exploitation in the search process. Mutation ensures that the algorithm can explore new solutions that may not be reachable through crossover alone.

6. Replacement: After generating the offspring through crossover and mutation, the new generation is formed by replacing some or all of the old population [106]. Replacement strategies vary, with some algorithms using elitism to ensure that the best solutions are carried over to the next generation. Elitism helps preserve the best solutions found so far, preventing them from being lost due to random genetic operations. The process of selection, crossover, mutation, and replacement is repeated until a termination criterion is met, such as a maximum number of generations, a satisfactory level of fitness, or convergence of the population.

EAs' flexibility and ability to handle multiple objectives simultaneously make them a powerful tool for multi-objective optimization [69, 61].

In the literature, several algorithms have gained prominence for solving real-world problems and serving as benchmarks for others. Notable examples include SPEA-II by Zitzler et al. [297], ϵ-MOEA by Deb et al. [71], and MOEA/D by Zhang and Li [290]. From these, one can argue that the NSGA-II algorithm has, in a certain sense, become the workhorse in MOO and a solid initial choice.

The core idea of NSGA-II is its non-dominated sorting mechanism, which divides the population into several fronts and selects individuals from the best fronts sequentially. Only solutions in the best fronts that fit the defined population size survive [70].

Fast Non-Dominated Sorting (FNDS) in NSGA-II calculates a domination count n_p and a set of solutions S_p for each solution $\vec{x}^{(p)}$. The domination count represents the number of solutions that dominate $\vec{x}^{(p)}$, while S_p contains solutions dominated by $\vec{x}^{(p)}$. Solutions with a domination count of zero are added to the first front F_1. For each member in F_1, the domination count

of solutions in S_p is reduced by one. Solutions that become non-dominated are added to the next front F_2. This process continues until all solutions are assigned to a front [70].

When a front cannot fully fit into the population, the crowding distance is used to select solutions. This measure indicates the density of solutions around a given solution in the objective space. Solutions with higher crowding distances are preferred, as they are in less crowded areas, making them easier for decision-makers to differentiate. This differentiation is crucial for achieving diversity in the final solution set [70].

The NSGA-II algorithm's ability to balance convergence and diversity results in a well-distributed approximation of the Pareto front, making it one of the more popular choices.

2.1.2 Multi-attribute decision-making methods

MADM methods are designed to evaluate and rank a finite set of alternatives based on multiple criteria [128]. These methods typically follow a structured approach consisting of the following:

1. Problem definition and structuring. This involves clearly defining the decision objective, identifying the alternatives to be evaluated, and determining the criteria that will be used for evaluation. In some cases, a hierarchical structure of criteria is established to better organize the decision problem [254].

2. Data collection and normalization. This involves gathering performance data for each alternative against each criterion. Since criteria may be measured on different scales, normalization is necessary to ensure comparability. Both qualitative and quantitative criteria are considered, with appropriate methods applied to handle each type [254].

3. Criteria weighting. This determines the relative importance of each criterion in the decision-making process. Various methods can be used to assign weights, such as direct weighting or pairwise comparisons. It is important to validate the consistency of the assigned weights to ensure they accurately reflect the decision-maker's preferences [128].

4. Alternative evaluation. The chosen madm method is applied to calculate final scores or rankings for the alternatives. Sensitivity analysis is often performed to assess the robustness of the results [254].

2.1.3 Classification and Analysis of Multi-Attribute Decision Making Methods

The classification of MADM methods has evolved over time, with different authors proposing various taxonomies based on different criteria. Here, a synthesized classification for madm applications is presented. The proposed classification borrows from both traditional classification approaches while incorporating modern developments in decision analysis.

Elementary Methods

Elementary methods form the foundation of madm, offering straightforward and transparent approaches to decision making [258]. These methods excel in situations where criteria can be clearly quantified and the decision structure is well-defined. Their simplicity makes them particularly valuable for initial analysis and as pedagogical tools for understanding basic madm concepts. One of the most commonly used methods from this category is the Weighted Sum Method (WSM).

The WSM represents one of the most straightforward and widely used MADM techniques [42]. Its popularity stems from its simplicity, transparency, and ease of implementation.

The WSM makes use of a weight vector to derive a score for each alternative [42]. Mathematically, it is calculated as:

$$A_i^* = \sum_{j=1}^{n} w_j x_{ij}$$

Where:

- A_i^* is the WSM score for alternative i.
- w_j is the weight of criterion j.
- x_{ij} is the normalized performance value of alternative i on criterion j.
- $\sum_{j=1}^{n} w_j = 1$ (weights sum to 1).

The WSM is still used to tackle MADM problems. Its simplicity makes it a very tempting choice for decision-makers in all contexts and especially interdisciplinary teams. However, despite its advantages, WSM faces several important limitations:

- An assumption of criteria independence. The method assumes that no interactions exist between different criteria, which is often in contrast to real-world situations where criteria frequently exhibit complex dependencies and relationships [119].
- Poor performance in one criterion must be compensated by good performance in others [32]. This is especially problematic where minimum performance thresholds must be maintained across all criteria or where certain criteria are critical to the decision outcome. As in the case of alternate-airport selection, certain criteria have set minimal safety limits that should not be violated.
- Linear preference assumptions, which may not accurately reflect real-world preference structures [119].

Value/Utility Methods

Value or utility-based methods provide a structured way to incorporate decision-maker preferences using systematic mathematical functions [140]. These approaches are especially powerful when both quantitative and qualitative criteria must be integrated and preferences must be modeled precisely.

The Analytic Hierarchy Process (AHP), introduced by Thomas L. Saaty [228, 227], is one of the best-known methods in this family. AHP helps decision-makers break down complex problems through three core principles: structuring the decision problem hierarchically, performing pairwise comparisons using a ratio scale, and synthesizing priorities [261].

AHP uses Saaty's fundamental 9-point scale for pairwise comparisons, allowing experts to express the importance of one element relative to others. The pairwise comparison matrix is reciprocal, with each entry defined by:

$$A = \begin{bmatrix} 1 & a_{12} & \cdots & a_{1n} \\ 1/a_{12} & 1 & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 1/a_{1n} & 1/a_{2n} & \cdots & 1 \end{bmatrix}$$

The weight vector representing the relative importance of each criterion can be derived either by solving the eigenvalue problem,

$$A\mathbf{w} = \lambda_{\max}\mathbf{w},$$

or via the geometric mean method,

$$w_i = \frac{\left(\prod_{j=1}^{n} a_{ij}\right)^{1/n}}{\sum_{k=1}^{n}\left(\prod_{j=1}^{n} a_{kj}\right)^{1/n}}.$$

After determining criteria weights, alternatives are compared pairwise under each criterion using a similar matrix structure. To ensure logical consistency in judgments, AHP calculates the Consistency Index (CI) and the Consistency Ratio (CR):

$$CI = \frac{\lambda_{\max} - n}{n-1}, \quad CR = \frac{CI}{RI}.$$

A CR below 0.1 indicates acceptable consistency. Final scores for each alternative are synthesized by multiplying the criteria weights by the performance scores for each alternative:

$$S_k = \sum_{i=1}^{n} w_i \cdot p_{ik}.$$

AHP has proven remarkably versatile across domains, such as ecological evaluation [20], supplier selection [29], energy planning [148], and healthcare decision-making [45], demonstrating its value as a flexible value-based MCDM approach.

Outranking Methods

Outranking methods, developed by the European school of decision-making, provide a more nuanced approach to modeling preferences [222]. Unlike simple ranking methods, they recognize that strict preference and indifference are not the only possible relationships between alternatives; weak preference and incomparability are also possible, making these methods suitable for complex, real-world decisions with incomplete or uncertain information. The ELimination Et Choix Traduisant la REalité (ELECTRE) family is among the most prominent examples [97].

The original ELECTRE I method, first documented by Roy and colleagues at SEMA in 1966 [33, 220], marked a significant departure from weighted-sum approaches. ELECTRE I does not seek a single optimal solution but instead identifies a kernel of superior alternatives through outranking relationships, even when options are incomparable [41].

This method builds on two key concepts: concordance and discordance indices. The concordance index quantifies the degree of evidence supporting that an alternative a is at least as good as b:

$$CI(a,b) = \frac{\sum_{j \in \psi(a,b)} w_j}{\sum_{j=1}^{n} w_j}$$

where w_j is the weight of criterion j and $\psi(a,b)$ is the set of criteria where a performs at least as well as b. The discordance index measures the extent to much b is strictly preferred to a:

$$DI(a,b) = \frac{\max_{j \in \Lambda(a,b)}[r(b,c_j) - r(a,c_j)]}{\max_{x,y}[r(x,c_j) - r(y,c_j)]}$$

Performance scores are first normalized:

$$r(a_i, c_j) = \frac{v(a_i, c_j)}{\sum_{j=1}^{n} v(a_i, c_j)}$$

An outranking relation exists if the concordance index exceeds a threshold CI^* and the discordance index remains below DI^*:

$$aOb \iff CI(a,b) \geq CI^* \text{ and } DI(a,b) \leq DI^*$$

Through these pairwise comparisons, ELECTRE I constructs a robust subset of feasible alternatives that demonstrate strong performance across multiple criteria [41]. This approach has proven valuable in complex applications where trade-offs between criteria make a single "best" choice unrealistic. ELECTRE I has been applied in domains such as ecotourism planning [202], water resources management [65], wastewater treatment [252], investment planning [17], and many others. Variants such as ELECTRE II, III, and IV extend the method to handle ranking or quasi-criteria problems [109, 223, 221, 224].

Recent developments continue to expand ELECTRE's flexibility through fuzzy logic and other integrations [117, 260]. Overall, the ELECTRE I method remains a well-established tool within the outranking family, known for its versatility in supporting complex, real-world multi-criteria decisions.

Distance-based Methods

Distance-based methods in MADM are founded on the principle that the best alternative should have the closest proximity to an ideal solution and the farthest distance from an anti-ideal (negative-ideal) solution [258]. This approach provides an intuitive geometric interpretation of alternative comparison.

The fundamental concept behind distance-based methods is the measurement of separation between alternatives and reference points (ideal and anti-ideal solutions) in a multi-dimensional space. Each criterion represents a dimension, and alternatives are viewed as points in this space [288]. The ideal solution represents a hypothetical best case that maximizes benefit criteria and minimizes cost criteria, while the negative-ideal solution does the opposite. A representative of these method is the Technique for Order Preference by Similarity to Ideal Solution (TOPSIS), developed by Hwang and Yoon [128]. TOPSIS evaluates alternatives through a decision matrix D containing m alternatives and n criteria:

$$D = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{m1} & x_{m2} & \cdots & x_{mn} \end{bmatrix}$$

where x_{ij} represents the outcome of the ith alternative with respect to the jth criterion.

The method first normalizes the decision matrix to enable comparisons across criteria. Each element r_{ij} of the normalized matrix R is calculated as:

$$r_{ij} = \frac{x_{ij}}{\sqrt{\sum_{i=1}^{m} x_{ij}^2}}$$

The weighted normalized decision matrix V is then constructed by multiplying each normalized value by its associated criterion weight w_j (where $\sum_{j=1}^{n} w_j = 1$):

$$v_{ij} = w_j r_{ij}$$

The ideal (A^*) and negative-ideal (A^-) solutions are determined as:

$$A^* = \{(\max_i v_{ij} | j \in J), (\min_i v_{ij} | j \in J') | i = 1, 2, ..., m\} = \{v_1^*, v_2^*, ..., v_n^*\}$$

$$A^- = \{(\min_i v_{ij} | j \in J), (\max_i v_{ij} | j \in J') | i = 1, 2, .., m\} = \{v_1^-, v_2^-, ..., v_n^-\}$$

where J represents the benefit criteria and J' represents the cost criteria.

The separation measures from the ideal (S_i^*) and negative-ideal (S_i^-) solutions are calculated using:

$$S_i^* = \sqrt{\sum_{j=1}^{n}(v_{ij} - v_j^*)^2}, \quad i = 1, 2, ..., m$$

$$S_i^- = \sqrt{\sum_{j=1}^{n}(v_{ij} - v_j^-)^2}, \quad i = 1, 2, ..., m$$

Finally, the relative closeness to the ideal solution is computed as:

$$C_i^* = \frac{S_i^-}{S_i^* + S_i^-}, \quad 0 \leq C_i^* \leq 1$$

where $C_i^* = 1$ if the alternative matches the ideal solution and $C_i^* = 0$ if it matches the negative-ideal solution. Alternatives are then ranked according to descending values of C_i^*.

TOPSIS has demonstrated its versatility across various fields. Behzadian et al. [31] provided a comprehensive literature review identifying key application areas.

In manufacturing and logistics, Parkan and Wu [208] applied TOPSIS for performance evaluation of manufacturing plants. Deng et al. [73] utilized TOPSIS for supplier selection in manufacturing systems. In environmental management, Soltanmohammadi et al. [244] employed TOPSIS for post-mining land-use determination.

Hybrid Methods

Hybrid methods combine two or more traditional MADM approaches to leverage their complementary strengths while addressing their individual limitations [258]. These hybrid approaches can be categorized into several types:

1. Integration of different MADM methods.
2. Extension with fuzzy set theory to handle uncertainty.
3. Enhancement via mathematical programming.

The Fuzzy Analytic Hierarchy Process [264] is one of these methods, which extends the traditional AHP by incorporating fuzzy logic to handle uncertainty in decision-making processes. Fuzzy AHP is particularly useful when decision-makers' judgments are not precise and can be better represented by fuzzy numbers.

Fuzzy numbers represent uncertain values with a range and a membership function. Triangular fuzzy numbers (TFNs) are commonly used, denoted as $\tilde{a} = (l, m, u)$, where l is the lower bound, m is the modal value, and u is the upper bound.

The usage of the method follows a similar pattern to the standard AHP method. Initially, the decision problem is structured hierarchically, with the goal at the top, criteria at an intermediate level, and the alternatives on the bottom. Following this, fuzzy numbers are used to represent pairwise comparisons, with fuzzy rations being provided by decision-makers. Employing this allows the decision makers to incorporate uncertainty in their preferences. For example, when comparing the importance of two criteria, instead of a crisp number like 3, a fuzzy judgment might be represented as $\tilde{a}_{ij} = (l_{ij}, m_{ij}, u_{ij})$. Having defined the importance of the criteria, fuzzy arithmetic is used to altercate the fuzzy weights of the criteria and alternatives. This involves operations like fuzzy addition or multiplication and taking the geometric mean of fuzzy numbers.

The geometric mean for a fuzzy number $\tilde{a}_{ij} = (l_{ij}, m_{ij}, u_{ij})$ is calculated as:

$$\tilde{r}_i = \left(\left(\prod_{j=1}^{n} l_{ij} \right)^{1/n}, \left(\prod_{j=1}^{n} m_{ij} \right)^{1/n}, \left(\prod_{j=1}^{n} u_{ij} \right)^{1/n} \right) \tag{2.4}$$

Afterward, the fuzzy weights are converted into crisp values using, e.g., the center of area (COA) method. The COA method for a triangular fuzzy number $\tilde{a} = (l, m, u)$ is given by:

$$\text{COA}(\tilde{a}) = \frac{l + m + u}{3}$$

This allows for the ranking of alternatives based on crisp scores. Although traditional consistency checks (such as CI and CR) are not directly applicable to fuzzy matrices, certain methods extend these concepts to fuzzy environments to ensure logical consistency in judgments. Finally, the weights of criteria and alternatives are combined. The final score for alternative k is computed as:

$$S_k = \sum_{i=1}^{n} w_i \cdot p_{ik}$$

The alternative with the highest score is considered the best choice. While fuzzy extensions of MADM methods (such as Fuzzy AHP or Fuzzy TOPSIS) are sometimes categorized as hybrid methods, they are more accurately described as extensions of traditional methods to handle uncertainty and imprecise information [136].

Hybrid approaches have been utilized in several fields to enhance decision-making processes:

In e-supply chain management performance, AHP was employed to determine criteria weights, and TOPSIS was used to rank alternatives [256].

In previous research, a fuzzy AHP-TOPSIS approach was applied to prioritize solutions for inverse reinforcement learning [147].

For smart sustainable waste management strategies, fuzzy AHP was used for weighting, and fuzzy TOPSIS was applied for ranking [72].

In healthcare, a hybrid AHP-TOPSIS model was proposed for selecting tomography equipment [212].

In shipyard selection, integrated AHP and TOPSIS were employed to enhance the decision-making process [257].

2.2 Robust Decision Making

Robust Decsion-making (RDM) is a framework of concepts, processes, and tools designed to shift the role of quantitative models and data from prediction to decision support in highly uncertain environments. Instead of using models to forecast outcomes and rank decisions, RDM leverages them to systematically assess assumptions, explore a wider range of possible futures, develop innovative responses to risks and opportunities, and navigate complex trade-offs. By focusing on improving decision quality rather than prediction accuracy, RDM helps decision-makers make more informed and resilient choices. [155, 156, 213].

Traditional Decision Analysis (DA) relies on an expected utility framework, which represents uncertainty using a single joint probability distribution over future states of the world. This probability distribution is often Bayesian, reflecting subjective judgments rather than purely frequentist interpretations. DA then applies optimality criteria to rank decision options, seeking the best possible choice given the assumed probabilities. In contrast, RDM views uncertainties as deep, meaning they are difficult to quantify precisely. Rather

than relying on a single probability distribution, RDM considers multiple possible distributions, drawing on the concept of imprecise probabilities. Instead of optimizing for the best outcome under a specific set of assumptions, RDM prioritizes robustness, selecting options that perform well across a wide range of possible futures. This approach is particularly useful for complex, uncertain challenges where future conditions may be unpredictable or contested, such as climate change or long-term policy planning [181].

RDM is built upon multiple analytical methodologies that, together, create a structured yet flexible framework for decision-making. These include:

- Scenario Planning: Constructing multiple plausible futures to explore a range of possible outcomes rather than relying on a single forecast [267, 235].
- Exploratory Modeling: Running numerous simulations to understand how strategies perform under different conditions [28].
- Assumption-Based Planning: Identifying key assumptions underlying strategies and evaluating their potential failures, enabling decision-makers to design more resilient policies [75].

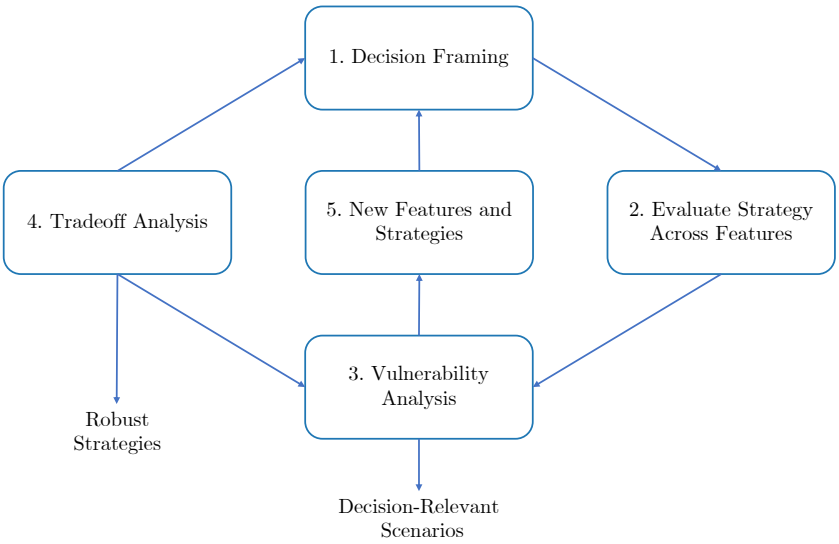

Figure 2.2: Steps in an RDM analysis. Source from [154] adapted from [181]

RDM explicitly follows a learning process called “deliberation with analysis” in which decision-making parties deliberate on their objectives and options [62]. The process consists of five steps [181], as depicted in Figure 2.4. The five steps are:

1. Decision Framing
   The RDM process begins with a decision-framing exercise, in which stakeholders identify the key factors in the analysis. These include the decision-makers’ objectives and criteria, the alternative actions available, the uncertainties that might influence the connection between actions and outcomes, and the relationships between actions, uncertainties, and objectives. This information is typically organized into a matrix called "XLRM," representing Exogenous Uncertainties (X), Policy Levers (L), Relationships (R), and Measures of Performance (M). This step ensures a comprehensive understanding of the problem by articulating the factors that will influence decision-making.

2. Evaluate Strategy Across Futures
   Once the factors have been defined, RDM evaluates the proposed strategies across a range of possible futures using simulation models. This step involves generating a large dataset from simulations, capturing each strategy's performance under various plausible scenarios. Strategies may originate from public debate, optimization routines, or a broad spectrum of simple strategies that are refined to more sophisticated alternatives. This evaluation helps assess the robustness of strategies under different future conditions.

3. Vulnerability Analysis
   In this step, analysts and decision-makers use visualization and data analytics to identify and assess vulnerabilities in the proposed strategies. Statistical tools, such as Scenario Discovery (SD) algorithms, are employed to categorize and display the key factors that distinguish futures where the strategies meet or fail to meet their goals. These "policy-relevant scenarios" highlight the vulnerabilities of policies, providing a clearer, reproducible, and unbiased understanding of strategy performance.

4. Trade-off Analysis
   The next step involves evaluating the trade-offs between different strategies using the identified policy-relevant scenarios. This analysis consists of comparing the performance of strategies on various objectives, such as reliability and cost, and visualizing the trade-offs through multi-objective curves. These analyses allow decision-makers to assess how to best balance their competing objectives in light of the uncertainties surrounding future scenarios.

5. New Futures and Strategies
   Based on the insights gained from the previous steps, analysts and decision-makers can identify more robust strategies—those that provide better trade-offs than the existing alternatives. These strategies may incorporate additional policy levers, often in the form of adaptive strategies, which include short-term actions, signposts, and contingent actions based on observed signals. In some cases, these strategies are developed using expert judgment, while in others, optimization algorithms are employed to determine the optimal combination of actions and signposts for adaptive strategies.

Many-Objective Robust Decision-Making

Several additional advanced methods have been developed in an effort to refine the RDM framework. Many-Objective Robust Decision Making (MORDM) extends traditional robust decision making by incorporating Multi-Objective Evolutionary Algorithms (MOEA) to explore trade-offs between competing objectives. Instead of relying solely on scenario analysis, this method seeks solutions that balance multiple decision criteria simultaneously. MORDM follows a process in which decision alternatives are optimized across many objectives and then tested for robustness against deep uncertainty. This approach is particularly useful in environmental and infrastructure planning, where decision-makers must consider trade-offs between economic, ecological, and social objectives. The outcome is typically a set of Pareto-optimal solutions that offer different trade-offs rather than a single best option [137].

Multi-Scenario Many-objective Robust Decision-Making

Multi-scenario MORDM builds upon MORDM by incorporating multiple scenarios during the optimization search phase rather than evaluating robustness only after identifying solutions. This adjustment prevents solutions from being overly optimized for a single reference scenario. By assessing candidate strategies across a range of possible futures early in the process, multi-scenario MORDM ensures that the identified solutions are broadly applicable and resilient to uncertainty. This method is particularly valuable in long-term strategic planning, where future conditions, such as climate change or economic shifts, are highly uncertain [272].

Many-Objective Robust Optimization

Many-Objective Robust Optimization (MORO) differs from MORDM in that it integrates robustness directly into the optimization process rather than treating it as a separate evaluation step [86]. In MORO, robustness is explicitly maximized alongside other performance metrics, ensuring that solutions are inherently stable across multiple uncertain futures. This approach is especially beneficial in high-stakes decision-making environments in which failing to meet performance criteria under uncertain conditions carries significant consequences. While MORO may lead to solutions that sacrifice some degree of optimality in idealized scenarios, it provides greater stability and reliability over a wide range of uncertain futures.

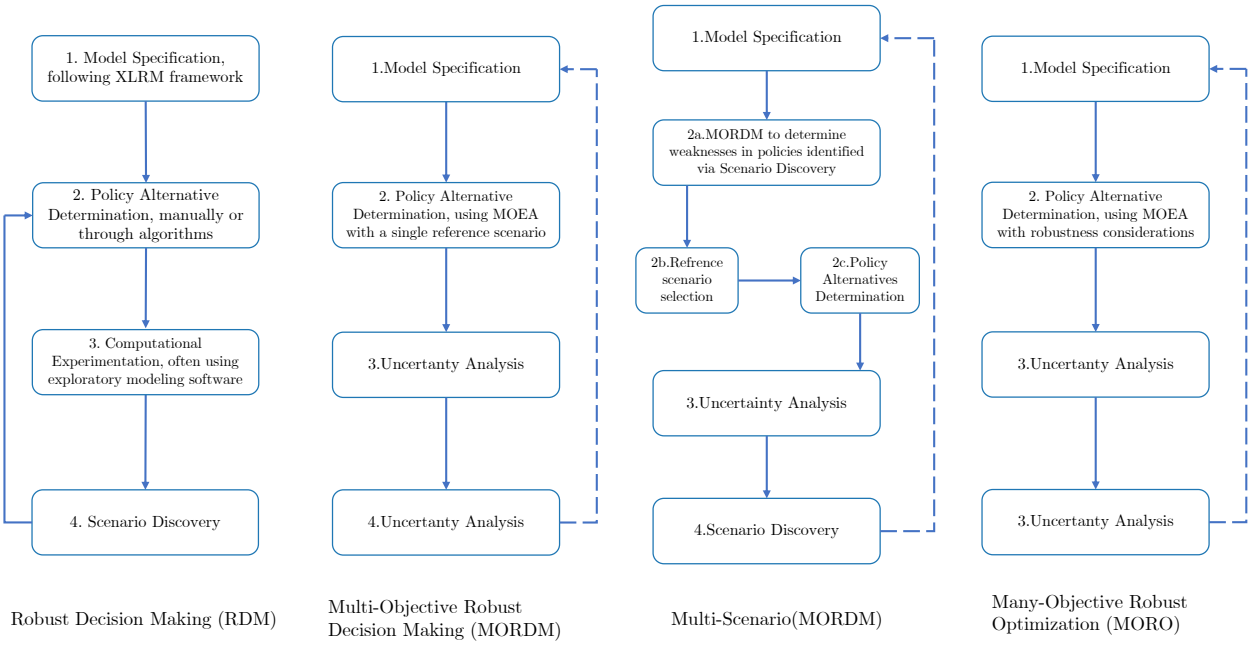

Figure 2.3: Four robust decision making approaches, adapted from [30]

By shifting the focus from prediction to robustness, RDM provides a powerful decision-support tool for navigating deep uncertainty. It allows policymakers to make informed, resilient choices without an over-reliance on uncertain forecasts, thereby improving long-term planning and reducing the risks of unexpected future developments [155, 153].

2.3 Machine learning

Having touched upon how the field of AI, and more specifically machine learning, has inserted itself into MCDM, a continuation and deeper exploration of the topic is warranted. The field of AI emerged from the fundamental question of whether machines can think, first formally posed by Alan Turing in his seminal 1950 paper "Computing Machinery and Intelligence" [255]. This work introduced the famous Turing Test and laid the groundwork for the field of AI. Since then, the field's relevance has only grown, especially as the amount of data in the world has increased, along with methods to both store and utilize it.

AI e encompasses a wide range of approaches and methodologies. However, the most prominent and widely applied paradigm today is Machine Learning (ML). Unlike early rule-based systems, which sought to emulate human decision making through a collection of human-written rules, ML focuses on developing algorithms that can learn directly from data, identifying patterns that may not

even be noticed by humans.

Machine Learning Definition

The term that gave the subfield its name and is widely used was defined in [187] as follows:

"A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E."

Based on the learning task being adressed and the learning feedback used, ML methods can be broadly categorized into three types: supervised, unsupervised, and reinforcement learning.

2.3.1 Supervised Learning

Supervised learning involves training a model on a labeled dataset, where the input data is paired with the known correct output. The goal or T is to then learn the input–output mapping that can be used to predict the output for new, unseen data [116].

Regresion

Regression is the supervised learning task used when the output variable one wishes to predict is continuous. The objective is to learn a function $f : \mathbb{R}^n \to \mathbb{R}$ that maps input features $\mathbf{x}$ to a continuous output y.

This function is learned from data by minimizing a loss function that quantifies the discrepancy between the predicted and true values. The choice of loss function directly affects the behavior of the model, including how it handles errors and outliers [116].

In regression tasks, two of the most commonly used loss functions are the mean squared error (MSE) and the mean absolute error (MAE). Both measure the difference between predicted values and ground truth targets, but they differ in how they penalize deviations [195].

The mean squared error (MSE) is defined as follows:

$$\mathcal{L}_{\text{MSE}}(\mathbf{w}, b) = \frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2 \tag{2.5}$$

where $y_i \in \mathbb{R}$ is the true target for the i-th sample and $\hat{y}_i$ is the model prediction. MSE penalizes larger errors more heavily due to the squared term, which makes it sensitive to outliers. It is commonly used when large deviations are particularly undesirable, and under the assumption that the error terms are normally distributed.

The mean absolute error (MAE) is an alternative that minimizes the average absolute difference:

$$\mathcal{L}_{\text{MAE}}(\mathbf{w}, b) = \frac{1}{N} \sum_{i=1}^{N} |y_i - \hat{y}_i| \tag{2.6}$$

MAE treats all deviations linearly, making it more robust to outliers and better suited to datasets where large deviations should not be disproportionately penalized. However, its non-differentiability at zero can make optimization slightly less smooth compared to MSE, though it is still widely used in practice [195].

Choosing between MSE and MAE depends on the specific application and the desired sensitivity to large errors. In some cases, hybrid loss functions such as the Huber loss are used to combine the advantages of both [116].

The most elementary approach for regression is linear regression, where the relationship between the inputs and the target is modeled as a weighted linear combination of the input features:

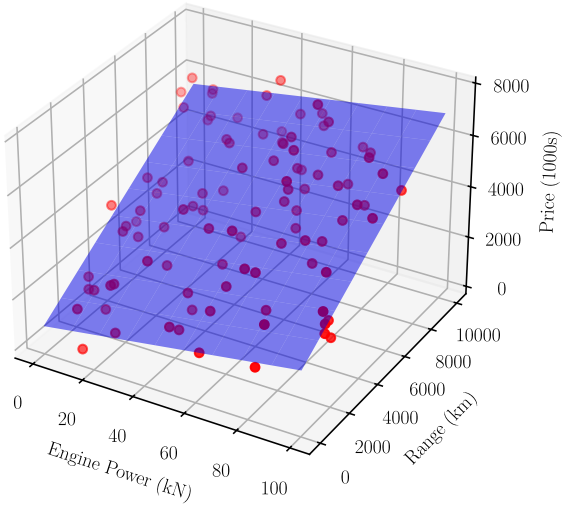

Figure 2.4: Private jet price prediction

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n + \epsilon \tag{2.7}$$

where $\beta_0, \beta_1, \ldots, \beta_n$ are the coefficients to be learned, and ϵ is the error term.

As a concrete example, the use of a linear regression to predict the price of a private jet is showcased. The features include engine power, number of seats, range, and age. The model can be expressed as:

$$\text{Price} = \beta_0 + \beta_1 \times \text{Engine Power} + \beta_2 \times \text{Number of Seats} + \beta_3 \times \text{Range} + \beta_4 \times \text{Age} + \epsilon \tag{2.8}$$

where β_0 is the intercept, $\beta_1, \beta_2, \beta_3$, and β_4, are the coefficients for each feature, and ϵ is the error term.

To better understand the model, one can visualize the predicted relationship in three dimensions by fixing certain variables and plotting how the price changes with two key features. For instance, the figure below illustrates how the predicted price varies as a function of engine power and range, assuming other variables are held constant.

Classification

Classification is a supervised learning task used when the target variable is categorical rather than continuous. The objective is to learn a function that maps an input feature vector $\mathbf{x}$ to one of several predefined classes. In the binary case, the model assigns each input to one of two categories, such as "positive" or "negative."

To train such models, a common approach is to minimize a probabilistic loss function that captures how well the predicted class probabilities align with the true labels. One of the most widely used loss functions for binary classification is log-loss, also known as binary cross-entropy loss [116]:

$$\mathcal{L}(\mathbf{w}, b) = -\frac{1}{N} \sum_{i=1}^{N} \left[y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i) \right] \tag{2.9}$$

Here, $y_i \in \{0, 1\}$ is the true label for the i-th sample, and $\hat{y}_i \in (0, 1)$ is the predicted probability that the sample belongs to class 1. These probabilities are typically obtained by applying sigmoid activation to a linear function of

the inputs:

$$\hat{y}_i = \sigma(\mathbf{w}^\top \mathbf{x}_i + b)$$

The cross-entropy loss penalizes confident but incorrect predictions more heavily than uncertain predictions. For example, if the true label is 0 but the model predicts a probability near 1, the loss is very high. This property makes cross-entropy particularly effective for training probabilistic models, as it encourages the model to produce confident predictions only when they are correct.

From an information-theoretic perspective, cross-entropy can be interpreted as the average number of extra bits required to encode the true labels using the predicted probability distribution. Minimizing this loss therefore helps to ensure that the model's output probabilities are both discriminative and well-calibrated [116].

Multi-class classification

While binary classification provides a foundational understanding of how models can distinguish between two categories, many real-world problems simple involve more than just two classes. Tasks such as handwriting recognition, document categorization, or disease diagnosis often require assigning inputs to one of several possible classes. This naturally leads to the need for multi-class classification [116].

In multi-class classification, the goal is to assign each instance to one of three or more distinct classes. Several strategies can be employed to adapt binary classifiers to this setting:

- One-vs.-All (OvA) involves training one classifier per class, where the samples belonging to the target class are treated as positive examples and all others as negatives. During inference, the class whose classifier yields the highest confidence score is selected.
- One-vs.-One (OvO) constructs a binary classifier for every pair of classes. Each classifier votes for one of its two classes, and the final prediction is made by selecting the class with the most votes across all pairwise comparisons.
- Regression (also known as multinomial logistic regression) directly models the probability distribution over all classes using the softmax function. For an input $\mathbf{x}$, the probability of class c is given by:

$$P(y = c \mid \mathbf{x}) = \frac{e^{\beta_c^\top \mathbf{x}}}{\sum_{j=1}^{C} e^{\beta_j^\top \mathbf{x}}} \tag{2.10}$$

where C is the total number of classes, and β_c is the parameter vector associated with class c. This formulation ensures that the predicted class probabilities are positive and sum to one, making it suitable for probabilistic interpretation and optimization using cross-entropy loss, generalized to the multi-class setting [116].

Performance metrics for classification

It stands to reason that after models have been trained, be it for binary or multi-class classification, their performances must be evaluated. Model evaluation is important to understand both how a model has learned the training data and how well it can generalize on unseen data.

Several performance metrics are commonly used to quantify classification accuracy. It is important to note that in many real-world problems, class distributions may not be perfectly balanced; thus, it is particularly important to include additional metrics where accuracy alone may be misleading. Among the most widely used are precision, recall, and the F_1 score [116]. These evaluation rely on the following basic classification outcomes:

- True Positives (TP): Correctly predicted positive instances.

- True Negatives (TN): Correctly predicted negative instances.
- False Positives (FP): Instances incorrectly predicted as positive.
- False Negatives (FN): Instances incorrectly predicted as negative.

Precision measures the proportion of positive predictions that are actually correct:

$$\text{Precision} = \frac{TP}{TP + FP} \tag{2.11}$$

Recall, also called sensitivity, measures the proportion of actual positives that were correctly identified:

$$\text{Recall} = \frac{TP}{TP + FN} \tag{2.12}$$

The F_1 score provides a single performance measure that balances precision and recall. It is defined as the harmonic mean of the two:

$$F_1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{2.13}$$

In the case of multi-class classification, precision, recall, and F_1 can be extended using averaging strategies:

- Macro-averaging computes the metric independently for each class and takes the unweighted mean.
- Micro-averaging aggregates all true positives, false positives, and false negatives across classes before computing the metric, effectively weighting by class frequency.

These evaluation metrics provide a more nuanced and reliable view of model effectiveness than accuracy alone, helping practitioners identify failure modes and improve robustness [214].

Learning to Rank

Learning to Rank refers to the application of ML techniques to train a ranking model that sorts items based on their relevance to a given input. Formally, given training data consisting of queries, documents, or options, and their associated relevance judgments, the goal is to learn a function $f(q, d)$ that assigns a score to each item d for a given query q. When items are sorted by this score in descending order, the output should reflect the correct relevance ranking. Unlike traditional regression or classification, the goal is concerned with producing correct orderings rather than predicting exact values or class labels [172, 159].

Learning-to-rank methods are generally divided into three categories: *pointwise*, *pairwise*, and *listwise* [172]. Each of these approaches models the ranking task differently based on how it processes the training data:

1. Pointwise Approach [159]: Each item is treated independently. The ranking problem is reduced to a standard regression or classification task where the model predicts a relevance score $\hat{y}_i$ for a single item x_i. A common loss function in this setting is the mean squared error:

$$\mathcal{L}_{\text{pointwise}} = \frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2$$

where y_i is the true relevance score of item i. While simple and fast to train, pointwise methods ignore relationships between items, which may limit their ranking effectiveness.

2. Pairwise Approach [48, 133]: This approach considers pairs of items and learns a binary preference function. The model is trained to predict which item in a pair should be ranked higher. For example, RankNet uses a probabilistic model based on a logistic function:

$$P(a_i \succ a_j) = \frac{1}{1 + e^{-(\hat{y}_i - \hat{y}_j)}}$$

The corresponding loss function is the binary cross-entropy over pairs:

$$\mathcal{L}_{\text{pairwise}} = -\sum_{(i,j)\in\mathcal{P}} [S_{ij} \log P_{ij} + (1 - S_{ij}) \log(1 - P_{ij})]$$

where $S_{ij} = 1$ if item i should rank higher than item j, and P_{ij} is the predicted probability. This approach effectively models relative preferences and is more aligned with ranking tasks than pointwise methods.

3. Listwise Approach [51, 47]: This approach treats the entire ranked list as the training dataset and attempts to directly optimize a ranking metric (e.g., NDCG, MAP). One example is ListNet, which models the permutation probability of the ground truth and predicted rankings using the softmax function:

$$P(y_i) = \frac{e^{\hat{y}_i}}{\sum_j e^{\hat{y}_j}}$$

The loss is typically the cross-entropy between the predicted distribution and the ground-truth relevance distribution:

$$\mathcal{L}_{\text{listwise}} = -\sum_{i=1}^{N} P^{\text{true}}(y_i) \log P^{\text{pred}}(y_i)$$

Listwise methods, such as LambdaMART [278], extend this by using gradient-boosted decision trees and differentiable surrogates of NDCG. These methods offer the closest alignment between training objectives and evaluation metrics.

Evaluation Metrics for Learning to Rank methods

Learning-to-rank models are evaluated using rank-sensitive metrics that reflect the quality of the produced ordering.

Precision@k measures the fraction of relevant items in the top-k ranked positions:

$$\text{P@}k = \frac{1}{k} \sum_{r=1}^{k} \mathbb{I}(r \text{ is relevant}).$$

Mean Average Precision (MAP) is the mean of the precision scores computed at the rank positions of relevant items, averaged over all queries [26].

$$\text{MAP} = \frac{1}{Q} \sum_{q=1}^{Q} \left(\frac{1}{|R_q|} \sum_{k\in R_q} P_q(k) \right) \tag{2.14}$$

where Q denotes the total number of queries, the set R_q contains the rank positions where relevant items appear in the retrieved list, $|R_q|$ is the total number of relevant items for that query, and $P_q(k)$ represents the precision at rank k, which is the proportion of relevant items among the top k retrieved results for query q.

Normalized Discounted Cumulative Gain (NDCG) accounts for both the position and graded relevance of items. DCG at rank k is computed as:

$$\text{DCG@}k = \sum_{r=1}^{k} \frac{2^{\text{rel}(r)} - 1}{\log_2(r + 1)},$$

and NDCG@k is obtained by dividing DCG by the ideal DCG@k for that query, producing a value between 0 and 1 [131, 111].

Unlike regression or classification, Learning to Rank seeks to optimize a model such that the induced ordering of items maximizes a ranking metric (e.g., NDCG, MAP). The model is not evaluated based on whether individual predictions are accurate, but rather on whether the sorted list matches the true relevance order [159]. Many surrogate losses used in training (e.g., logistic or hinge loss) serve as upper bounds on ranking metric loss, making them tractable to optimize in practice [172].

In summary, Learning-to-rank methods aim to learn a model that outputs a ranking of items tailored to a specific query or context. Pointwise methods offer simplicity, pairwise methods capture relative preferences, and listwise methods often yield the best alignment with ranking objectives. Evaluation is based on metrics that reward correct ordering, and recent advances have improved both the performance and interpretability of ranking models [111].

Weakly supervised learning

Supervised learning techniques often require a large number of ground-truth-labeled training data in order to successfully learn. However, acquiring the myriad labels needed for strong supervision information can be rather expensive and time-consuming, so efforts have been made regarding ML methods to learn from weak supervision [296].

Typically, weak supervision approaches can be grouped into three types [296]:

1. Incomplete supervision, in which only a small subset of data from the dataset has labels while the rest are unlabeled. A common strategy in this setting is using semi-supervised learning methods that leverage the structure of the unlabeled data to improve generalization, such as graph-based label propagation or consistency regularization.

2. Inexact supervision, in which certain supervision information is provided, albeit less precise than desired. For example, instead of instance-level labels, supervision may occur in the form of group-level labels or ranking preferences, whereby models are trained to infer instance-level signals using techniques like multi-instance learning or expectation maximization.

3. Inaccurate supervision, in which the labels provided are not exactly ground truth and may suffer from errors. This is often addressed by modeling label noise explicitly or using robust loss functions, noise transition matrices, or probabilistic methods to denoise the labels during training.

One example of a system designed to support learning from weak supervision is Snorkel [215]. In Snorkel, supervision is provided through multiple noisy or heuristic labeling functions instead of manually annotated labels. These functions, denoted as $\lambda_1, \lambda_2, \ldots, \lambda_n$, each map an instance x_i to a label in $\{0, 1, \perp\}$, where $\perp$ indicates an abstention. The outputs of the labeling functions form a label matrix $\Lambda \in \{0, 1, \perp\}^{m \times n}$, where m is the number of instances and n is the number of labeling functions. Snorkel then learns a generative label model $P_\theta(Y \mid \Lambda)$ that estimates the latent true label Y given the noisy labels in Λ, by modeling the accuracy and correlation of each λ_j. This makes it possible to generate training labels from weak sources, allowing ML models to be trained without relying on extensive ground-truth annotation [215].

2.3.2 Unsupervised Learning

Unsupervised learning deals with unlabeled data, where no ground truth is available. The aim is then to uncover hidden patterns or structures within the data. It focuses on creating clusters of similar data points to help reveal these structures. Another important aspect is the use of dimensionality-reduction techniques, in which the data is transformed in an effort to more clearly capture its underlying characteristics. This not only reduces the dimensionality of the data but also enables it to be viewed and inspected from different perspectives.

In many cases, it even helps in visualizing high-dimensional data, making it more understandable to humans [197].

Unlike in supervised learning, where a known ground truth is established and can be used to gauge how well the task is performed, unsupervised learning relies on alternative evaluation methods. In clustering, the quality of the resulting clusters is typically assessed using metrics that capture intra-cluster similarity and inter-cluster separation. In the case of dimensionality reduction, the focus is on how well the transformed data represents the original dataset, ideally preserving its structure and minimizing information loss [197].

2.3.3 Reinforcement Learning

Reinforcement Learning (RL) is a type of ML where an agent learns to make decisions by interacting with an environment. The agent's goal is to maximize cumulative rewards over time. The basic components of RL include the agent, environment, state (s), action (a), and reward (r). The goal in reinforcement learning is to devise a policy (π), that determines the behavior of the agent given a certain state of the environment [250].

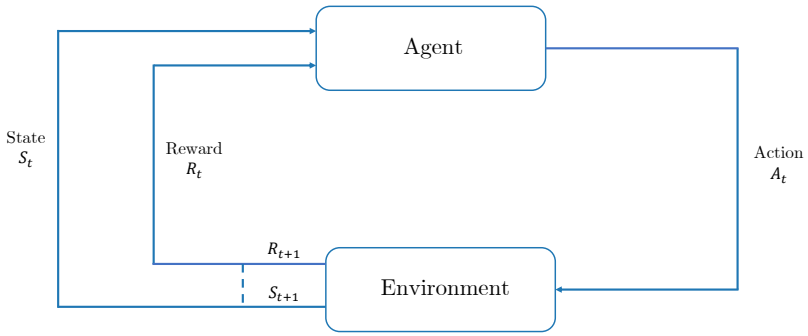

Figure 2.5: Reinforcement learning

Many reinforcement learning problems are modeled as a Markov Decision Process (MDP). MDP is a mathematical framework used to describe decision-making in which outcomes of the future state depends only on the current state and action. An MDP is formally defined by the tuple $(\mathcal{S}, \mathcal{A}, P, R, \gamma)$, where

$\mathcal{S}$ is the set of states; $\mathcal{A}$ is the set of actions; $P(s' \mid s, a)$ is the state transition probability, i.e., the probability of transitioning to state s' when taking action a in state s; $R(s, a)$ is the reward function, giving the immediate reward received after taking action a in state s; and $\gamma \in [0, 1]$ is the discount factor, representing the preference for immediate rewards over future rewards.

The way the MDP is structured makes it suitable for many reinforcement learning problems, especially those involving sequential decision-making [250].

Bandits

The Multi-Armed Bandit Problem is a simplified RL problem in which an agent must choose between multiple options, often refereed to as arms, to maximize rewards. Each arm provides a random reward from a probability distribution. The key challenge is balancing exploration, by testing new arms and exploitation, which results in selecting the arms that are known to provide good rewards [23].

Contextual Multi-Armed Bandits

Contextual Multi-Armed Bandits extend the traditional multi-armed bandit problem by incorporating context or side information into the decision-making process. In a such a setting, the agent observes a context vector before selecting an arm, allowing it to tailor its actions based on the current situation.

This approach is particularly useful in scenarios in which the reward distribution of each arm depends on the context, such as personalized news articles recommendations or adaptive clinical trials [160].

Mathematical Formulation

In a contextual Multi-Armed Bandit problem, at each time step t, the agent observes a context vector $\mathbf{x}_t \in \mathbb{R}^d$ and selects an arm a_t from a set of available arms. The reward r_t received is a function of both the chosen arm and the context:

$$r_t = f(a_t, \mathbf{x}_t) + \epsilon_t \tag{2.15}$$

where $f(a_t, \mathbf{x}_t)$ is the expected reward function, and ϵ_t is a noise term.

The goal is to maximize the cumulative reward over time:

$$\text{Maximize} \sum_{t=1}^{T} r_t \tag{2.16}$$

A common approach to solving these problems is to use linear models, such as LinUCB [160] and Linear Thompson Sampling [11], where the expected reward is assumed to be a linear function of the context:

$$f(a, \mathbf{x}) = \theta_a^\top \mathbf{x} \tag{2.17}$$

where θ_a is a parameter vector specific to arm a.

The agent's objective is to then learn the parameter vectors θ_a for each arm in order to make informed decisions that will maximize the expected reward. This involves balancing exploration (testing different arms to learn their reward distributions) and exploitation (choosing the arm with the highest expected reward).

2.4 Ensemble Methods and Gradient Boosting

Ensemble learning methods aim to improve predictive performance by combining the outputs of multiple individual models. These approaches are based on the idea that a group of diverse learners, when properly aggregated, can outperform any single model [104]. A widely used base model for this task is the decision tree.

Decision trees

Decision trees are simple models that make predictions by recursively splitting the data into regions based on feature values. Each internal node compares a feature to a threshold and routes the input left or right, depending on the outcome. This process continues until a leaf node is reached, which contains a fixed prediction. Regarding regression, the prediction is often the average of the target values in that region. Trees are constructed by selecting splits that minimize a loss function, such as mean squared error [44].

Gradient boosting

While decision trees are easy to interpret and fast to train, individual trees often overfit the training data. To address this, ensemble methods, such as boosting, combine many trees to improve accuracy and robustness. In this setting, individual trees are trained sequentially, with each new model attempting to correct the residual errors of the ensemble so far. This strategy is known as gradient boosting [104].

XGBoost

Extreme Gradient Boosting (XGBoost) is a powerful ensemble learning method based on decision trees, designed for speed, accuracy, and efficiency. It has become a popular choice across a range of ML tasks, particularly structured data problems, due to its ability to combine predictive power with computational scalability [57].

Mathematically, the model builds an additive function:

$$\hat{y}_i = \sum_{k=1}^{K} f_k(x_i), \quad f_k \in \mathcal{F} \tag{2.18}$$

where $\mathcal{F}$ is the space of regression trees, K is the number of boosting rounds, and each f_k is a tree that maps the input x_i to a score.

The learning objective includes both a loss function $\mathcal{L}$ measuring the prediction error and a regularization term Ω to penalize model complexity:

$$\mathcal{L}_{\text{total}} = \sum_{i=1}^{n} \ell(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k) \tag{2.19}$$

The regularization term encourages simpler models:

$$\Omega(f) = \gamma T + \frac{1}{2}\lambda \|w\|^2 \tag{2.20}$$

where T is the number of leaves in the tree, w represents the leaf weights, γ controls tree complexity, and λ regularizes the weights.

During training, XGBoost constructs trees greedily, choosing splits that optimize a second-order Taylor expansion of the loss function. For each split, the algorithm evaluates the *gain* in loss reduction:

$$\text{Gain} = \frac{1}{2}\left[\frac{G_L^2}{H_L + \lambda} + \frac{G_R^2}{H_R + \lambda} - \frac{(G_L + G_R)^2}{H_L + H_R + \lambda}\right] - \gamma \tag{2.21}$$

where G and H are the accumulated first and second derivatives (gradient and Hessian) of the loss function over the left and right children. This allows XGBoost to use rich optimization techniques not available to traditional tree learners [57].

Given an input feature vector x, the prediction $\hat{y}$ is the sum of scores from all trees:

$$\hat{y} = \sum_{k=1}^{K} f_k(x) \tag{2.22}$$

Each tree f_k is added iteratively by minimizing the objective:

$$\mathcal{L}^{(t)} = \sum_{i=1}^{n} \left[\ell(y_i, \hat{y}_i^{(t-1)} + f_t(x_i))\right] + \Omega(f_t) \tag{2.23}$$

This is optimized using a greedy tree construction algorithm that selects the best split by maximizing the gain.

XGBoost key aspects

XGBoost introduces several key innovations that contribute to its strong empirical performance and widespread adoption. It includes regularization through both ℓ_2 penalties on the leaf weights and a complexity term penalizing the number of leaves, which helps reduce overfitting and improve generalization [57]. The algorithm employs second-order optimization by utilizing both the first and second derivatives of the loss function. This allows for more accurate and stable split decisions during tree construction. Additionally, XGBoost performs column subsampling, where a random subset of features is used when constructing each tree. This increases model diversity and improves its ability to generalize. Another notable feature is its sparsity-aware algorithm, which can handle missing or sparse data by automatically learning the optimal default directions in tree splits. Furthermore, XGBoost supports parallel computation across features during split finding, enabling fast and efficient training, even on large datasets [57, 34].

Comparative studies have showcased XGBoost's strong performance, especially on tabular data, often surpassing Random Forests and standard gradient boosting regarding speed and accuracy [34].

2.5 Neural Networks

Neural Network (NN)s are powerful function approximators inspired by biological neurons, which are specific types of cells most commonly found in animal brains [118]. Much like their biological counterparts, NNs are composed of layers of interconnected neurons, each performing simple computations, which together can model complex mappings. Mathematically, an NNs defines a parameterized function:

$$f_\theta : \mathbb{R}^n \to \mathbb{R}^m \tag{2.24}$$

where θ denotes the trainable weights and biases across the layers.

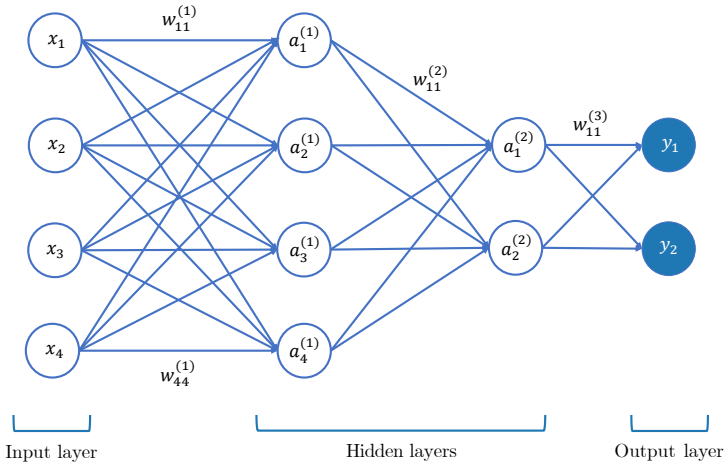

Figure 2.6: Feedforward Neural Network Diagram

A typical feedforward network, as depicted in Figure 2.6, consists of L layers. Each layer performs an affine transformation followed by a non-linear activation. Given an input vector $x \in \mathbb{R}^d$, the computation steps are as follows:

$$a^{(0)} = x \quad \text{(input layer)} \tag{2.25}$$

$$z^{(l)} = W^{(l)} a^{(l-1)} + b^{(l)} \quad \text{(linear transformation)} \tag{2.26}$$

$$a^{(l)} = \sigma^{(l)}(z^{(l)}) \quad \text{(non-linear activation)} \tag{2.27}$$

where: $W^{(l)} \in \mathbb{R}^{n_l \times n_{l-1}}$ is the weight matrix for layer l, $b^{(l)} \in \mathbb{R}^{n_l}$ is a bias vector, $z^{(l)}$ is a pre-activation linear output, $a^{(l)}$ is an activation output, and $\sigma^{(l)}$ is a non-linear activation function (ReLU, Sigmoid, Hyperbolic tangent).

$$\sigma(z) = \begin{cases} \max(0, z) & \text{(ReLU)} \\ \frac{1}{1+e^{-z}} & \text{(Sigmoid)} \\ \tanh(z) & \text{(Hyperbolic tangent)} \end{cases}$$

The output of the final layer, $a^{(L)}$, is denoted as $\hat{y}$ and represents the network's prediction.

To train the network (in order for the model to learn), a loss function $\mathcal{L}(\hat{y}, y)$ is used, which quantifies the error between the prediction $\hat{y}$ and the true target y. For a dataset $\{(x^{(i)}, y^{(i)})\}_{i=1}^N$. The two most common loss functions include:

Mean Squared Error (MSE) for Regression

$$\mathcal{L}_{\text{MSE}} = \frac{1}{N} \sum_{i=1}^{N} \|\hat{y}^{(i)} - y^{(i)}\|^2 \tag{2.28}$$

$$\mathcal{L}_{\text{CE}} = -\frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{C} y_j^{(i)} \log(\hat{y}_j^{(i)}) \tag{2.29}$$

where C is the number of classes.

The learning process involves minimizing the loss over the training set by adjusting the network parameters θ. This is commonly achieved using the gradient descent, where the parameters are updated iteratively in the opposite direction to the gradient:

$$\theta \leftarrow \theta - \eta \nabla_\theta \mathcal{L}(\theta) \tag{2.30}$$

where $\eta > 0$ is the learning rate controlling step size and $\nabla_\theta \mathcal{L}$ is the gradient of the loss with respect to the parameters. In practice, the gradient $\nabla_\theta \mathcal{L}$ is commonly computed using the backpropagation algorithm [226].

Backpropagation

The core idea of backpropagation is to compute the gradient of the loss with respect to each parameter in the network, layer by layer, starting from the output layer and propagating backwards.

For example, given the *error signal* $\delta^{(l)}$ at layer l:

$$\delta^{(l)} = \frac{\partial \mathcal{L}}{\partial z^{(l)}} \tag{2.31}$$

which represents the sensitivity of the loss to the linear output $z^{(l)}$ of layer l. The backpropagation process unfolds in the following steps steps:

1. Output Layer Gradient:
   At the output layer L, we compute the gradient of the loss with respect to the activation $a^{(L)}$; then, the chain rule is used:

   $$\delta^{(L)} = \frac{\partial \mathcal{L}}{\partial a^{(L)}} \odot \sigma'^{(L)}(z^{(L)}) \tag{2.32}$$

   This uses the element-wise derivative of the activation function $\sigma^{(L)}$.

2. Recursive Backward Propagation:
   For each hidden layer $l = L-1, \ldots, 1$, the error signal is computed recursively:

   $$\delta^{(l)} = (W^{(l+1)})^T \delta^{(l+1)} \odot \sigma'^{(l)}(z^{(l)}) \tag{2.33}$$

   This step functions based on the fact that the gradient of the loss with respect to $z^{(l)}$ depends on the gradient from the next layer and the local derivative of the activation function.

3. Gradient of the Parameters:
   Once the error signals $\delta^{(l)}$ are known, we can compute the gradient of the loss with respect to the parameters:

   $$\frac{\partial \mathcal{L}}{\partial W^{(l)}} = \delta^{(l)} (a^{(l-1)})^T \tag{2.34}$$

   $$\frac{\partial \mathcal{L}}{\partial b^{(l)}} = \delta^{(l)} \tag{2.35}$$

   This arises from the chain rule, in which the gradient of the loss with respect to a weight is the product of the error and the activation from the previous layer.

The overall complexity of backpropagation is linear regarding the number of weights and activations, making it efficient and scalable. However, updates of the model's parameters using gradients computed over the entire training set can still be computationally expensive; therefore, the Stochastic Gradient Descent is often employed [219]. It is a practical alternative that works by approximating the true gradient using a smaller subset (mini-batch) of the dataset each iteration. This also introduces noise into the updates but allows the model to explore the loss landscape more efficiently, often escaping shallow local minima and speeding up convergence.

Universal Approximation Theorem

A key reason for the success of NNs is their remarkable ability to approximate complex functions. This capability is formally supported by the Universal Approximation Theorem [126, 63], which states that a feedforward NNs with a single hidden layer containing a finite number of neurons can approximate any continuous function on a compact subset of $\mathbb{R}^n$, given a suitable non-linear activation function.

The activation function σ plays a crucial role in this result. It introduces non-linearity into the network, allowing it to model intricate patterns and decision boundaries. Without non-linear activations, a NNs would be reduced to a series of linear transformations and could not represent complex mappings. In practice, deeper networks are often used to improve training efficiency and generalization, but the universal approximation result provides a foundational guarantee that NNs are capable of representing the wide class of functions needed in real-world learning tasks.

2.6 Learning Classifier Systems

Origin and Concept

Learning Classifier Systems (LCS) were introduced by John H. Holland as part of his pioneering work on adaptive systems in the 1970s and 1980s. Holland envisioned a framework in which a population of *if–then* rules, called "classifiers", could collectively adapt to solve problems, a strategy somewhat inspired by biological evolution and learning in complex adaptive systems [125]. In essence, an LCS is a rule-based machine learning approach that evolves and learns a set of condition–action rules online. Holland's early LCS (e.g., *CS-1*) aimed to model an agent in an unknown environment by enabling rules to evolve and credit to propagate based on received rewards, thereby overcoming the inflexibility of hand-coded expert systems [125]. This concept positioned LCS in the broader context of complex adaptive systems, viewing the learning system as a population of interacting, adaptive rules that collectively produce intelligent behavior. Holland's approach is often termed the *Michigan* style of LCS ,where learning is performed within a single rule population. In contrast, the *Pittsburgh* style also exists, where entire rule sets evolve as individuals [259]. The Michigan LCS paradigm, originated by Holland, became the standard for subsequent research [125].

Paradigms

LCS combines three key paradigms: reinforcement learning for credit assignment, evolutionary algorithms for rule discovery, and rule-based knowledge representation [275, 259]. The knowledge in an LCS is encoded as a set of explicit rules, which helps to ensure the learned model is human-interpretable as a collection of conditions and associated actions or predictions. Each rule, also called a classifier, typically has an associated strength or utility estimate reflecting the expected payoff. The system learns through interactions with an environment, using reinforcement learning techniques to update these rule parameters based on the feedback received [249]. At the same time, a genetic algorithm drives exploration by evolving the rule population. Rules that have a higher fitness are more likely to be selected to reproduce, thus creating new candidate rules and eliminating poorly performing rules [125]. By integrating reinforcment learning with an evolutionary search in a rule-based framework, LCS perform a variation of global searching for useful hypotheses in tandem with local incremental learning from reward. For example, early LCS used Holland's bucket brigade algorithm to propagate rewards internally [125], whereas modern variants, such as Wilson's XCS, employ explicit temporal-difference methods for credit assignment [275, 249]. This hybrid design allows LCS to continually discover, evaluate, and refine a set of decision rules, making it well-suited to complex problems where the solution may require balancing exploration and exploitation across a large, unknown search space.

Typical Architecture of an LCS

Conceptually, an LCS consists of three primary subsystems [259]:

- Performance Component: Interfaces with the environment by sensing

inputs and selecting actions.

- Reinforcement Component: Applies credit assignment and updates rule parameters based on feedback.
- Discovery Component: Generates new rules and removes or generalizes existing ones, usually via a genetic algorithm.

These three components work together in a tightly coupled learning cycle, where each interaction with the environment triggers a series of steps that enable the system to sense, decide, learn, and adapt. The typical sequence of operations in an LCS is outlined below:

1. The system perceives the current state from the environment through input sensors and encodes it in a format suitable for the rules. This input constitutes the situation to which the LCS must respond.
2. The LCS maintains a population $[P]$ of classifiers, each with a condition that specifies the states in which it is applicable. All classifiers in $[P]$ are validated against the current state, and those whose condition matches form the match set $[M]$.
3. If no rule in the population matches the current state, the LCS invokes a covering mechanism to create a new rule that covers that state [259]. The new rule is added to $[P]$ and $[M]$.
4. Given $[M]$, the LCS computes a predicted payoff for each action a using:
$$P(a) = \frac{\sum_{i \in [M]_a} F_i \cdot p_i}{\sum_{i \in [M]_a} F_i}, \tag{2.36}$$
where $[M]_a$ is the subset of classifiers in $[M]$ advocating a. The action with the highest $P(a)$ is selected using a variety of selection mechanisms, with ϵ-greedy exploration being a prime example [249]..
5. All classifiers in $[M]$ that proposed the chosen action form the action set $[A]$.
6. The system executes the action on the environment via its effectors.
7. The environment returns a reward r, and possibly a new state, which the LCS uses as feedback.
8. The system updates each p_i for classifiers in $[A]$ using:
$$p_i \leftarrow p_i + \beta \left[r + \gamma \max_{a'} P_{t+1}(a') - p_i \right], \tag{2.37}$$
where β is the learning rate and γ is the discount factor.
9. The fitness F_i is updated based on the error ϵ_i using:
$$\kappa_i = \begin{cases} 1, & \text{if } \epsilon_i < \epsilon_0, \\ \left(\frac{\epsilon_i}{\epsilon_0}\right)^{-\nu}, & \text{if } \epsilon_i \geq \epsilon_0, \end{cases} \tag{2.38}$$
and F_i is moved toward κ_i.
10. The Genetic Algorithm selects parents from $[A]$, applies crossover and mutation, and inserts offspring into $[P]$, replacing weaker classifiers [275].

These ten steps are repeated as the LCS interacts with the environment. Steps 1–7 form the performance component, steps 8–9 constitute the reinforcement component, and step 10 contains the discovery component.

Advantages of LCS LCS are inherently adaptive to non-stationary environments [259]. They dy-

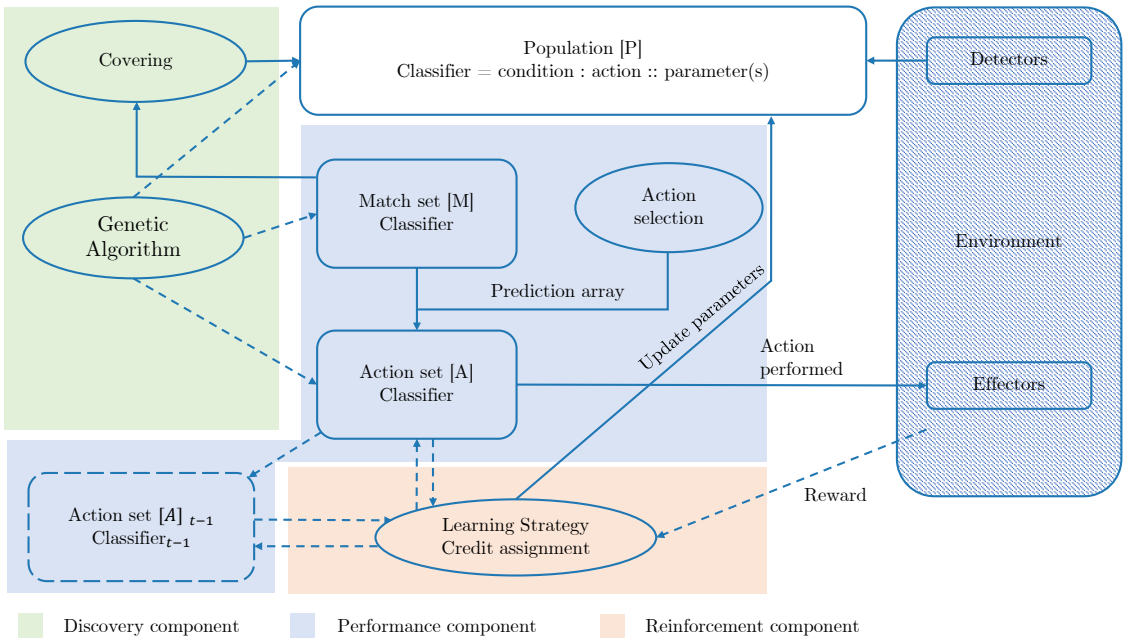

Figure 2.7: LCS diagram adapted from [259]

namically evolve and refine their rule population based on ongoing feedback. Unlike static models, LCS can accommodate shifting reward structures and concept drift. Moreover, LCS break down decision spaces into modular rules, enabling effective coverage of heterogeneous domains. Accuracy-based LCS like XCS evolve complete and general mappings from states to actions, yielding robust and transparent policies [275]. LCS also excel in environments with sparse or delayed rewards. Their reinforcement learning mechanisms can assign credit to actions across multiple steps, and genetic algorithms support exploratory rule creation, which facilitates better long-term policies. Finally, their rule-based nature makes them interpretable and transparent, making LCS suitable for domains where explainability is critical, such as healthcare, cybersecurity, and robotics [259].

2.7 Explainable and Interpretable AI

Explainable Artificial Intelligence (XAI) refers to methods and techniques that make the outputs of AI systems understandable to humans. As AI systems are increasingly deployed in critical domains such as healthcare, finance, and criminal justice, the need for transparency and accountability is paramount [22]. XAI aims to bridge the gap between complex, often opaque models and the human need for comprehension, trust, and control.

In the 1970s and 1980s, rule-based expert systems were used to emulate human decision-making. Due to their straightforward rule-based nature, these systems were inherently transparent, allowing users to trace the reasoning behind decisions. However, the advent of more complex models, such as NNs and ensemble methods, introduced challenges in interpretability due to their "black-box" nature [193, 22], which limited full comprehension by humans.

Thus, the interest in XAI is driven by both practical considerations for improving models and broader concerns about their real-world deployment. Initially, understanding model behavior helps with debugging, enhancing performance and ensuring robustness. However, as models are developed and applied in practice, ethical considerations and transparency become increasingly important for identifying biases, promoting fairness, and meeting legal obli-

gations such as the EU's General Data Protection Regulation (GDPR) [108] and the European Union Aviation Safety Agency (EASA) Artificial Intelligence Roadmap [91]. These frameworks emphasize the right to explanation in automated decision-making and advocate for a human-centric approach to AI, particularly in safety-critical domains, such as aviation.

Definitions and Goals of Interpretability and Explainability

Several definitions of interpretability have been proposed in the literature. For instance, Biran and Cotton [35] define it as "the degree to which a human can understand the cause of a decision," while Kim, Khanna, and Koyejo [142] describe an interpretable method as one where "a user can correctly and efficiently predict the method's results." Despite these efforts, clear and consistent definitions of both interpretability and explainability remain elusive, and the terms are often used interchangeably. However, a useful distinction is that interpretability typically refers to a model's transparency, i.e., the degree to which its internal mechanics can be understood by humans, whereas explainability pertains to post hoc methods that provide human-understandable justifications for a model's behavior, even if its internal workings remain opaque [167].

Based on [185], the primary goals of XAI include the following:

- Transparency: Providing insights into how models make decisions.
- Trust: Building user confidence in AI systems.
- Fairness: Detecting and mitigating biases in decision-making.
- Accountability: Enabling stakeholders to hold systems responsible for their outcomes.
- Compliance: Meeting legal and ethical standards for automated decisions.
- Debugging and Model Improvement: Providing aid to developers and data scientists in diagnosing model errors and improving performance.

Interpretability of models

Although specific methods have been developed to enhance the interpretability of machine learning models, it is important to note that some models are inherently more interpretable than others due to their structure. Models such as linear regression, decision trees, and rule-based systems are generally considered highly interpretable. These models offer clear and direct mappings between input features and outputs, allowing users to easily trace how predictions are made. For example, in linear regression, the influence of each feature is represented by an explicit coefficient, and in decision trees, the decision path can be followed node by node. In contrast, ensemble methods, such as random forests and gradient boosting machines, are often classified as moderately interpretable. While they are built from simpler base learners, such as decision trees, their aggregated and often complex structure makes individual decision paths harder to follow, thereby reducing their transparency. At the other end of the spectrum, models such as deep NNs and support vector machines are typically regarded as having low interpretability. These models rely on highly abstract mathematical transformations and operate in high-dimensional feature spaces, making it difficult for humans to intuitively understand how predictions are derived. Consequently, the level of interpretability can be seen as a function of both the model's internal complexity and the ease with which its decision-making process can be understood and communicated [22, 192].

Taxonomy of XAI Approaches

Interpretable machine learning has led to the development of various taxonomies aimed at organizing the growing body of methods and evaluation approaches. These taxonomies serve to clarify the diverse goals, characteristics, and audiences of XAI techniques. Here, two influential taxonomies are presented: one categorizing XAI methods across multiple technical and practical dimensions, and another focused specifically on how interpretability should be evaluated in different research or deployment contexts.

Based on [234], XAI methods can be categorized along several dimensions:

1. Scope of explanation
   (a) Global: These methods aim to explain the overall behavior of the model across the entire input space. They help users understand general patterns, trends, and the logic that governs the model's predictions as a whole.
   (b) Local: These methods focus on explaining individual predictions or small subsets of the data. They are especially useful for understanding specific decisions and providing instance-level insights into why a model made a particular prediction.
2. Model specificity
   (a) Model-agnostic: These methods are independent of the underlying model architecture and can be applied to any black-box model. They typically rely on probing the model with inputs and analyzing outputs to construct explanations.
   (b) Model-specific: These methods are designed to work with particular classes of models, such as decision trees, linear models, or NNs. They leverage the internal structure and properties of the model to generate explanations.
3. Timing of explanation
   (a) Intrinsic (Ante hoc): In this case, interpretability is built into the model architecture from the outset. Examples include linear regression, decision trees, and rule-based models, which are transparent and interpretable by design.
   (b) Post hoc: These explanations are generated after the model has been trained. They aim to provide insights into the model's decision-making process without altering the model itself. Techniques such as LIME, SHAP, and saliency maps fall into this category.
4. Target audience
   (a) Experts: These are individuals with machine learning expertise, such as data scientists or engineers, who seek detailed, technical insights into the model's functioning and decision logic.
   (b) Lay users: These are non-technical end-users, such as consumers, patients, or citizens, who need simplified and intuitive explanations to understand how and why certain decisions affect them.
   (c) Domain experts: These users are professionals in specific fields, such as healthcare or law, who require explanations that align with their domain knowledge and support decision-making within their area of expertise.
5. Explanation form
   (a) Visual: Explanations are provided in the form of images, plots, or heatmaps (e.g., saliency maps or feature importance plots) that illustrate model behavior or highlight influential features.
   (b) Textual: Explanations are expressed in natural language or descriptive text, summarizing the factors that contributed to a prediction in a human-readable format.
   (c) Example-based: These methods use representative instances, such as prototypes or counterfactuals, to explain decisions by showing similar examples or contrasting outcomes based on small changes to the input.

(d) Mathematical/Formal: Explanations take the form of symbolic rules, equations, or formal logic that describe the decision process in precise mathematical terms.

6. Fidelity and completeness

   (a) Faithful: These explanations accurately reflect the model's true behavior and decision-making process. They aim to be consistent with how the model actually works internally.

   (b) Approximate: These explanations provide simplified or abstracted versions of the model's logic to improve human understanding, though they may not fully align with the exact model behavior.

7. Evaluation approach

   (a) Application-grounded: Evaluation is performed with real users in the context of real-world tasks. This approach offers high ecological validity and measures the utility of explanations in practical scenarios.

   (b) Human-grounded: This approach uses human subjects in simplified tasks to evaluate explanation quality, often through user studies or surveys, without requiring a full application context.

   (c) Functionally grounded: These evaluations do not involve humans and, instead, rely on formal metrics or proxy tasks, such as sparsity or accuracy of approximation, to assess interpretability.

The taxonomy highlights that interpretability is a multifaceted concept shaped by the goals of explanation, the nature of the model, and the needs of the intended audience. Importantly, they emphasize that not all explanation methods serve the same purpose, and that both method selection and evaluation should be context-sensitive. These frameworks provide a foundation for systematically understanding and comparing different approaches, as well as for aligning technical solutions with practical requirements in real-world applications.

2.7.1 Prominent Explanation Techniques

Permutation Feature Importance (PFI)

Permutation Feature Importance (PFI) quantifies the extent to which a model relies on a feature by measuring the change in predictive performance when that feature's values are randomly permuted, thereby breaking its association with the outcome [43, 98]. The permutation feature importance approach, based on [98], for a trained model f, data $(\mathbf{X}, \mathbf{y})$, and loss L, involves the following steps:

1. Estimating a baseline model error.

$$e_{\text{orig}} \;=\; \frac{1}{n}\sum_{i=1}^{n} L\big(y^{(i)},\, f(\mathbf{x}^{(i)})\big). \tag{2.39}$$

2. For feature j, a permuted dataset is constructed $\mathbf{X}_{\pi_j}$ by shuffling column j and computing the error based on the predictions of the permuted data:

$$e_{\text{perm},j} \;=\; \frac{1}{n}\sum_{i=1}^{n} L\big(y^{(i)},\, f(\mathbf{x}^{(i)}_{\pi_j})\big). \tag{2.40}$$

3. Calculation of the importance of feature j as a difference or a ratio:

$$FI_j^{(\Delta)} \;=\; e_{\text{perm},j} - e_{\text{orig}} \qquad \text{or} \qquad FI_j^{(\times)} \;=\; \frac{e_{\text{perm},j}}{e_{\text{orig}}}, \tag{2.41}$$

with repeated permutations and averaging often used to reduce variance.

Should permuting a feature j degrade model performance, then that model depends on the feature, and if performance is unchanged, then that feature contributes little to the trained model f on the evaluated data [98, 192].

PFI is highly valued for its simplicity and model-agnostic nature [192]. PFI also enables insight into the data itself as if there is a feature that the model uses for prediction but, in reality, is irrelevant, then PFI will still show, approximately zero importance for that feature [192]. Additionally, PFI takes into account all interactions with other features because permuting the feature not only breaks the main effect of the feature itself but also interaction effects with other features. Consequently, interaction contributions are counted for each interacting feature, so importance values usually do not equate to the overall drop in performance [192].

However, despite being a simple and model-agnostic measure, PFI has certain limitations. First, the measure is tied to model error and therefore performance—when robustness rather than performance impact is of interest, additional steps are needed, such as analysis of output variance [192]. Second, the feature importance provided is only an overall ranking, with no additional information regarding the direction of the effect on the prediction or interactions with other features [192]. Third, randomness is introduced by the shuffling procedure, so repeated permutations and averaging are recommended for stability at the cost of additional computation [192]. Fourth, issues arise concerning correlated features. When a feature is permuted, the correlation structure is broken and unrealistic instances can be generated, so biased importance can be obtained. In addition, if a correlated feature is added, the importance can be split between the pair so that each appears less important individually, which makes interpretation and prioritization more difficult [192].

Integrated Gradients

Integrated Gradients is a widely used gradient-based attribution method for interpreting deep neural networks, attributing a model's prediction to its input features in a mathematically principled way. Integrated Gradients is defined for a differentiable model f and an input $\mathbf{x}$, relative to a baseline input $\mathbf{x}'$, which is often a zero vector or neutral reference. This baseline is one of the key defining aspects of the approach, representing a state where the features provide no or neutral information. IG computes the feature attributions by integrating the gradients of f along the straight-line path from the baseline to the input, formally defined as:

$$\mathrm{IG}_i(\mathbf{x}) := (x_i - x_i') \int_{\alpha=0}^{1} \frac{\partial f(\mathbf{x}' + \alpha(\mathbf{x} - \mathbf{x}'))}{\partial x_i} \, d\alpha \tag{2.42}$$

In essence, for each feature i, the gradient of the model's output with respect to that feature is calculated at many smaller steps along the straight-line path from the baseline to the input, and these calculated gradients are then aggregated [248].

Integrated Gradients satisfies several desirable axioms, including Sensitivity and Implementation Invariance, making it theoretically grounded and consistent with intuitive expectations about feature importance [248]. The Sensitivity axiom stipulates that if a feature's change from a baseline $\mathbf{x}'$ causes a prediction change, it must recieve a non-zero attribute. Simple gradients can fail here due to "saturation." Integrated Gradients addresses this problem by using integration—by integrating along the path, the full effect of the feature is captured, not merely its local sensitivity.

The Implementation Invariance axiom specifies that if two models perform the exact same task, they should have identical explanations, regardless of how they're built internally. Since Integrated Gradients relies on a concept (gradients) that is invariant to implementation details, it can quite readily satisfy this axiom.

Due to the aforementioned properties, the Completeness property is fulfilled as a direct result of the design. This property guarantees that all feature attributions amount to exactly the total difference between the actual prediction and

the baseline prediction. This leads to fully accountable and easy-to-understand explanations.

$$\sum_{i=1}^{n} \text{Attribution}_i(\mathbf{x}, \mathbf{x}') = f(\mathbf{x}) - f(\mathbf{x}') \tag{2.43}$$

The approach's strengths include its somewhat model-agnostic nature (requiring a differentiable model), robust theoretical background, and efficient computation speed as a gradient-based method. However, despite these strengths, a significant problem was highlighted by [8]: these methods can be insensitive to either the model's parameters or the data distribution, and in the case of image recognition, often yield results resembling simple edge detection. For tabular data, a primary difficulty lies in specifying a meaningful baseline. Another issue for tabular data is that the assumption of a straight-line path between a baseline and an input can generate implausible intermediate data points, especially with complex feature relationships or categorical features. This can lead to inaccurate gradients along the path, meaning the final attribution is less reflective of the feature's true contribution [192].

Shapley Values and SHAP

The idea behind Shapley Values borrows concepts from cooperative game theory and attempts to explain the prediction by assuming that each feature value of the instance is a "player" in a game where the payout is the prediction itself [239]. Formally expressed, in a setting where a group of n players $N = \{1, \ldots, n\}$ cooperatively contribute to a total value represented by a characteristic function $v : 2^N \to \mathbb{R}$, the Shapley value ϕ_i for a player i is defined as the average marginal contribution of i across all possible subsets $S \subseteq N \setminus \{i\}$:

$$\phi_i(v) = \sum_{S \subseteq N \setminus \{i\}} \frac{|S|!(|N| - |S| - 1)!}{|N|!} \left[v(S \cup \{i\}) - v(S) \right].$$

The Shapley value is uniquely determined by several natural axioms:

1. Efficiency: The total allocated value equals the full coalition value. That is, the Shapley values ϕ_i equate to the total value of the grand coalition (excluding the empty set), ensuring additive completeness.

2. Monotonicity: If a player's marginal contributions in one game are always at least as large as in another, its Shapley value should not be smaller in the former.

3. Symmetry: If two players contribute equally to all coalitions, they receive equal Shapley values.

4. Missingness: A player that adds no marginal value to any coalition receives a Shapley value of zero.

These axioms collectively define the Shapley value and justify its use as a principled method for feature attribution in explainable AI [239]. A visual represesntaion of the defining terms related to the Shapley value are shown in Figure 2.8.

Lundberg and Lee [176] adapted Shapley values for use in machine learning by interpreting each input feature as a player and the model's output as the value function. SHAP (SHapley Additive exPlanations) explains a prediction $f(x)$ by attributing contributions to each feature, such that:

$$g(\mathbf{z}') = \phi_0 + \sum_{j=1}^{M} \phi_j z'_j \tag{2.44}$$

where g is the explanation model, $\mathbf{z}' = (z'_1, \ldots, z'_M)^T \in \{0, 1\}^M$ is the coalition vector, M is the maximum coalition size, and $\phi_j \in \mathbb{R}$ is the feature attribution

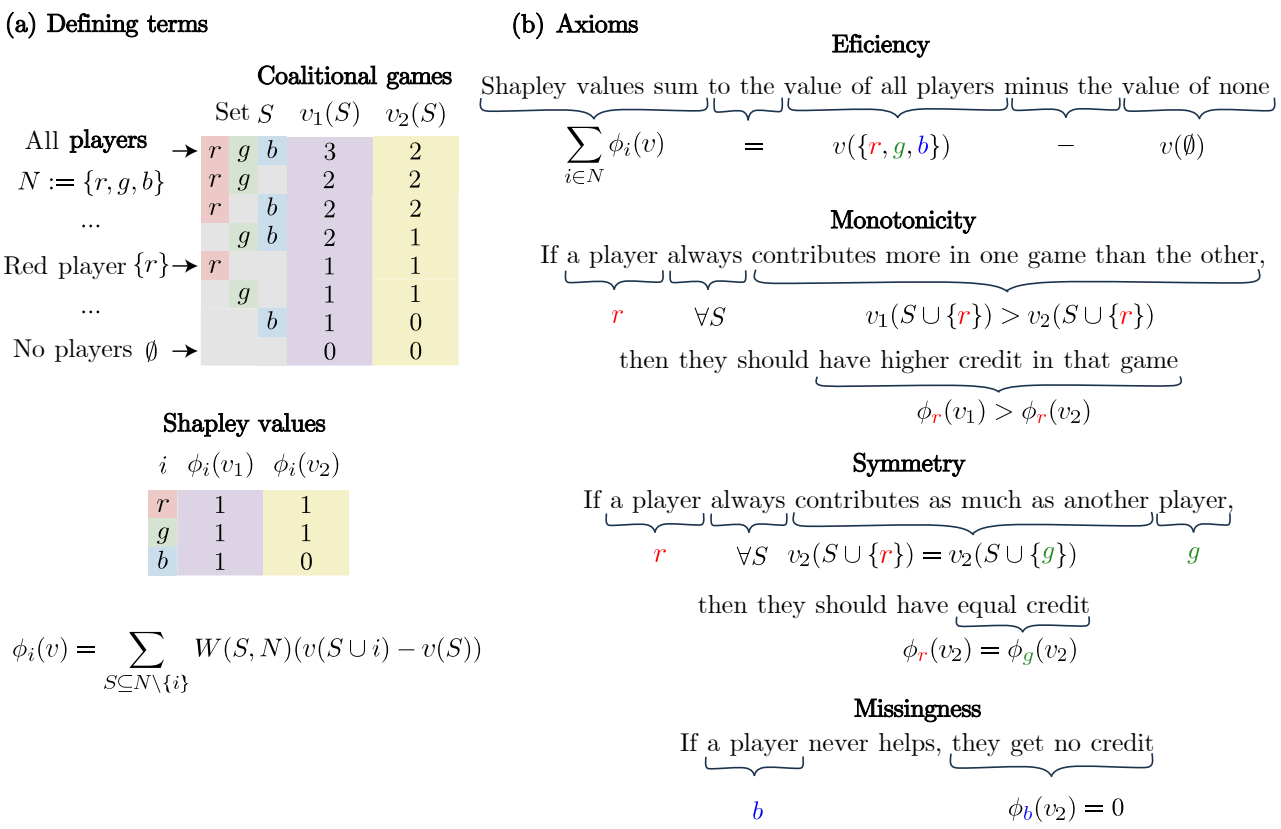

Figure 2.8: Defining terms related to Shapley values, adapted from [56]

for feature j. Here, the term coalition vector is taken from [192] and replaces the term 'simplified features' that was used in the original paper.

Based on its use of Shapley values, SHAP uniquely satisfies the axioms of Efficiency, Symmetry, Dummy and Additivity. However, in their paper [176], the following properties are described:

1. Local Accuracy
   This property ensures that the sum of feature attributions equals the model output for a given instance. Let $\hat{f}(\mathbf{x})$ be the original model output and $g(\mathbf{x}')$ be the explanation model using a coalition vector $\mathbf{x}'$, then:

$$\hat{f}(\mathbf{x}) = g(\mathbf{x}') = \phi_0 + \sum_{j=1}^{M} \phi_j x'_j \tag{2.45}$$

   If all $x'_j = 1$, this is reduced to:

$$\hat{f}(\mathbf{x}) = \phi_0 + \sum_{j=1}^{M} \phi_j = \mathbb{E}[\hat{f}(X)] + \sum_{j=1}^{M} \phi_j \tag{2.46}$$

   where $\phi_0 = \mathbb{E}[\hat{f}(X)]$ is the expected model output over the background distribution.

2. Missingness
   This property states that if a feature is absent from the input, i.e., excluded from the coalition, it should have zero attribution:

$$x'_j = 0 \Rightarrow \phi_j = 0 \tag{2.47}$$

   This ensures that only features present in the coalition receive non-zero SHAP values, which is particularly useful for handling missing data or conditional feature masking.

3. Consistency
   If a model changes such that the marginal contribution of a feature increases, the SHAP value assigned to that feature should not decrease. Let $\hat{f}$ and $\hat{f}'$ be two models. For all binary input coalitions $\mathbf{z}' \in \{0,1\}^M$ and any subset $\mathbf{z}'_{-j}$, if:

$$\hat{f}_{\mathbf{x}}(\mathbf{z}') - \hat{f}_{\mathbf{x}}(\mathbf{z}'_{-j}) \geq \hat{f}'_{\mathbf{x}}(\mathbf{z}') - \hat{f}'_{\mathbf{x}}(\mathbf{z}'_{-j}) \tag{2.48}$$

   then the SHAP values must satisfy:

$$\phi_j(\hat{f}', \mathbf{x}) \geq \phi_j(\hat{f}, \mathbf{x}) \tag{2.49}$$

   This guarantees that SHAP respects increasing feature importance across model versions.

SHAP's main strength lies in its solid theoretical foundation. The attributions it produces are grounded in cooperative game theory, yielding consistent and additive explanations. Unlike other methods, SHAP ensures that the sum of the attributions matches the actual model output, allowing both local and global interpretability . Moreover, SHAP values can be aggregated across multiple instances to produce reliable global feature importance scores [175, 192].

However, the exact computation of Shapley values is exponential regarding the number of features, which poses a computational challenge. KernelSHAP approximates these values via sampling and weighted linear regression, making it model-agnostic but computationally intensive [176]. TreeSHAP [175], in contrast, is optimized for decision tree models and computes exact values in polynomial time, drastically improving efficiency.

Aspect	**SHAP**	**Permutation Feature Importance (PFI)**	**Integrated Gradients (IG)**
Explanation Type	Additive feature attributions based on Shapley values	Performance change after randomly permuting a feature's values to break its association with the target	Gradient-based cumulative attribution from a baseline to the input
Scope	Local and Global	Global (dataset-level ranking of feature reliance)	Local
Compatibility	Model-agnostic and model-specific	Model-agnostic (works with any predictive model and metric)	Model-specific (differentiable models)
Output Format	Feature contributions that sum to the prediction	Importance score as difference or ratio of error/score before vs. after permutation	Attribution vector showing each feature's contribution to the prediction
Key Assumptions	Cooperative game theory axioms	Permutation disrupts information carried by the feature; chosen metric reflects task performance	Input space is differentiable; baseline reference is meaningful
Advantages	Fair, theoretically grounded; consistent explanations; applicable to complex models	Simple, fast, no retraining; model-agnostic; captures main effects and interactions; produces clear global rankings	Computationally efficient; captures gradient sensitivity; faithful for deep models
Limitations	Computationally intensive; can be slow for large models	Sensitive to feature correlation (bias/splitting); randomness from permutation; global and loss-based (no directionality)	Sensitive to choice of baseline; only applies to differentiable models
Use Cases	Feature attribution; auditing decisions; both local and global insight	Global importance ranking; model monitoring and drift checks; sanity checks and feature selection	Interpreting deep neural networks; attribution in vision, NLP, or tabular domains

Table 2.1: Comparison of Popular Model Explanation Techniques

Challenges in Explainable and Interpretable AI

Many researchers have explored the problems and open questions in Interpretable and Explainable Machine Learning, indicating that a strong scientific basis for interpretability is needed [83], and several practical and technical challenges have been described in detail [194]. These often include concerns about how accurate and trustworthy explanations are, how clearly they reflect the model's reasoning, and how useful they are in real-world situations. Some of the most important and widely discussed challenges include the following:

1. Statistical uncertainty: Many XAI methods do not quantify the uncertainty of their explanations, which can mislead practitioners when interpreting noisy or sparse data.
2. Causal interpretation: Most XAI techniques reflect correlation, not causation, limiting their use in domains where actionable or causal insights are needed.
3. Feature dependence: Techniques that rely on independent feature assumptions (e.g., permutation methods) can break when features are correlated, leading to misleading attributions.
4. Lack of definition: Interpretability remains vaguely defined, making evaluation inconsistent and comparisons across methods difficult.
5. Evaluation metrics: There is no agreed-upon ground truth for explanations, so evaluation often relies on qualitative or subjective human-centered metrics.
6. Local vs. global explanations: Bridging the gap between localized instance-level explanations and broader global understanding remains challenging.
7. Scalability and complexity: Many XAI methods are computationally intensive, limiting their application to high-dimensional or real-time scenarios.
8. Human factors: Interpretations must be understandable by humans, yet what counts as "understandable" varies greatly depending on the user's domain knowledge and cognitive abilities.

Explainable AI is crucial for the responsible deployment of AI systems. By providing insights into model decisions, XAI fosters trust, ensures fairness, and facilitates compliance with ethical and legal standards. As AI continues to permeate various aspects of society, the development and adoption of XAI methods will be essential in aligning AI systems with human values and expectations.

2.8 Summary of this Chapter

This chapter provides the theoretical basis for the thesis by outlining the fundamental concepts and methods that guide the development of the proposed approach. It begins with a structured overview of Multi-Criteria Decision-Making and its subfields, with a particular focus on Multi-Attribute Decision-Making, which plays a central role in the evaluation of alternatives under multiple criteria. This is followed by the introduction of Robust Decision-Making, which highlights strategies for handling uncertainty and variability in complex decision contexts. The discussion then turns to core principles of machine learning, encompassing supervised, unsupervised, and reinforcement learning, while also touching on Learning to Rank and weak supervision, both of which are directly relevant to modeling preferences and learning from imperfect data. Building on this foundation, specific learning methods, such as Gradient Boosting, Neural Networks, and Learning Classifier Systems, are introduced and briefly explained. Finally, the growing importance of explainable and interpretable AI is addressed, underlining why transparency is essential for decision support in safety-critical domains.

3 Related Work

This chapter reviews the relevant literature on decision-making in emergency management. It begins by examining the role of MCDM methods in emergency contexts, highlighting its proven utility and identifying key research gaps. This is followed by an exploration of how AI has been applied to support decision-making during emergencies, additionally addressing the important point of of ethical considerations when deploying AI in high-stakes environments. The chapter then examines how MCDM and AI have been integrated, showcasing emerging hybrid approaches. Finally, decision-support systems and mental models used in the field of aviation are presented, offering insights into how critical decisions are made under pressure in real-world, safety-critical domains.

3.1 MCDM in Emergency Management

3.1.1 Overview of Emergency Management and MCDM

Emergency management encompasses the coordinated activities of mitigation, preparedness, response, and recovery, which aim to reduce the impact of sudden disasters on people, property, and the environment. As modern societies have grown more interconnected and technologically complex, the types of risks encountered have expanded correspondingly. These not only include natural hazards but also industrial accidents, public health emergencies, and operational disruptions in domains such as aviation. The latter category includes tasks like alternate-airport selection, where decisions must be made rapidly under uncertainty and time pressure [161].

A substantial portion of this subsection draws on the comprehensive survey by Li et al. [161], which provides an excellent overview of how emergency management is conceptualized and how MCDM methods have been incorporated into its various phases. Their work served as both an intellectual foundation for understanding the structure of emergency management and as a roadmap of the wider literature discussed throughout this chapter. According to their synthesis, the central objectives of emergency management can be summarized into four overarching goals:

1. Protect human life through preparedness measures and timely, effective responses during disaster events.

2. Minimize damage to property and infrastructure via proactive mitigation strategies and coordinated emergency interventions.

3. Safeguard the environment by preventing or reducing long-term ecological harm and enabling environmentally responsible recovery.

4. Restore social order by facilitating post-disaster recovery, strengthening community resilience, and supporting the return to normal societal functioning.

3.1.2 Applications of MCDM in Emergency Scenarios

MCDM methods have a rich history of application in emergency management, spanning all major phases: mitigation, preparedness, response, and recovery. Several studies have emphasized the importance of understanding key success factors in managing emergencies effectively. For example, Fakhry et al. [93] used a neutrosophic DEMATEL approach to handle uncertainty in expert evaluations and identified "Preparedness and Planning" as the most critical factor among six main dimensions. This need for strategic foresight is echoed in studies like Hsu et al. [127], who used modified DEMATEL, Dombi aggregation, and VIKOR to assess airport disaster resilience in Taiwan, confirming that pre-disaster planning was the most influential factor.

Logistics and infrastructure planing

Logistics and infrastructure location planning is another key area where MCDM plays a crucial role. Feng et al. [95] and Xu et al. [284] applied GIS-based MCDM models to optimize the placement of emergency logistics centers, factoring in access to populations, infrastructure robustness, and risk exposure. Liu [170] extended this idea with a multi-hazard assessment framework for post-earthquake emergency medical facilities in China, while Sirbiladze et al. [241] developed a two-stage fuzzy MADM model to locate temporary logistics hubs. Eelagh and Abbaspour [85] integrated CRITIC and TOPSIS to optimize the spatial distribution of post-earthquake shelters, emphasizing that effective coverage is more important than quantity alone.

Public health emergencies

In the context of public health emergencies, particularly during the COVID-19 pandemic, numerous studies adopted MCDM to prioritize medical interventions and patient care. Abdulkareem et al. [6] created the Multidimensional Examination Framework (MEF), combining CRITIC and VIKOR to rank patients based on clinical data. Alsalem et al. [18] designed a real-time triage system using AHP and VIKOR to allocate mesenchymal stem cell therapies to critical patients. Ahmad et al. [12] addressed uncertainty in COVID-19 diagnostics using TOPSIS with non-linear Diophantine fuzzy numbers. Mao et al. [180] incorporated cumulative prospect theory into MADM to account for psychological behavior in epidemic response, while Davodabadi et al. [66] combined fuzzy MCDM with simulation to prioritize ICU admissions.

Healthcare system performance

Healthcare system performance and preparedness were further explored by Ortiz-Barrios and Alfaro-Saiz [205], who developed a hybrid model integrating FAHP, FDEMATEL, and TOPSIS to assess emergency department performance. Pegoraro et al. [211] employed DEMATEL and PROMETHEE II to identify operational improvements. Aminjarahi et al. [19] used Entropy, VIKOR, and SAW to rank lean techniques in emergency departments based on feedback from physicians and nurses.

Natural hazard risk assessment

Natural hazard risk assessment is another major application of MCDM. Pathan et al. [210] and Sun et al. [247] evaluated flood risks in India and China, respectively, using AHP, TOPSIS, and VIKOR. Hajilo et al. [113] assessed seismic risk in rural Iran using GIS and VIKOR, while Zhang et al. [291] applied entropy-weighted 3D-TOPSIS to assess flood defense capacity across Chinese regions. Shafiei Shiva et al. [240] used TOPSIS to map urban heatwave risks in Arizona, revealing unequal vulnerability.

Resilience assessment

Broader resilience assessment has also seen innovative use of MCDM. Wang et al. [271] studied unsafe behaviors in intelligent mines using fuzzy DEMATEL and MMDE, highlighting the role of leadership and regulation. Caylor and Hammell [53] applied AHP-TOPSIS to identify vulnerable counties during COVID-19 using infection and social data. Nagy et al. [198] evaluated emergency suppliers under uncertainty using four different MCDM models, confirming robustness across methods. Wang et al. [268] introduced a WSR-

based framework for emergency medical facility siting by separating criteria into physical, systemic, and human domains.

Strategic decision-making in emergencies

MCDM has also informed strategic decision-making tools. Chen et al. [58] developed a hybrid GRA-CBR approach for real-time, adaptive emergency decisions. Ali et al. [16] introduced a three-way decision model using fuzzy soft dominance to manage uncertainty in complex scenarios. Pagano et al. [207] used AHP to design a DSS for drinking water emergencies that included expert and societal input. Otay and Jaller [206] applied IVIFS to assess national disaster readiness.

Reconstruction

In the reconstruction phase, Mohammadnazari et al. [190] compared four MCDM methods (TOPSIS, ELECTRE III, VIKOR, and PROMETHEE) to prioritize post-disaster recovery projects, validating their findings using ANN models. Bait et al. [27] used AHP-TOPSIS to help companies identify suitable African nations for post-COVID investment, reflecting shifts in logistics and infrastructure needs. Liu et al. [169] used a two-stage fuzzy TOPSIS and optimization framework to plan emergency material allocation during chemical spills.

MCDM for emergency management methodology improvements

Finally, several contributions aimed to improve MCDM methodology itself. Jiang et al. [132] combined Pythagorean fuzzy sets with the Chebyshev distance in TOPSIS to manage uncertainty in disaster planning. Zhan et al. [289] proposed a PF-TOPSIS approach built on Pythagorean fuzzy rough sets to support decisions in unconventional emergencies. Abdel-Basset et al. [5] demonstrated the power of plithogenic sets in combining AHP, TOPSIS, and VIKOR to evaluate smart disaster response systems.

Albarracín Zambrano and Villalta Jadan [287] further extended the field into maritime safety by using MCDM with information fusion to assess risks in autonomous ship navigation. Their work highlights the growing importance of MCDM in emerging risk domains where data uncertainty, technical complexity, and system integration are key challenges.

In summary, MCDM has proven to be a highly adaptable and insightful framework across the extremely varied field of emergency management. Its ability to combine diverse information, accommodate uncertainty, and provide clear decision support makes it an indispensable tool for researchers, practitioners, and policymakers facing the increasing complexity of modern emergencies.

3.1.3 Research Gaps and Challenges

Much of the discussion of methodological limitations in this section is based, again, on the excelent survey paper by Liu et al. [161], which provides an excellent analysis of gaps within MCDM for emergency management.

Static Nature

Although MCDM methods have been increasingly applied in emergency management, several significant research gaps remain. First, most current uncertain MCDM methods are developed under static environments and do not effectively consider temporal continuity or historical decision data. As emergencies evolve dynamically over time, decision models should incorporate temporal aspects, such as the decay of information reliability, which remains largely unaddressed [161]. Furthermore, many existing approaches treat decision attributes independently, overlooking the interactions between attributes that can significantly influence decision outcomes [161]. In the case of a diversion to an alternate airport, many of the key factors evolve over time and may depend on one another. For example, changing wind and weather conditions can affect fuel consumption and aircraft performance, which in turn may alter the prioritization of available diversion options.

Heterogeneity Challenges

A fundamental challenge in emergency decision-making is the inherent heterogeneity of information. Decision-makers must process diverse data types, including numerical measurements, categorical classifications, and qualitative descriptions. Current MCDM approaches typically address this heterogeneity

either through indirect methods, which convert all data to a common numerical format at the cost of information loss, or direct methods which rely on similarity measures while potentially neglecting the semantic meaning of attributes. This becomes particularly problematic in emergency contexts in which the interpretation of heterogeneous data often depends on contextual relationships that can be lost in these transformations [161]. In an alternate-airport selection scenario, pilots must interpret heterogeneous data, such as numerical fuel estimates, categorical airport operational statuses, and qualitative Air Traffic Control (ATc) reports describing expected delays. Transforming these diverse data types into a suitable decision-making format while preserving their contextual meaning is challenging.

Dynamic Information Processing

Emergency situations are characterized by dynamic information environments in which data evolves continuously. While some MCDM methods can accommodate changing information, they often fail to address two critical aspects: the heterogeneous nature of temporal changes across different data types, and the evolution of attribute interactions over time. For instance, in wildfire management, the relationship between meteorological factors and fire behavior changes as the incident progresses, requiring dynamic reassessment of both individual attributes and their interdependencies. This necessitates methods that can adapt not only to new information but also to evolving relationships between decision factors [161]. For instance, in alternate-airport selection, different types of information evolve at different temporal scales: wind conditions may change minute-by-minute, fuel state evolves deterministically, and runway closures can occur abruptly without notice. These heterogeneous temporal dynamics alter how criteria such as weather, aircraft performance, and operational constraints interact, requiring decision methods that update both attribute values and their interdependencies over time.

Attribute Interaction Modeling

The modeling of attribute interactions presents another significant challenge. While advanced mathematical frameworks such as Choquet and Sugeno integrals have been proposed to capture attribute interdependencies, most MCDM implementations still rely on the simplifying assumption of attribute independence [161]. This limitation is particularly problematic in emergency contexts where factors often exhibit complex, time-varying interactions. For example, in the alternate-airport selection problem, the interaction between weather conditions and aircraft performance constraints is highly non-linear. A shift in wind direction may simultaneously affect the required landing distance, fuel margins, and the feasibility of multiple airports, meaning that it is difficult to evaluate the importance of one attribute independently of the others.

Emergency management from the outside

One important aspect to consider is that the MCDM methods proposed and applied in emergency management have typically been developed from the perspective of external response organizations. Although these entities are responsible for coordinating disaster relief and emergency response, they are seldom embedded within the emergency scenario itself to the point of directly experiencing the outcomes of the decisions made. In the alternate airport selection problem, the pilots (who are the ultimate decision-makers) are directly affected by the decisions made and their outcomes, especially in situations where an unsuitable diversion choice could have severe consequences.

3.2 AI in Emergency Management

Just as MCDM has been used to support decision-makers in emergency contexts, methods from the field of AI have also been recognized for their potential to assist in this endeavor. AI techniques have been employed across various phases of emergency management, including prevention, preparedness, response, and recovery [173, 138, 286].

Natural hazard risk assessment

Given AI methods' ability to learn complex patterns from data, they are suitable for predicting emergencies and supporting decision-makers—not only in

responding effectively but also in identifying specific vulnerabilities [150]. For instance, Moustra et al. [196] utilized neural networks with diverse input data to forecast earthquake magnitudes. Similarly, Reyes et al. [217] used neural networks to estimate both the occurrence and recurrence probabilities of seismic events. Turning to hydrological hazards, Sahay et al. [230] developed a machine learning model to forecast monsoon river flows one day in advance, which posed challenges due to their erratic and irregular behavior. Transitioning to wildfire prediction, Deparday et al. [74] integrated weather data and image classification techniques to significantly improve forest fire predictions. Finally, addressing human-induced emergencies, Kim et al. [143] proposed a neural network-based time-series forecasting model to predict storm surges, enabling more timely evacuation decisions.

Human recognition in emergency situations

Human detection and rescue in emergencies often involve time-critical decisions vital for saving lives. Lygouras et al. [177] demonstrated how deep learning and computer vision can be employed to detect open-water swimmers. Dong et al. [82] applied machine learning to thermal imagery for human detection, enabling effective aerial search operations. During the COVID-19 pandemic, Nguyen et al. [200] used AI methods for crowd monitoring, supporting both citizens and authorities in maintaining social-distancing protocols.

Comparison with MCDM methods

While both AI and MCDM methods have distinct strengths in emergency management, they tend to serve complementary roles. AI methods primarily support the information phase, i.e., providing decision-makers with data-driven insights, forecasts, and classifications, such as estimating the likelihood of an event or identifying high-risk locations. MCDM, by contrast, aides in the evaluation and choice phase, helping structure and weigh alternatives based on multiple criteria, ultimately guiding the selection of the best course of action. In high-stakes emergency scenarios, where both human lives and critical resources are at risk, this combination is especially valuable. AI enhances situational awareness, while MCDM facilitates transparent and justifiable decision-making.

3.2.1 Challenges of using AI in Emergencies

Lack of Data

One of the more immediate and glaring problems in using AI for emergencies is the general lack of data. Emergencies represent rare events that do not occur frequently, and as a result, the amount of available data is often quite limited. This presents two key challenges: a scarcity of training data and limited opportunities for end-users to apply and evaluate these models in real-world settings [150]. One way to counteract this is through the use of synthetic data and data augmentation techniques. In particular, the growing use of digital twins, i.e., virtual replicas of physical environments that simulate interactions between people, places, and devices, offers a promising solution [186, 103]. By replicating real-life conditions in a virtual space, these systems enable the development and testing of ML algorithms in scenarios that closely resemble actual emergencies. This not only provides large volumes of synthetic data for training and validation but also allows emergency responders and safety professionals to observe how ML systems behave and assess their practical utility [294, 150].

Trustworthiness, explainability, and interpretability

Having discussed the importance of explainable and interpretable models in the previous chapter, it is important to emphasize that, in emergency situations, decisions made with the help of such systems must meet even higher standards and more stringent requirements. Given the high-stakes nature of emergencies and the potential consequences of poor decisions, it is vital that all involved actors have a clear understanding of the algorithm's capabilities as well as the rationale behind its estimations or recommendations, with trust and confidence in the systems and the resulting decision being particularly important.

Ethical considerations of using AI in emergencies

The use of AI in emergency management presents significant ethical considerations that must be addressed alongside its technical capabilities. The paper by Jaideep Visave [266] focused specifically on the ethical implications of applying these new and promising techniques in emergency situations. As AI systems become increasingly involved in decision-making processes, it must be

clear who is responsible for the consequences of AI-guided decisions. Furthermore, concerns about bias and fairness are particularly important in emergency decision-making, where time is limited and historical data may reflect systemic inequalities. For instance, AI models might inadvertently prioritize well-resourced regions over underserved communities, thereby reinforcing existing disparities that are entrenched in the data used to train the models. Privacy and data protection also play a central role, as real-time emergency responses often require access to sensitive personal information, such as health data, geolocation, or surveillance input. These requirements must be carefully balanced against individuals' rights and legal safeguards. Lastly, transparency and public trust are essential for the legitimacy of AI-based interventions, yet public engagement in the development and deployment of such systems is often lacking. Ethical deployment demands that not only are technical requirements satisfied but also legal ones, ensuring that decisions are explainable, affected populations are informed, and AI tools reflect shared values rather than merely optimizing for efficiency.

3.3 AI and MCDM

Incorporating AI methods into MCDM frameworks is increasingly common in the literature. The ability of modern machine learning models to estimate feature importance or generate option scores aligns naturally with the fundamental MCDM tasks of ranking and selecting alternatives.

Criteria Extraction

In traditional MCDM approaches, decision-makers determine the relevant factors based on domain expertise or brainstorming, which can be subjective, incomplete, or impractical when the problem space is large and data-rich. For example, the study by Darko and Liang [64] ascertained that big unstructured data, such as online reviews, pose a challenge to extract meaningful evaluation criteria. Modern machine learning techniques have been shown to help automate the extraction of relevant criteria from unstructured or high-dimensional data [163]. Machine learning methods geared toward dimensionality reduction and clustering have become common tools for automatically extracting relevant criteria. Among these, Principal Component Analysis (PCA) is one of the most frequently used. This method reduces complex, high-dimensional data into a smaller set of principal components while retaining the essential information. Leveraging this, Lim et al. [165] used PCA to identify key elements in vehicle exhaust datasets, while Boodhun et al. [39] employed it for feature selection in life insurance risk prediction. The study by Ye et al. [285] developed a framework that incorporated PCA, fuzzy sets, and TOPSIS to refine evaluation systems, and Jomthanachai et al. [134] applied PCA to determine the most informative economic indicators for forecasting national logistics performance. In a similar vein, the paper by Stevic et al. [245] integrated PCA with MCDM to assess transport company efficiency by generating new principal components for input into efficiency calculations, while in the paper by Modibbo et al. [188], PCA was used to identify the top criteria for pharmaceutical supplier selection.

Beyond PCA, other advanced ML methods, including deep learning, have also been adapted for criteria extraction. Sadhu et al. [229] employed an NN to uncover non-linear relationships among frying criteria and sensory evaluation factors, integrating AHP and a multi-objective genetic algorithm for optimal parameter tuning. From fryers to the restaurant industry, Han et al. [114] applied correlation-based feature subset selection to identify the attributes most closely linked to restaurant sales performance. Eshkevari et al. [87] leveraged the Dawid–Skene algorithm to combine user sentiment at the aspect level, then used the Best–Worst Method for hotel ranking, treating aspects as decision criteria. Wang et al. [270] combined deep learning with MCDM techniques to help organizations improve decision-making by extracting more relevant criteria from data-rich business intelligence systems. Finally, in gaming, the paper by Ince [130] demonstrated that dynamic player preferences in gaming envi-

ronments can be modeled using a BiLSTM combined with fuzzy AHP, showing how adaptive hybrid systems can capture changing preferences based on player behavior.

Criteria Interaction

Many real-world MCDM problems involve criteria that interact in non-linear, non-additive ways. Classical approaches often oversimplify this by assuming that criteria are independent and can be aggregated linearly—a common criticism of traditional MCDM methods. By incorporating ML models, these complex interdependencies can be better captured and modeled [163].

One widely recognized tool for capturing interdependencies is the Choquet integral. The Choquet integral is a widely used aggregation operator for modeling interactions among criteria in MCDM problems, especially when criteria are interdependent. Unlike simple additive models, the Choquet integral employs a fuzzy measure to assign importance not only to individual criteria but also to all possible subsets, allowing it to capture synergy or redundancy effects. Mathematically, for a vector of criteria evaluations $x = (x_1, x_2, \ldots, x_n)$, the discrete Choquet integral is defined as:

$$C_g(x) = \sum_{i=1}^{n} \left[x_{(i)} - x_{(i-1)}\right] \cdot g(A_{(i)}), \tag{3.1}$$

where $x_{(1)} \leq x_{(2)} \leq \cdots \leq x_{(n)}$ are the input scores sorted in ascending order, $x_{(0)} = 0$, and $A_{(i)}$ is the subset of criteria with the top i largest values. The function $g(\cdot)$ represents the fuzzy measure, which defines the importance of each subset of criteria. By computing differences between consecutive ordered scores and weighting them by the fuzzy measure, the Choquet integral generalizes linear aggregation and better reflects real-world dependencies among criteria [59].

However, as Moghtadernejad et al. [189] pointed out, manually assigning importance weights for the Choquet integral becomes highly complex when dealing with a large number of criteria. Mohebbi et al. [191] addressed this challenge by combining a fuzzy neural network with the Choquet integral, enabling the automated handling of interaction effects in MCDM processes. Building on this idea, Moghtadernejad et al. [189] further proposed integrating PCA and a neural network to estimate the necessary weights for the Choquet integral, demonstrating reliable performance when ranking optimal building façade systems.

In addition to these indirect methods, some machine learning approaches directly address criteria dependency by transforming the data. For example, Liu et al. [168] introduced the IVIF-PAC model for large-group decision-making tasks with many criteria, using PCA to reduce dimensionality and produce more independent principal components. Similarly, Liu et al. [171] found strong interdependencies when evaluating navigation signal performance and applied PCA with fuzzy clustering to decorrelate and restructure the underlying criteria set. Guo et al. [112] proposed a neural network-based MCDA model that uses an NN to capture higher-order interactions and non-linear relationships between attributes. Zhong et al. [295] took this further by designing a multi-criterion-based reward quantification approach and introducing a multi-criterion Q-learning algorithm to manage complex interaction effects. Despite these advances, criteria interaction remains a significant modeling challenge. Cai et al. [50] noted that deep neural networks occasionally underperform on structured data precisely because they cannot adequately capture non-additive interaction patterns among criteria. To overcome this, they incorporated the Kano model to better handle non-linear and non-compensatory interactions within the decision process.

Parameter Determination

A central challenge in MCDM is determining model parameters such as criteria weights, thresholds, and utility functions. Conventionally, these are elicited through surveys or expert pairwise comparisons, which are time-consuming and can become inconsistent under cognitive stress or uncertainty. Machine

learning models can be employed to derive these parameters automatically by training on historical data [163].

For example, in the paper by Arabameri et al. [21], the COPRAS model was combined with machine learning algorithms such as Random Forest, Boosted Regression Trees, and logistic regression to specifically derive criteria weights. Dugger et al. [84] argued that traditional elicitation methods can be inefficient and subjective, so PCA was applied to reduce dimensionality and assign weights based on the proportion of variance explained by each principal component. The approach demonstrated how PCA can effectively generate a rank order for U.S. Army pilots by evaluating attributes like emotional intelligence, safety attitudes, and safety citizenship. Xu et al. [283] also integrated D numbers with PCA to identify and weight secondary indicators for offshore wind turbine selection, while in another paper, Marzouk and Hassan [184] developed a hybrid PROAFTN-K-means-genetic algorithm framework to determine indicator weights for modeling museum evacuation and visitor proximity simulations.

Aside from learning from historical data, preference learning has been applied to improve how decision rules and weights are derived in MCDM. Preference learning is a branch of machine learning focused on modeling and predicting human or group preferences. In their work, Birlutiu et al. [36] presented an active preference learning framework that intelligently selects queries from available preference data, reducing respondent burden and computation time. Zhen et al. [293] proposed a semi-parametric preference learning model that combines the strengths of both parametric and non-parametric techniques to capture more flexible preference structures. Zhao et al. [292] introduced an online graph-regularized user preference learning (OGRPL) framework for social recommendations, integrating collaborative user–item relationships and content information into a unified process. Similarly, in their paper, Sun et al. [246] developed a Long- and Short-Term Preference Modeling approach to point-of-interest recommendations, enabling the model to capture both recent and stable user preferences more accurately.

Integrated Decision Support Solutions

Beyond isolated tasks such as weighting or ranking, the literature increasingly reports fully integrated systems that combine machine learning and MCDM into robust, explainable decision support systems. Such hybrid DSS frameworks typically embed ML modules to extract, model, and forecast relevant decision data, while MCDM modules aggregate this information in a structured and transparent way for ranking, selection, or policy recommendations [163]. For instance, Kuznetsova et al. [149] designed a reinforcement learning framework to optimize battery scheduling decisions for individual consumers, treating this as an MCDM problem that balances criteria such as utilization rates and demand forecasts. Their system processed historical battery usage and real-time conditions to output battery control actions.

In the work by Ferreiro-Cabello et al. [96] tackled housing slab design as an MCDM problem, developing a DSS that combines heuristic algorithms and deep learning to generate a database of solutions. This system also integrated Pareto optimality techniques and graphical tools to help decision-makers analyze trade-offs more effectively. He et al. [120] created a deep reinforcement learning-based DSS for textile chemical process optimization, treating the process as a Markov decision process and combining Random Forests with the AHP method. Their proposed approach utilized the DQN algorithm to solve the sequential decision problem. Similarly, dos Santos et al. [233] developed an ANN-powered DSS to help users choose between electric and conventional fuel vehicles, using input criteria such as purchase price and driving range.

Of somewhat greater relevance

While the broader literature provides a comprehensive overview of how AI and MCDM have been combined, certain papers stand out for their unique contributions or particularly relevant methodologies. These works deserve special attention and are briefly highlighted below.

Abdulla and Baryannis [7] proposed a hybrid supplier selection framework that combined ML and MCDM with the explicit goal of retaining explainability and trust for stakeholders. Their motivation stemmed from the fact that classical MCDM methods, such as AHP or TOPSIS, are familiar and transparent but struggle with scalability when the number of criteria or suppliers is large, whereas modern machine learning can handle large datasets but often lacks transparency. To bridge this gap, the authors maintained the centrality of MCDM for the final supplier ranking and selection, while using machine learning primarily as a complexity reduction mechanism.

The framework operates in three phases. First, relevant data is collected and preprocessed, merging multiple datasets and performing feature cleaning and transformation to ensure quality. Second, an interpretable machine learning model, such as a decision tree, is used to identify the most important features for supplier selection or to narrow down the number of potential suppliers. For example, feature importance is calculated using Gini impurity reduction:

$$H(Q_m) = \sum_k p_{mk}(1 - p_{mk}) \tag{3.2}$$

where Q_m represents the data at node m of the decision tree, and p_{mk} is the proportion of samples belonging to class k at that node.

After the feature importance is calculated, only the top features are retained. In their case study, this reduced the number of criteria from 26 to 8, making the problem manageable for MCDM.

The third phase applies an MCDM method, via the AHP method. The calculated feature importance values are used as criteria weights. For each supplier, their scores on each criterion are normalized, weighted, and summed to obtain an overall score:

$$S_i = \sum_{j=1}^{n} w_j \cdot x_{ij} \tag{3.3}$$

where S_i is the final score for supplier i, w_j is the weight of criterion j, and x_{ij} is the normalized score of supplier i on criterion j. The supplier with the highest score is then selected, and the entire process is fully explainable, as each step can be traced and justified.

The framework was validated through two real-world case studies in the oil, gas, and aerospace supply chains. In addition to combining ML and MCDM methods, this work also showcases and integrates the aspect of interpretability and highlights its importance within a decision-making process.

The paper by Liang et al. [162] introduced reference dependence and loss aversion functions. The motivation for their work was derived from the idea that while machine learning models can predict preferences, they often lack transparency, making them less useful for practical decision support. Classical MCDM models, on the other hand, assume that criteria are independent or linearly additive, which does not realistically capture how people perceive trade-offs, gains, and losses.

To address this, the authors extended the MAU to what they call the Reference-dependent Multiplicative Multiattribute Utility (RMAU) function. Each attribute is modeled relative to a reference point, with gains and losses perceived differently. This results in better alignment between the function and real-world behavioral effects such as loss aversion.

The general form of the multiplicative utility function is given by:

$$v(a_1, a_2, \ldots, a_n) = \sum_{j=1}^{n} k_j v_j(a_j) + k \sum_{j=1}^{n} \sum_{q>j} k_j k_q v_j(a_j) v_q(a_q) + \ldots + k^{n-1} k_1 k_2 \ldots k_n v_1(a_1) v_2(a_2) \ldots v_n(a_n) \tag{3.4}$$

Here, $v_j(a_j)$ is the utility of attribute j, k_j represents the attribute weights, and k is a normalizing constant. To include reference dependence, each attribute-specific utility function is defined relative to a reference point r_j:

$$v_j(a_j) = \begin{cases} < 0, & a_j < r_j \\ 0, & a_j = r_j \\ > 0, & a_j > r_j \end{cases} \tag{3.5}$$

This captures the idea that falling below the reference point is seen as a loss, while increasing above is a gain. The slope of the utility function around the reference point is adjusted to reflect diminishing sensitivity and steeper penalties for losses. This is achieved via piecewise linear segments using parameters α_j and β_j:

$$\Delta v_{j,s+1} = \begin{cases} \alpha_j \Delta v_{j,s} & \text{if } s > r_j, \\ \beta_j \Delta v_{j,s} & \text{if } s < r_j. \end{cases} \tag{3.6}$$

The final model is estimated by solving a convex optimization problem that fits the utility function to observed preference data, ensuring that the piecewise segments remain smooth and interpretable.

Here, again, interpretability can be seen as an important consideration; however, aside from that, the real focus is on loss aversion. In some sense, the paper introduces an interesting way to represent soft constraints for each criterion, showing that failing to meet them can have a more pronounced effect on the final ranking of an option. This aligns well with problems related to emergency decision-making, whereby safety is often a critical factor and the satisfaction of certain safety-related factors is crucial.

The work of Liao et al. [164] proposed an Environmental Emergency Decision-Support System that used a hybrid framework combining a Case-Based Reasoning approach with an NN. Their motivation emerged from the idea that emergency response for sudden environmental incidents, such as hazardous chemical leaks, must be fast and reliable, yet in practice, it often relies on static plans or expert judgment that may be incomplete or outdated. The authors argued that historical accident cases contain valuable decision knowledge that should be reused to improve emergency response decisions.

The framework begins with the collection and digitization of historical environmental accident cases. Input features such as leakage quantity, hazard level, weather conditions, and nearby population are coded numerically through Feature Assignment . Output actions, such as emergency personnel deployment, equipment use, and containment measures, are defined according to Intensity Hierarchical levels that reflect how strongly each measure should be applied.

A Case-Based Reasoning system forms the backbone of this approach. When a new incident occurs, the system first searches the historical case library for the most similar past scenarios. This provides an initial basis for decision-making, even when direct training samples for the specific situation are limited. The retrieved case measures are then used as reference input–output pairs to help the NNmake consistent, robust predictions.

The NN itself maps current accident conditions to recommended emergency measures. It is trained on the digitized cases to learn the relationships between input features and the intensity of each action measure. To improve the network's performance and avoid local minima, a EA is used to optimize the NN's weights, hidden node configuration, and learning parameters in a global search process. However, the specific technical details of the network training are less critical than the fact that the NN serves to generalize knowledge across cases and produce a full set of recommended measures for new scenarios.

The output of the system is a vector indicating the recommended level for each measure, such as how many responders to deploy, how much equipment to use, and what containment actions to prioritize. A translation method is

then applied to convert these numerical outputs back into practical resource plans, based on conventional ratios and expert rules.

This papers deals directly with emergency decision-making and acknowledges many of the additional issues it presents in comparison to traditional MCDM methods in various scenarios such as supplier selection. Particularly interesting is the case-based approach, which acknowledges that the decision-making in an emergency scenario ought to be considered on a case-by-case basis, and that since it differs from standard practices, the decision needs to be considered within the confines of the current situation.

3.3.1 Unresolved Challenges and Research Gaps

Data challenges Despite promising advances, integrating ML with MCDM still faces significant challenges that warrant further research. One persistent issue is the complexity of accurately extracting, weighting, and modeling the interactions among multiple criteria from massive, heterogeneous, and often weakly correlated data sources, especially as real-world decision contexts become more dynamic and data-driven. Sparse and imbalanced data also remain a challenge, particularly for tasks such as preference learning, emergency decision-making, and modeling criteria interactions.

Dynamic adaptation Of particular note, and increasingly apparent in an emergency context, as shown by the data used in much of the literature, is the lack of adaptation. Specifically, once a model is trained to understand preferences from a large dataset, there seems to be a tendency to fixate on the average preferences of the decision-makers as a whole. This might work well under relatively stable conditions in which the decision-makers' preferences remain valid. However, would this still hold true if the situations or contexts in which these decisions are made change? For example, in supplier selection, would models trained to learn the decision-makers' preferences and rank suppliers still be suitable in the event of a pandemic? To what extent are these learned preferences robust to changes in the context in which they are applied? Does the system need to be updated with new data and preferences, and if so, how often?

Interpretability and explainability The black-box nature of many ML models is, again, a limiting factor. This is critical in high-stakes fields such as healthcare or emergency management, where decision rationales must be explainable to stakeholders. In addition, inconsistency in results when applying different ML techniques to the same problem suggests a lack of robust guidelines for selecting the most suitable algorithms for specific decision contexts.

Real world DSS Building integrated and flexible decision support systems that combine subjective human inputs with objective ML insights in a meaningful and trustworthy way continues to be an open research frontier. Addressing these gaps will require interdisciplinary approaches, the development of novel hybrid models, and a stronger focus on explainability, robustness, and adaptation to domain-specific needs.

3.4 Decision-Making Models and Support Systems for Cockpit Operations

Pilots are among the best-trained decision-makers, carefully selected and rigorously trained to handle unexpected situations and emergencies. While it is important to seek techniques to enhance and support their decision-making capabilities, it is equally valuable to understand the mental models and structured frameworks these experts already use to make critical judgments under stress.

3.4.1 Decision Mental Models

Pilots are trained to use structured decision-making models to handle unexpected or critical situations in the cockpit. These models provide a logical

framework to ensure that, under stress, crucial steps are not overlooked. They are an essential part of Crew Resource Management (CRM) and Aeronautical Decision Making (ADM) training. Before applying any model, pilots follow the basic principle "Aviate, Navigate, Communicate," meaning they maintain control of the aircraft, navigate away from danger, and communicate as needed. Once the aircraft is stable, a structured decision model can guide the crew through diagnosing the problem, exploring solutions, and executing a safe outcome [10]. To support this process, several structured decision-making models have been developed and refined through decades of aviation safety research and operational practice. What follows is a brief showcase of some of the most well-known frameworks used in cockpit operations for making decisions.

DECIDE Model The Detect, Estimate, Choose, Identify, Do, Evaluate (DECIDE) Model is one of the earliest and most widely taught frameworks. This model encourages pilots to define a problem clearly, systematically work through possible solutions, implement an action, and then evaluate its effectiveness. It is especially useful when there is time to analyze multiple options, such as deciding whether to divert due to weather [10].

FOR-DEC Model The Facts, Options, Risks & Benefits, Decision, Execution, Check (FOR-DEC) Model was developed by Lufthansa and the German Aerospace Center, DLR. It adds structure and logic to crew decision-making by emphasizing fact-gathering and risk analysis before committing to a plan. The model encourages all crew members to participate actively, thus improving coordination and helping to avoid snap judgments. Its structured approach has even been adopted beyond aviation, such as in nuclear power plants and medical crisis management [243].

T-DODAR Model The Time, Diagnose, Options, Decide, Assign, Review (T-DODAR) Model is commonly used in British aviation and was developed by British Airways. It emphasizes assessing the time available before problem-solving begins, which is vital in emergencies. The model then guides the crew through diagnosing the issue, considering options, deciding on a course of action, assigning roles clearly, and reviewing the outcome. This ensures that even under time pressure, the crew maintains a shared understanding and adapts as needed [24, 243].

PIOSEE Model The Problem, Information, Options, Select, Execute, Evaluate (PIOSEE) Model is similar to T-DODAR but uses slightly different terminology. It consists of problem identification, gathering information, developing options, selecting the best solution, executing it, and then evaluating the result. Some airlines prefer PIOSEE for its simplicity and clarity [60].

Other Models There are, of course, other models, and while the acronyms vary, they all share core principles: maintain situational awareness, generate options, analyze risks, make a decision, act, and review the outcome. These frameworks help pilots break down complex problems into manageable steps, ensure clear communication among crew members, and maintain safety even under high workload and stress. Through consultation with an experienced DLR pilot, it was also emphasized that the primary priority in any real cockpit situation is always flying the aircraft and managing the immediate, safety-critical actions, with navigation and higher-level decision making intentionally relegated to secondary and tertiary roles. Consequently, these models are designed not as rigid checklists but as cognitive scaffolds that support pilots in structuring their thinking while preserving the ability to take instinctive, time-critical action when necessary. Pilots are therefore trained to recognize when immediate intervention is required without stepping through every element of a decision model.

Integration with Pilot Mental Decision Models Combining highly trained and skilled decision makers with well-designed mental models and a robust, interpretable decision support system should augment human judgment during time-critical MCDM problems. Importantly, concepts for systems such as the one being developed in this work can be aligned with existing cockpit decision frameworks by providing structured, context-aware insights that fit well with current models. Namely, a system like the IPAS, with its ability to provide suggestions for dynamic alternate selection, can rapidly

evaluate and rank diversion options, which directly supports the Options, Risks, and Decision phases of many of the models. Rather than completely replacing the pilot's reasoning process, the system can serve as a cognitive aid that helps break complex, high-uncertainty situations into manageable components. In this way, human expertise and algorithmic assessment reinforce one another, enabling safety and efficiency even under demanding and rapidly changing conditions.

3.4.2 Cockpit Decision Support Systems

Decision support systems in the cockpit are not a new concept. Over the years, various prototypes and operational tools have been developed to help pilots manage complex tasks, unexpected scenarios, and dynamic decision-making under stress. These systems aim to complement the expertise of trained crews. Reviewing these earlier approaches provides useful context for the broader challenge of supporting decision making in dynamic, multi-criteria operational environments such as alternate-airport selection.

Diverter NASA's "Diverter" AI-based decision aid [236] is one of the earliest concepts specifically aimed at employing AI for alternate-airport selection and diversion support. The idea centered on continuously updating the feasibility of each potential diversion airport against strict operational constraints such as required runway length, aircraft landing performance, and weather conditions. The system focused primarily on constraint satisfaction and ranking feasible alternatives. However, the approach relied heavily on manually defined rules and logic, also illustrating how the meaning of the term AI has evolved over time. Although never fully implemented, Diverter clearly demonstrates early interest in using AI-inspired methods to support decision making in emergency, time-critical tasks such as alternate airport-selection.

The FINDER system [37] is an early knowledge-based cockpit decision aid developed to support crews during in-flight replanning and diversion scenarios. Its core idea was to continuously monitor weather, aircraft state, and operational constraints and to generate alternative routes or diversion options when conditions deviated from the original plan. FINDER used structured knowledge rules and an A*-based optimization mechanism to evaluate candidate routes according to criteria such as fuel requirements, airspace restrictions, and operational feasibility. A central element of the system was its explanation and dialogue component, allowing pilots to query why a particular option was suggested and to engage in "what-if" exploration. Although FINDER was not deployed operationally, it remains one of the most detailed attempts to design an intelligent cockpit assistant capable of dynamic replanning. FINDER also demonstrates that explainability and interpretability are not afterthoughts but rather important design principles in cockpit decision support. Its architecture shows that such qualities are essential for DSS of this type, reinforcing their inclusion as core requirements in the methodology being developed in this work.

CASSY and CAMA The CASSY and CAMA prototypes [203] represent an exploration of cognitive automation in aviation, with the aim of designing cockpit assistants that reason in a manner closer to human decision making. These systems were built around Rasmussen's model of human cognition and introduced modules for environment interpretation, pilot intent monitoring, conflict detection, and dynamic mission planning. Unlike traditional automation, which follows predefined rules, these assistants attempted to interpret evolving situations, detect deviations from expected behavior, and propose proactive solutions. CASSY/CAMA treated the assistant as a "third crew member," supporting—but not replacing—the pilot's judgment, particularly under high workload or rapidly changing conditions. Although still conceptual prototypes, they demonstrate a shift toward context-aware, cooperative decision support. This directly relates to the aims of this thesis: embedding structured reasoning, uncertainty handling, and interpretable recommendations within a system that remains subordinate to human expertise during time-critical operations such as alternate-airport

selection.

Enhancement of the Diversion Airport Selection Methodology

The 2020 study on enhancing diversion airport selection [298] presents a more recent approach to improving the decision process during diversion scenarios. The methodology focuses on integrating real-time operational constraints, such as runway performance data, weather conditions, aircraft limitations, and airport serviceability, into a structured evaluation framework. A key element of this approach is the construction of an Airport Diversion Risk (ADR) score, defined as a weighted combination of seven predefined risk elements. In their formulation, each element is represented by a severity score S_j and a corresponding availability or status index A_j. The overall diversion risk is computed as

$$\mathrm{ADR} = \sum_{j=1}^{7} S_j \cdot A_j, \tag{3.7}$$

where each term reflects a different operational factor, such as meteorological risk, aircraft performance constraints, fuel state, or airport service availability. Airports with lower ADR scores are preferred because they represent lower operational diversion risk. While the structure is transparent and easy to interpret, it relies on manually specified weights and a linear combination of factors, which limits adaptability and may oversimplify the interactions present in rapidly evolving, time-critical scenarios. Another critique of the proposed methodology is the fact that it simply highlighted criteria and proposed their weighted sum without providing the actual weights or specifying how they could be obtained.

Synthesis and Relation to This Work

As can be observed from these real-world concepts and methods, there is a clear interest in approaching alternate-airport selection and time-critical decision-making problems more broadly using structured support tools, especially in the field of aviation. However, many of the existing concepts are either purely conceptual or too rigid, relying on rule-based systems in environments that would benefit from more adaptable, data-driven approaches. Finally, it is noteworthy that the CASSY and CAMA systems place particular emphasis on transparency in their decision-support concepts, a sentiment that is shared and further developed in this work.

3.5 Summary of this Chapter

This chapter reviews decision-making in emergency management with a focus on MCDM, AI , and their integration into decision-support systems. It begins by presenting how MCDM methods have been applied across all phases of emergency management, including logistics planning, public health crises, natural hazard assessment, resilience evaluation, and post-disaster recovery. These applications demonstrate MCDM's ability to combine diverse information, handle uncertainty, and provide structured guidance for complex choices. At the same time, the chapter highlights key limitations, such as the static nature of many models, the difficulty of processing heterogeneous and dynamic information, and the limited ability to capture interdependencies between attributes—factors that are particularly critical in rapidly evolving emergency scenarios and time-critical emergency decision-making and that this thesis aims to adress.

The chapter then examines the role of AI in emergency contexts, outlining how machine learning techniques have been applied to predict hazards, support real-time decisions, and enhance situational awareness. Examples include earthquake and flood prediction, wildfire monitoring, and human detection in search-and-rescue missions. While these studies underline the potential of AI to uncover patterns and improve decision support, challenges remain regarding data scarcity, robustness, trustworthiness, and explainability. Ethical considerations are also central, particularly the need to address issues of bias, fairness, accountability, and privacy when deploying AI systems in high-stakes envi-

ronments where human lives are at risk. Of particular note are aspects such as robustness from a performance standpoint, given that in a dynamic environment like alternate-airport selection, intelligent DSS should be capable of adapting to changing circumstances and providing resilient suggestions for a course of action. Furthermore, the inclusion of elements such as accountability, trustworthiness, and transparency is vital if AI-based systems are to be employed in emergency decision-making.

Next, the chapter explores the integration of AI and MCDM, where hybrid approaches combine the strengths of both. AI can support tasks such as criteria extraction, modeling interdependencies, and learning parameters from data, while MCDM provides a transparent framework for structuring and explaining results. Integrated decision-support systems are presented to illustrate how these approaches can be applied in practice, but unresolved challenges remain, especially regarding adaptation to changing conditions, maintaining interpretability, and ensuring robust cooperation between human decision-makers and automated systems.

Finally, the chapter turns to the aviation domain, where decision-making under pressure is most evident. Mental models such as DECIDE, FOR-DEC, T-DODAR, and PIOSEE illustrate the structured approaches that pilots use to make critical judgments under stress. These frameworks are complemented by cockpit decision-support systems such as FINDER, CASSY, and CAMA, which highlight how knowledge-based reasoning and cognitive automation can assist crews in complex and time-critical situations. Together, these examples show both the interest in the topic and the relation to practical problems, as well as some of the challenges faced when designing a system that can handle atypical scenarios.

Part II

Methodology Development

4 Approach Development

This chapter illustrates the steps taken to address some of the challenges associated with MCDM in time-critical emergency applications. It explores the authors publications, highlighting the lessons learned in each paper and how it relates to the problem. By doing so, the chapter illustrates the progression of ideas, methods, and practical considerations. The aim is to showcase how certain challenges associated with the problem were addressed and provide a clearer train of thought as to why certain approaches were employed.

4.1 Bi-objective optimization problem

The following is largely based on the author's paper "Description and First Evaluation of an Approach for a Pilot Decision Support System Based on Multi-attribute Decision Making" [77],

4.1.1 Main Idea

Having identified the need for an approach to help decision makers in time-critical emergency contexts, both as a gap in the literature and as a real world need as shown by pilot interviews in [279], the initial idea was simply to determine how to weigh the specific factors in a weighted-sum style. This was a straightforward concept and aligns well with the AI and MCDM literature, which often uses AI or similar methods to learn how to weight the factors and then incorporates these learned weights into an automation mechanism that uses them to rank possible options.

In [77], a two-stage approach was proposed. In the initial stage, the available alternatives are assessed and filtered so that only the most viable options remain. In the second stage, the system uses the learned weights to appropriately assign a score to each option and thus provide the decision makers with a ranking and, by logical extension, a recommended go-to option. A visual representaion of the approach is shown in Figure 4.1.

The goal was to develop and conduct an initial evaluation of this approach and explore whether it could serve as a component of a pilot decision-support system based on MCDM methods, addressing the dynamic alternate-airport selection problem. The aim of such a system would be to quickly provide decision-makers with a course of action during an emergency in an effort to salvage a failed mission. The term 'mission' in this context refers to the successful transportation of goods and/or passengers from point A to point B. Thus, by salvaging the mission, the idea is that the approach and any system built upon it will help the decision makers (in this case, the pilots) make a decision that aims to achieve the best possible outcome in the predicament they have found

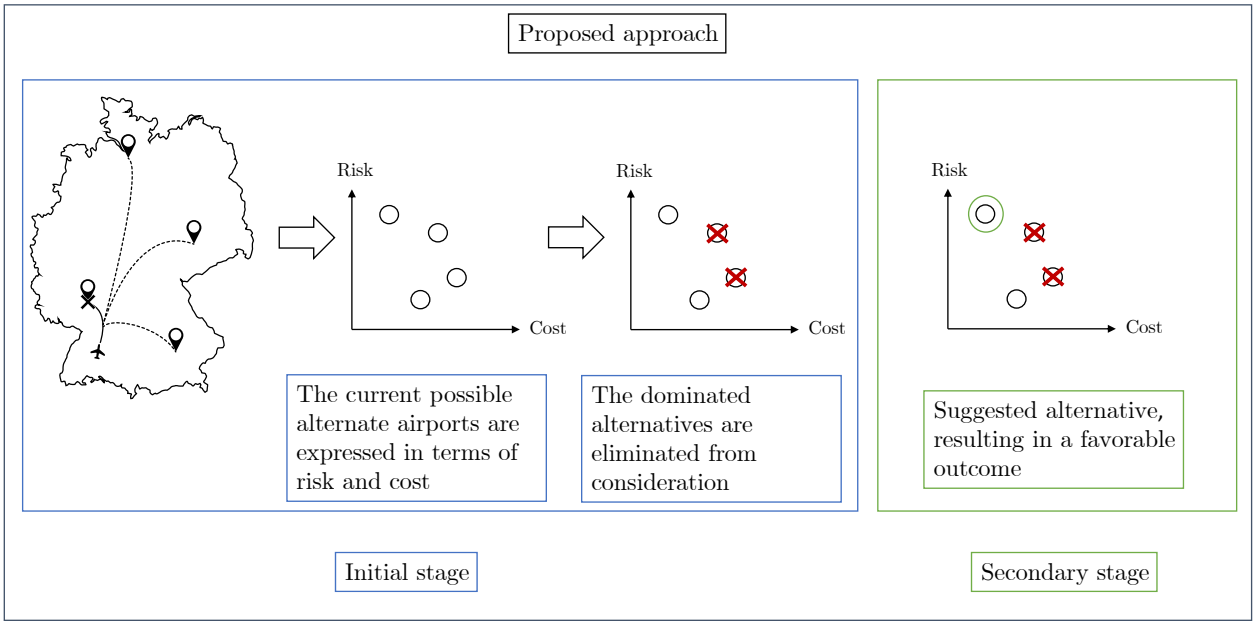

Figure 4.1: Graphical representation of the proposed approach, taken from [77]

themselves in.

4.1.2 Optimization Problem description

To include both a filtering mechanism and a way to determine the weights for the features, the decision-making problem was reframed as a scenario-based bi-objective optimization problem consisting of the following objectives:

1. Risk, which indicates the extent to which the decision affects the safety requirements.
2. Cost, which measures the financial implications of the chosen airport.

Using these criteria, in the initial stage, the options are filtered so that only those that are part of the Pareto front (that is, the best trade-off between the objectives) remain. In the next stage, these objectives are used together to determine the weights of the factors.

Risk objective

Within the paper [77], the risk was initially expressed as:

$$Risk = \frac{1}{(ld_{have} - ld_{need})} + cw + \frac{1}{f_r} \tag{4.1}$$

where ld_{have} and ld_{need} denote the available and required landing distances, respectively, cw is the crosswind component in knots at the alternate airport, and f_r refers to the remaining fuel upon arrival at the selected airport.

However, for various critical situations, a piecewise definition was used to ensure that particularly dangerous combinations are assigned a significantly higher risk value:

$$Risk = \begin{cases} 12 & \text{if} \quad cw > 38 \text{ kn} \\ 15 & \text{if} \quad ld_{need} > ld_{have} \\ 20 & \text{if} \quad f_r < 1600 \text{ kg} \\ 25 & \text{if} \quad ld_{need} > ld_{have} \text{ and } cw > 38 \text{ kn} \\ 30 & \text{if} \quad ld_{need} > ld_{have} \text{ and } f_r < 1600 \text{ kg} \\ 35 & \text{if} \quad ld_{need} > ld_{have},\ f_r < 1600 \text{ kg},\ \text{and } cw > 38 \text{ kn} \end{cases} \tag{4.2}$$

To account for the different scales of these variables, the landing distance and fuel values were normalized on a scale of one to ten, while the crosswind values were normalized from zero to one. This ensures that the combined risk function remains positive and that each component contributes meaningfully and proportionally: for example, an increase in crosswind always results in an increase in the risk value. The thresholds and normalization ranges were chosen based on findings from pilot interviews [279] and the Airbus Flight Crew Operating Manual (FCOM) [13].

Cost objective

The cost is defined as:

$$Cost = (d_{planned} - d_{actual}) \times cf \pm f_u \tag{4.3}$$

where $(d_{planned} - d_{actual})$ represents the difference between the aircraft's planned final position and its actual final position, i.e., the gap between where the aircraft was intended to land and where it ultimately did. The underlying idea is that a larger deviation from the original plan results in higher financial costs for the airline, as passengers may need to be compensated, accommodated, or rerouted to reach their intended destination. The cost factor cf, which is currently set to 1, allows airlines to adjust the model to better reflect their specific cost structures if needed. The term f_u accounts for the fuel consumed. It should be noted that the purpose of these risk and cost metrics is not to produce precise or absolute figures but rather to capture the key considerations that pilots are likely to weigh when making time-critical decisions.

Clarification on how the objectives relate to one another

Given how the objectives are formulated, some clarifications are necessary regarding their dependence on the positions of the aircraft and the airports. According to the flight crew operating manual (FCOM) [13], the landing distance required for an aircraft depends greatly on its mass and the condition of the runway. The mass of the aircraft can be seen as consisting of two parts: the passengers and luggage, which remain constant, and the fuel on board, which changes during flight depending on the distance flown. In addition, the condition of the runway, its length, and the prevailing wind conditions all depend on the specific airport, meaning they are tied to its location. Therefore, the key variables linking both objectives are the locations of the airports and the aircraft, along with the initial fuel load. The objective values were structured and presented in this way to reflect the type of information and variables that would realistically be available and relevant to a human decision maker, such as a pilot. In this sense, the components of the objective functions represent higher-level factors that already incorporate other variables within them.

4.1.3 Tackling the Optimization Problem

To learn how to weigh the relevant factors, a scenario-based approach was employed in which a dataset of decision-making scenarios was created. Simply put, a dataset of possible decision-making problems was and used as a dataset in order to learn how to weigh the factors such that they would be applicable in a wide range of emergency scenarios.

Regarding this scenario-based dataset, a weighted sum was employed to determine the option chosen for a specific scenario based on the best value. Mathematically, the choice for a scenario S_i can be expressed as:

$$C(S_i) = min(S_i \times \vec{w}) \tag{4.4}$$

where $\vec{w}$ represents a vector of weights corresponding to the m number of characteristics of the options/airports, within scenario S_i, i.e., $\vec{w} \in R$ is an m dimensional vector. The goal is to find a weight vector $\vec{w}$ that minimizes the following objective functions:

$$\begin{aligned} \min_{\vec{w}} \quad & Risk_{avg}(\vec{w}),\ Cost_{avg}(\vec{w}) \\ \text{s.t.} \quad & \textstyle\sum_{i=1}^{m} w_i = 1 \end{aligned} \tag{4.5}$$

where $Risk_{avg}$ and $Cost_{avg}$ are the average risk and cost values for the whole data set, i.e.,

$$Risk_{avg}(\vec{w}) = \frac{1}{k}\sum_{i=1}^{m} Risk_i \tag{4.6}$$

$$Cost_{avg}(\vec{w}) = \frac{1}{k}\sum_{i=1}^{m} Cost_i \tag{4.7}$$

where the $Risk$ and $Cost$ values for an airport/option are calculated based on equations 4.1 and 4.3. As can be seen, the learning of the weights can be condensed to a MOOP.

Scenario Dataset

Since real-world aviation data is highly sensitive and not readily available, with limited well-structured datasets regarding emergency scenarios, the scenario-based dataset was created using information from the pilot survey [279] and the flight crew operating manual [13]. The dataset consisted of many scenarios, each representing a moment in which pilots must make a decision under certain conditions.

Each scenario was structured as an $n \times m$ matrix S_i with $n = 11$ representing the major commercial airports in Germany. One airport was always the planned destination and therefore unavailable as an alternate [274]. The remaining $m = 6$ features describe the conditions at each airport:

- Longest runway length available $[m]$.
- Wind speed at the location of the airport $[kn]$.
- Wind direction at the location of the airport [°].
- Status of the runway, expressed as a value from 0 to 6 4.1.
- Distance from current aircraft position to planned destination $[nm]$.
- Distance from current aircraft position to appropriate alternate airport $[nm]$.

Runway lengths were sourced from the BlueSky ATC Simulator Project [123]. Wind speed, direction, and runway status were generated using Latin hypercube sampling within defined ranges:

- Wind speed, 5 to 45 $[kn]$.
- Wind direction 0 to 359 [°].
- Runway status from 0 to 6, see Table 4.1.

Higher wind speeds were favored to make the scenarios more challenging. Wind speed and direction were transformed into crosswind and tailwind components to better reflect how pilots interpret wind data, as illustrated in Figure 4.2.

For each scenario, the aircraft's position was defined using realistic flight plans between airports. These plans were taken from an online flight planner used by amateur pilots [204]. In total, 26 routes were created connecting the 12 airports, with waypoints selected within the cruise phase of each flight, shown in Figure 4.3, to simulate realistic decision points. The most frequent routes were determined using data from FlightRadar24 [101].

The fuel required was calculated by simulating the flights, from take-off to landing at each of the possible available options. The custom-made simulation only encompasses the Airbus A320 aircraft and makes use of information from the Airbus A318/A319/A320/A321 Flight Crew Operating Manual [13], the

Code	Runway condition description	Reported braking action
6	Dry	Dry
5	Wet or damp surface; up to 1/8" (3 mm) of slush, dry snow, or wet snow	Good
4	Frost; compacted snow (outside air temperature at or below −15°C)	Good to medium
3	Slippery when wet; more than 1/8" (3 mm) of dry snow (max. 5" / 130 mm) or wet snow (max. 1 $\frac{1}{8}$" / 30 mm); compacted snow (outside air temperature above −15°C)	Medium
2	More than 1/8" (3 mm) of water (max. 1/2" / 12.7 mm) or slush (max. 1/2" / 12.7 mm)	Medium to poor
1	Ice (cold and dry)	Poor
0	Wet ice; water on compacted snow; dry or wet snow over ice	Nil

Table 4.1: Airport estimated runway condition assessment.

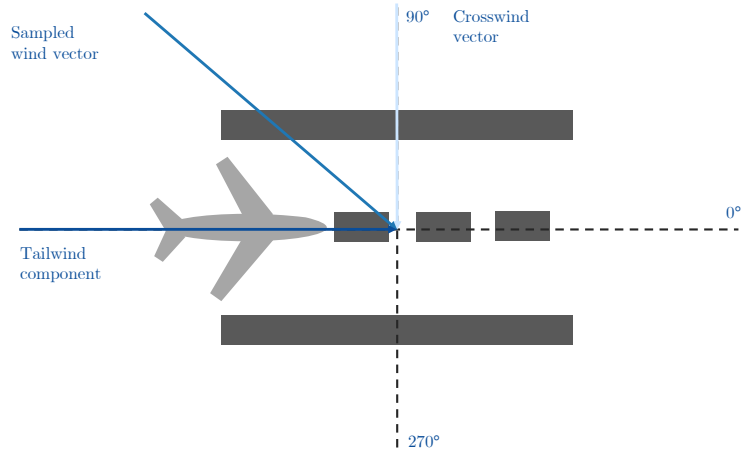

Figure 4.2: Relative wind direction representation, adapted from [77]

Eurocontrol A320 performance database [88], and consultation with a pilot employed at the DLR.

In summary, each scenario represented a snapshot of a situation in which a diversion decision must be made. Twelve airports were considered, one of which is always unavailable. Each airport option was defined according to the key features described above. Combining 100 variable combinations for wind and runway conditions with 26 flight plans resulted in a dataset of 2600 scenarios, structured as the tensor $D_{k,n,m}$

4.1.4 Result analysis

For the determination of the weight vectors $\vec{w}$, the NSGA-II algorithm [70] was employed. The experiments were conducted using the python framework pymoo version 0.5.0. [38]. The algorithm was run 31 times to control for the heuristic component element of the NSGA-II algorithm.

A cumulative Pareto front, as presented in Figure 4.4, was created by compiling the non-dominated solutions identified across all 31 runs. From this figure, it can be seen that the solutions tended to cluster near the extreme ends of the objective axes, leaving a noticeable gap in the central region. One possible explanation for this pattern is that the basic weighted sum approach may not have been able to fully capture the complexities of the trade-offs involved. Additionally, the choice of hyperparameters likely contributed to this distribution, and applying more refined hyperparameter tuning could help generate a smoother, more continuous front. In the paper, it was specified that greater focus should be placed on solutions with average risk values around 8 or lower.

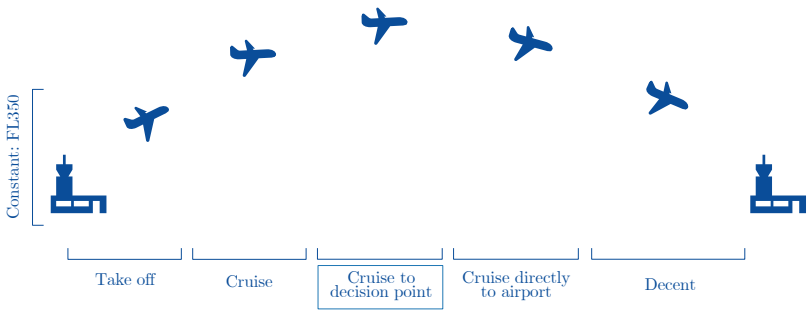

Figure 4.3: Phases of a flight, adapted from [77]

Selecting solutions with minimal cost but risk levels exceeding 8 was deemed unwise, as such values would come dangerously close to catastrophic failure.

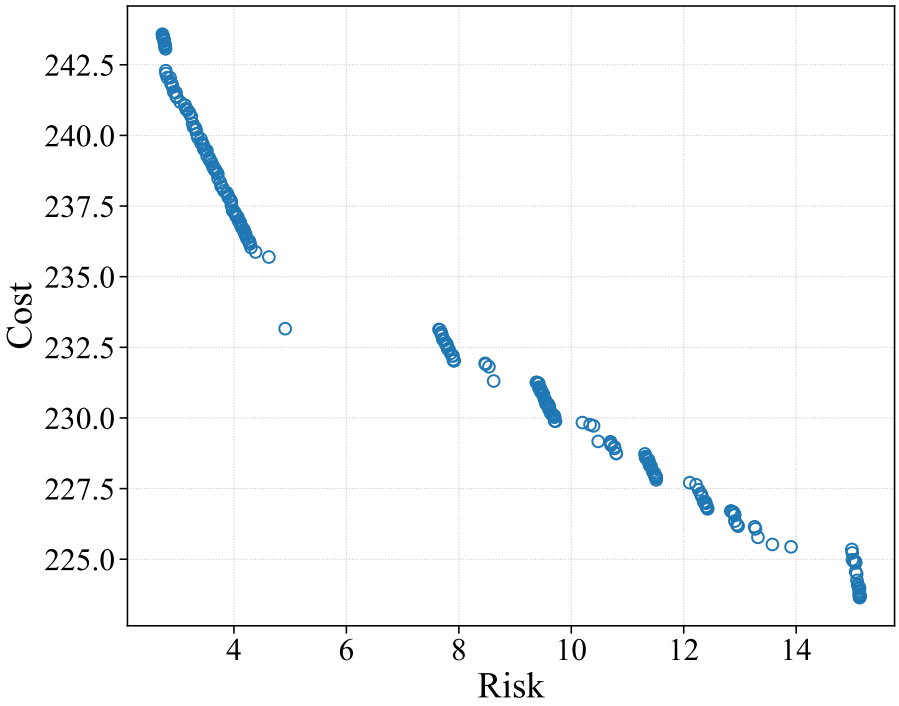

Figure 4.4: Front of non-dominated solutions from all runs of the algorithm, adapted from [77]

4.1.5 Lessons Learned

The paper served as an initial investigation into how an MCDM problem could be addressed under uncertain, time-sensitive, and atypical situations in which the decision-makers represent multifaceted interests, are directly embedded in the situation, and will bear both the responsibility for and the consequences of the outcome. The limitations of traditional MCDM approaches were identified as making them unsuitable for such emergency scenarios, and a decision-support-based approach was explored. This approach considered how the decision-making strategy could be determined in advance and then, when needed, called upon to provide assistance in reaching the final decision, similar to the automation-focused AI–MCDM hybrid methods.

Data availability

The paper highlighted a common issue regarding data availability, noting that certain types of data, such as those related to emergencies, are less common,

less accessible, and often have more problematic distributions. It also emphasized that real-world problems must contend with legal, safety, and practical constraints when gathering data. To address this, a scenario-based approach was employed, using synthetic data generation techniques informed by expert knowledge and other available sources to produce suitable inputs. The work further demonstrates that many real-world problems will not come with clean, ready-to-use datasets. Instead, it is often necessary to aggregate information from different sources to construct inputs that the model can learn from and that match the type of inputs the system will encounter when put to use.

Factor constraints

Through the bi-objective formulation, what could be observed is that in safety-focused fields (and perhaps many that deal with emergencies), there are certain factors that need to be satisfied to a certain degree while optimizing the other goals as much as possible. Additionally, soft constraint mechanisms are also necessary as scenarios may arise in which all options break certain constraints, thus eliminating the possibility of simply filtering all options based on constraint violations.

4.2 Multi-objective Multiplexer and Context

The following is largely based on the author's paper "Multi-objective Multiplexer Decision Making Benchmark Problem" [76],

4.2.1 Main Idea

Inspired by the alternate-airport selection problem, which the paper defines as the Dynamic Alternate Airport Selection (DAAS) Problem, and recognizing parallels with the well-known multiplexer problem, a new scalable multi-objective benchmark problem was proposed. The Multi-objective Multiplexer Decision-Making Benchmark Problem was developed to enable the exploration of a multi-objective decision-making problem that is decoupled from the real data requirements of the DAAS Problem, given that data from the aviation industry is not easily obtainable. Although similar benchmark problems have been proposed [52], they have typically been defined with a fixed number of objectives and cannot easily be scaled to larger or more complex scenarios. This is further compounded by the fact that they are often designed around simplified games, which is common practice in the reinforcement learning field.

The new benchmark combined two well-known problems from the literature: the multiplexer problem [146] and the DTLZ problem [71, 68]. The multiplexer was selected for its incorporation of interactions between its features and its heterogeneous structure. This was conducted due to its similarity with the way pilots must prioritize different airport characteristics depending on the situation. The DTLZ problem was included because it captured the trade-offs inherent in the pilot's decision-making process and due to its known Pareto front, which can be scaled as necessary. Thus, the benchmark problem was designed to reflect the challenges seen in the DAAS problem while remaining conceptually straightforward, scalable, and suitably challenging as a test problem.

4.2.2 Component problems

To provide a better understanding of the newly proposed benchmark problem, a short primer will be given on both the multiplexer and DTLZ problems.

Multiplexer problem

The n-bit multiplexer is a classic benchmark problem for LCS. It is a Boolean function defined over a binary string of length $n = k + 2^k$, comprising k address bits and 2^k data bits, also called register bits [259]. Conceptually, the multiplexer is based on an electronic multiplexer circuit in which the address bits select one of the data bits, and the output is the value of that selected data bit. In this problem, each input instance is a binary string, and the task is to

predict the single output bit, usually thought of as a class label, determined by this addressing scheme.

A typical example used in demonstrations is the 6-bit multiplexer (here, $k = 2$, so $n = 6$). This instance has 2 address bits and $2^2 = 4$ data bits. In the example, the input `010110` is split into an address part `01` and a data part `0110`. The address `01` (binary for 1) points to the data bit at index 1 (counting from 0) in the data part. The data bits in this example are `0110`, and the bit at index 1 (the second data bit) is 1. Thus, for this input, $f(010110) = 1$.

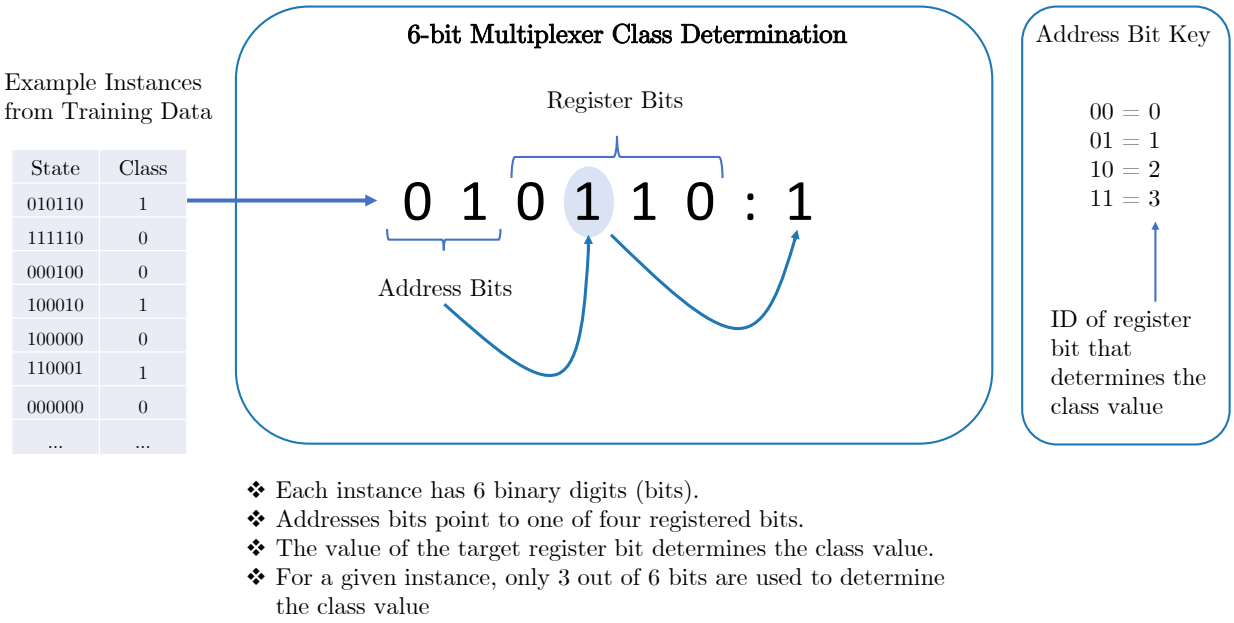

Figure 4.5: 6-bit Multiplexer Problem Example, adapted from [259]

In general, for the 6-bit multiplexer, an address of `00` (decimal 0) selects the first data bit, `01` (decimal 1) selects the second data bit, `10` (decimal 2) selects the third, and `11` (decimal 3) selects the fourth data bit as the output. The same principle extends to any n-bit multiplexer: for k address bits, there are 2^k possible address values (contexts), each selecting one corresponding data bit to determine the output.

Learning algorithms is challenging for the multiplexer problem because of its epistatic structure [259] whereby epistasis means that the output depends on an interaction of multiple bits rather than any single feature in isolation. In a multiplexer, knowing the value of any one bit alone provides no information about the class label, thus the contribution of a given bit to the output is only meaningful in conjunction with others. For instance, the value of a particular data bit matters only when the address bits point to it, and conversely, the address bits only indicate the output when combined with the corresponding data bit's value.

Aside from epistasis, the problem is also characterized by heterogeneity , meaning that different subsets of features determine the class in different instances. Each unique address value defines a distinct context in which a different data bit becomes the relevant feature for predicting the output. Thus, the function effectively consists of 2^k local sub-functions (one for each address context), each involving an interaction of $k + 1$ bits (the k address bits plus one data bit).

These characteristics have resulted in the multiplexer being a difficult function for many machine learning methods to learn [249]. Nonetheless, the multiplexer has served as an important and rigorous test for LCS algorithms, precisely because it encapsulates high-order feature interactions and context-specific fea-

ture relevance in a clean, scalable form [259]. The presence of varying importance between the factors depending on the situations is what makes the multiplexer problem and DAAS problem so similar. Namely, in the DAAS problem, the characteristics of the actual aircraft, i.e., its position, fuel level, altitude, and planned mission, represent the address bits. They dictate which characteristics representing the airports should be considered more and by how much more. This is one of the reasons that the multiplexer problem was chosen.

DTLZ benchmark problem

The DTLZ benchmark problems were introduced by Deb et al. in [68] and further expanded in [71]. In these works, the authors describe a bottom-up approach for constructing test problems: the shape of the Pareto-optimal front is defined first, and the objective space is then formulated accordingly. The suite includes multiple problems, labeled DTLZ1 through DTLZ9, distributed across the two papers. For the purposes of this work, only the DTLZ2 problem is illustrated. The DTLZ2 problem is defined mathematically as follows:

$$
\begin{aligned}
&\min(f_1(\vec{x})) = (1 + g(x_m)) \cos(x_1\pi/2) \dots \cos(x_{m-1}\pi/2), \\
&\min(f_2(\vec{x})) = (1 + g(x_m)) \cos(x_1\pi/2) \dots \cos(x_{m-2}\pi/2) \sin(x_{m-1}\pi/2), \\
&\min(f_3(\vec{x})) = (1 + g(x_m)) \cos(x_1\pi/2) \dots \sin(x_{m-2}\pi/2), \\
&\dots \\
&\min(f_m(\vec{x})) = (1 + g(x_m)) \sin(x_1\pi/2), \\
&\text{where} \\
&g(x_m) = \sum_{x_i \in x_m} (x_i - 0.5)^2, \quad 0 \le x_i \le 1, \quad i = 1, \dots, n.
\end{aligned}
$$

In this formulation, $\vec{x}$ is a vector containing $k = n - m + 1$ variables. The Pareto-optimal solutions occur when $x_i = 0.5$ for each $x_i \in x_m$, ensuring that the objective values satisfy the condition $\sum_{i=1}^{m} f_i^2 = 1$. Figure 4.6 shows an example of the DTLZ2 problem tackled by the NSGA-II algorithm for a case with three objectives. Including a multi-objective optimization element like this reflects the trade-offs pilots must weigh when selecting an alternate airport during dynamic decision-making.

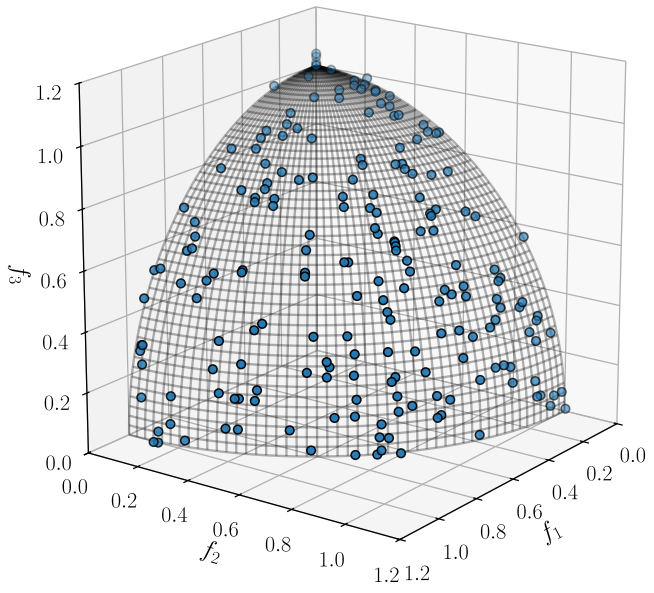

Figure 4.6: NSGA-II population on the DTLZ2 test problem

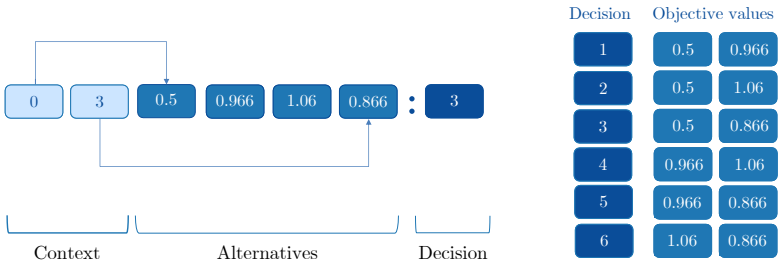

Figure 4.7: General structure of the MOMP, adpated from [76]

4.2.3 Multi-objective Multiplexer Decision Making Benchmark Problem description

In [76], the Multi-Objective Multiplexer Problem (MOMP) was defined as a multi-objective, multi-class classification task in which the aim is to select the correct class—choosing the appropriate option from a set of possible alternatives. This problem is designed as a synthetic dataset, in which each instance is composed of a sequence of positive real numbers, in a similar vein to the multiplexer problem. Hence, each instance follows a structure similar to the traditional multiplexer problem. It consists of two main segments:

1. Context section, which is meant to mimic the adress bits from the multiplexer problem.
2. Alternatives section, which mirrors the register bits.

A visual overview of the general layout of the MOMP is presented in Figure 4.7.

In the example shown, the string of numbers represent a single instance drawn from a dataset that can be easily produced synthetically. The correct context values indicate the positions of the objective values that lie on the Pareto front, which, in this design, are taken from the DTLZ problem, satisfying the condition $\sum_{i=1}^{m} f_i^2 = 1$. The total number of possible choices equates to the number of unique ways the alternative values can be arranged. The total number of classes or possible selections is given by $\binom{n}{k}$, where n is the number of alternative values, and k is the number of objectives. The number of context variables depends on the number of objectives, which is left to the user's discretion. Combinations rather than permutations are used to ensure that there is only one Pareto-optimal selection for each case. One additional detail is that the objective values are always positioned from left to right as they appear in the sequence. The possible options are then derived from the combinations of the remaining alternative values. The method for generating these decision combinations can be chosen freely, but it is essential that the same method is applied consistently across the entire dataset. For instance, decision one could always mean selecting the first and second values in the sequence in that specific order.

To increase the challenge and to blur the distinction between context and alternative parts, the context values are substituted with numbers that fall within predefined ranges. This design choice reflects real-world conditions such as the DAAS scenario, where distinguishing relevant variables can be difficult. This is accomplished by dividing an interval, such as the range from 0 to 1, into equally sized sub-intervals that match the number of available alternatives. For example, as shown in Figure 4.7, if four alternatives are possible, the range is divided into segments of size 0.25. A context value of zero may then be replaced by any

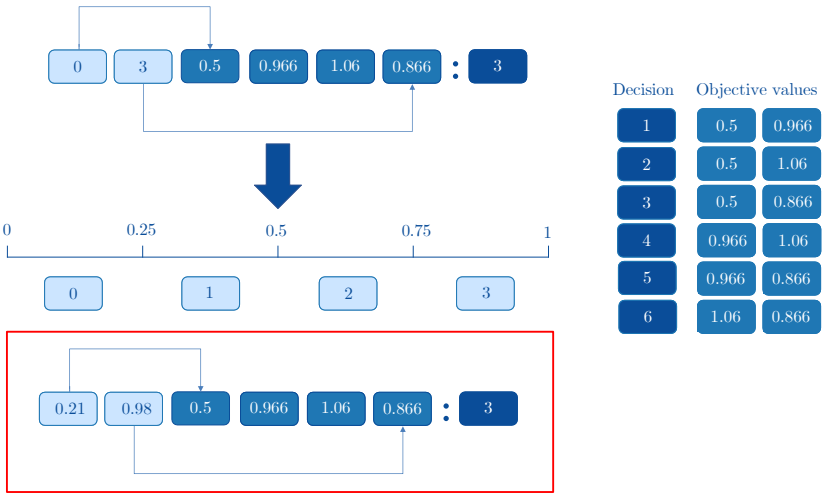

Figure 4.8: Final MOMP, adapted from [76]

number within the interval [0, 0.25]. The complete version of the benchmark problem is shown in Figure 4.8.

MOMP is formulated as a multi-objective optimization problem with the aim of minimizing the objectives. The specific goal is to find

$$\min(F_1, F_2, F_3, \ldots, F_m)$$

where each F_m is calculated as the average of all values for objective m across the complete dataset of length n:

$$F_m = \frac{1}{n} \sum_{i=1}^{n} f_m.$$

Essentially, the overall aim is to determine the decisions that minimize the objectives across the entire dataset.

4.2.4 Lessons Learned

In this paper, the alternate-airport selection problem is more formally defined and termed the DAAS problem, which will be the term used moving forward. It also showcased a scalable benchmark problem that combined elements of two well-established problems from neighboring areas in the literature, which together reflected significant similarities to the practical DAAS problem.

Context inclusion

Aside from proposing the new MOMP benchmark problem, the key insight from this paper that relates to MCDM is the incorporation and emphasis on context within decision-making. Namely, it recognizes that the options available and their factors are viewed differently depending on the context in which they are being made. In fact, the paper argues that the use of preference-elicitation methods such as weight vectors, pairwise comparison, and preference points serve to both express decision-makers' preferences and encode context-related information within the problem. This is of great importance to atypical decision-making problems such as those in emergencies, as it is likely that these contexts will be varied and therefore of greater importance to the decision-making process. For example, in the DAAS problem, the same set of alternate airports with the same defining factors would be viewed rather differently if the pilots also had to consider an engine failure or a medical emergency. This context lens shapes the way in which the options are viewed. Hence, one of the greater contributions from this paper is the integration of the context aspect within the scenario-based approach for MCDM in time-critical emergencies, allowing for decision-making approaches and strategies to also take into account

the circumstances in which the decisions are being made and thus have greater adaptation to the decision-making problem at hand, which is an important consideration in decision-making in emergency and time-sensitive scenarios.

Learning Classifier Systems model focus

In the paper to tackle the MOMP problem and an LCS was used as an initial method to tackle the problem. The LCS was used as it was one of the methods closely associated with the multiplexer problems while also being a model characterized by its explainability due to its more transparent and rule based structure. Due to this it became one of the models of greater interest going forward.

4.3 Learning Directly from Decision-Makers

The following is largely based on the author's paper "Wings of Wisdom: Learning from Pilot Decision Data with Interpretable AI Models" [80].

4.3.1 Main Idea

Given the somewhat limited availability of data, the aim was to go directly to the source and determine what information could be collected and how it might be applied within an AI–MCDM hybrid framework. To achieve this, a survey was carried in which pilots were asked to resolve possible variants of the DAAS problem. The resulting dataset would then serve to explore how interpretable AI models, such as the LCS, could be trained on this information and potentially integrated as a component in systems that could offer decision support.

4.3.2 Dataset

Dataset acquisition

The dataset was initially collected to gather greater information about the importance of certain factors regarding the DAAS problem based on a master's thesis entitled "Are there any factors that make the pilots' decisions predictable? An analysis of the impact of conditions on the choice of alternate airports" [141]. The data collection process relied on an anonymous online questionnaire using LimeSurvey [166]. This survey ran from October 28 to November 12, 2022, and gathered responses from 46 participants. Recruitment was performed via direct email invitations sent to contacts listed in the Institute of Flight Guidance's mailing list, with no monetary or other incentives offered to participants.

The study included the following components:

- A single inclusion condition: an active type rating for the Airbus A320 family (binary response).
- Sociodemographic details: birth year and gender.
- Questions on professional experience: total flight hours, hours flown in the preceding year, flight hours on the A320 type, and current pilot rank.
- A set of twelve decision-making scenarios.

MCDM scenario characteristics

In each of the twelve scenarios, the participants were asked to rank three possible alternate airports according to their preference in the event of a mid-flight diversion. The underlying reason for the diversion was intentionally unspecified, and scenarios operated on the assumption that the aircraft remained fully operational without any technical issues. To maintain standard conditions, each scenario described the same baseline context:

An Airbus A320 carrying a full passenger and baggage load amounted to a total weight of 64.5 tons at its maximum allowable landing weight, cruising at FL350. Weather conditions were set to standard ISA, with calm winds

until reaching the alternates. Landings were to be performed manually, with autothrust disengaged, medium auto brakes, and reversers in use. This setup ensured that the aircraft's operational status reflected an average, realistic configuration for this type.

Each of the twelve scenarios included detailed information on the relevant airport characteristics:

1. Distance between the original destination airport and the alternate.
2. Landing performance indicator, translating runway surface conditions (e.g., wet, icy) into expected landing behavior.
3. Margin between the available runway length and the aircraft's required stopping distance.
4. Crosswind component at the airport.
5. Tailwind component, calculated using wind direction and speed.
6. Fuel reserves remaining when reaching the alternate's airspace, relative to the minimum regulatory requirement.
7. Total number of runways available at the alternate.

To minimize bias, each airport was labeled with generic two-letter codes, preventing pilots from being influenced by personal experience or preferences linked to real airport names. The scenarios were purposefully designed to be challenging, including more extreme cases where the optimal choice was less obvious. As discussed in [141], these complex scenarios aimed to reveal deeper insights into the trade-offs pilots make when balancing various airport factors. The scenarios were also classified as either difficult or very difficult, with difficulty defined by how close the parameters came to exceeding standard safety margins.

Dataset Analysis

Figure 4.9 presents the Rank 1 airport preferences chosen by the pilots for each scenario. It should be noted that although a total of 46 pilots participated in the study, not every participant completed all scenarios, which accounts for the slight variation in the number of responses visible in Figure 4.9. The figure shows that in several scenarios, participants tended to agree more strongly, resulting in clearer preferences. However, in some cases, such as scenarios S7 and S9, the difference between the airport most frequently ranked first and the second most preferred option is relatively small, which indicates that the level of agreement was lower. Despite these occasional differences in individual responses, the dataset remains highly valuable as it captures insights from experienced professionals, making it a robust source of information for training and evaluating models.

In the study described in [141], the scenario data were mainly analyzed by considering all airport options as separate entries within one comprehensive dataset. While there was some examination of each scenario as an individual unit, this was addressed to a lesser extent. To gain more detailed insight into the factors shaping the pilots' decisions, an additional approach was used in which each scenario was treated as its own subset within the dataset. The first step in this process involved simplifying the target variable. Instead of working with three separate variables representing each ranking, the pilots' responses were combined into a single score for each airport. This score was created by incorporating all three ranks as follows:

$$SC(a_i) = R1(a_i) + \frac{R2(a_i)}{2} + \frac{R3(a_i)}{3} \tag{4.8}$$

In this equation, a_i represents a specific airport, and Rm corresponds to the ranking position from the survey results. This scoring method makes it possible to include the influence of all ranking positions rather than only focusing on the

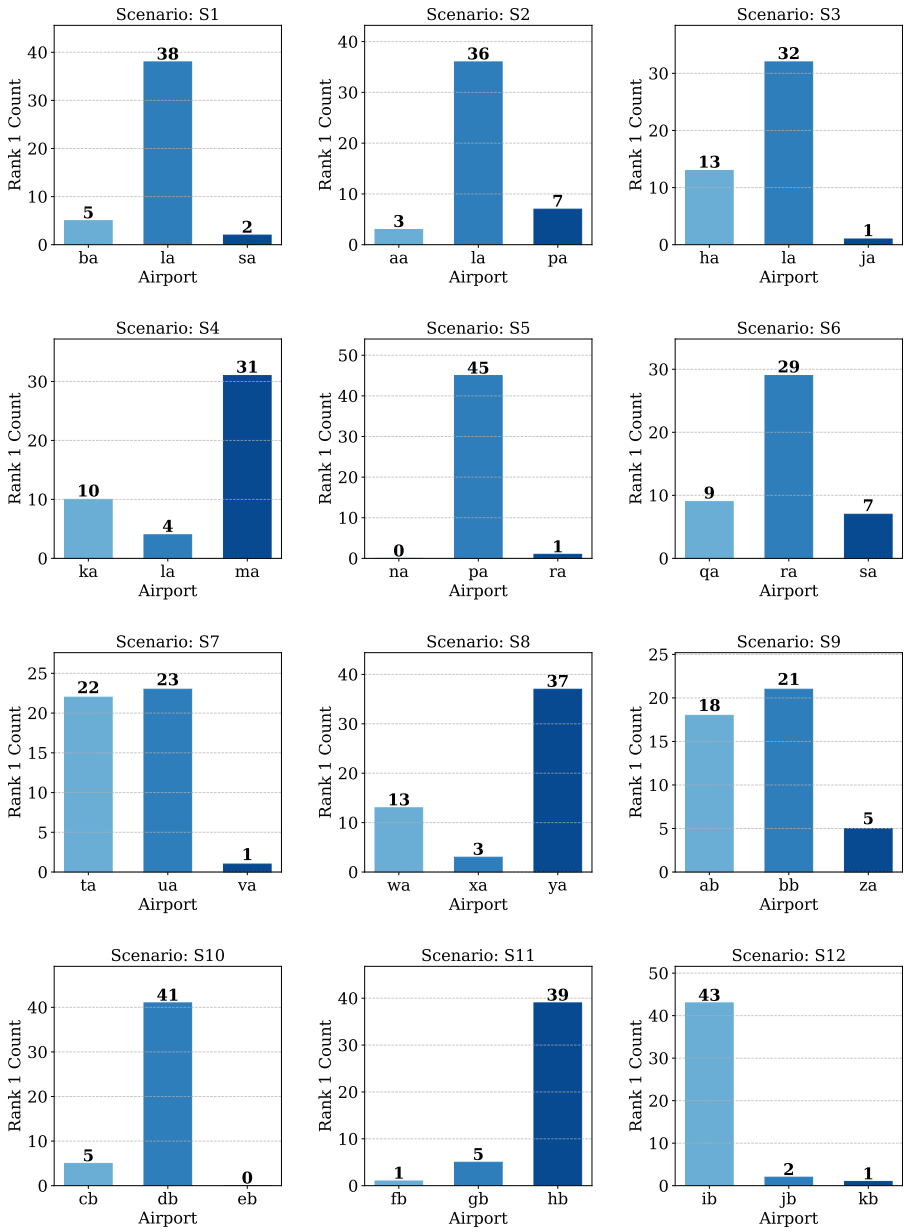

Figure 4.9: Distribution of pilots' top airport selections, adapted from [80]

first choice. This approach is especially important since Figure 4.9 indicates that for certain scenarios, there is no clearly dominant choice.

Once the new scores were calculated, Pearson correlation coefficients were determined for each scenario and then averaged. To ensure that these correlation values were suitable for further analysis, Fisher's z-transformation [99] was applied. This transformation adjusts the variance and normalizes the distribution, which makes the results more robust. This method allows the possibility of evaluating the impact of individual features across all scenarios, as shown in Figure 4.10.

The results demonstrate that the Marginal Landing Distance and Related Landing Performance are the strongest factors influencing pilots' decisions, with correlations approaching 1. This outcome is consistent with operational reasoning, given that having an adequate landing distance is critical to avoiding runway excursions. A runway excursion is an incident in which an aircraft unintentionally leaves the runway during takeoff or landing. This can happen for various reasons, including challenging weather conditions, pilot error, or technical issues. Such incidents can involve the aircraft overrunning the end of the runway, veering off the side, or completely leaving the runway surface.

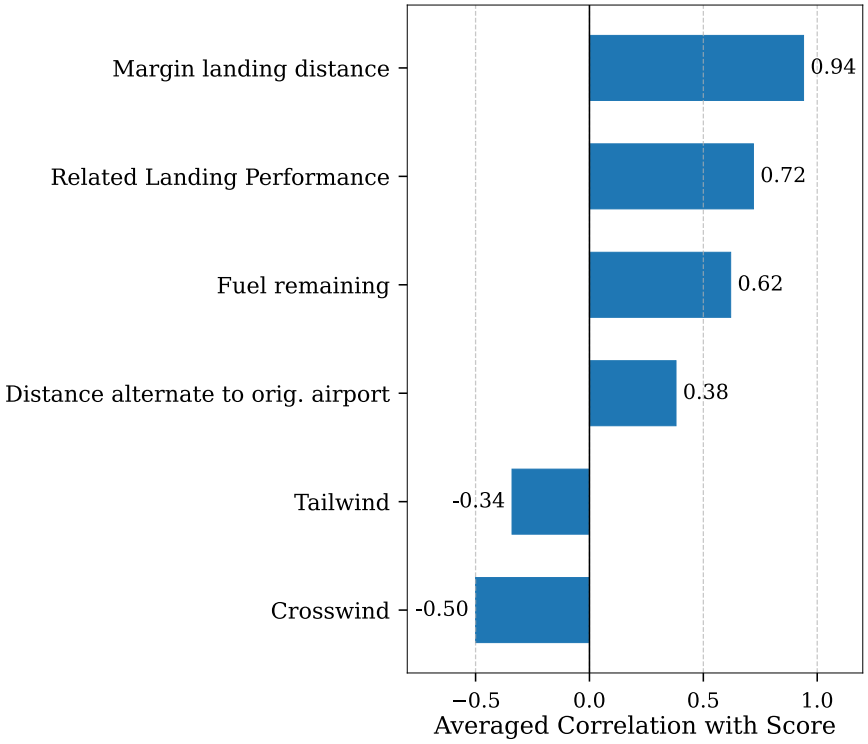

Figure 4.10: Average feature correlations, adapted from [80]

These situations pose safety risks and often lead to further investigation to identify causes and improve procedures. The strong correlation observed for Related Landing Performance is also logical, as runway surface conditions directly affect the aircraft's braking effectiveness, which plays a significant role in preventing excursions.

Another interesting finding is that both crosswind and tailwind factors are negatively correlated with the preferred airport choices, with crosswind having a greater influence than tailwind. It was initially expected that tailwind would have a larger impact because strong tailwinds can make landing more difficult or unsafe under certain conditions. However, the results suggest that crosswind conditions are more strongly linked to pilots' decisions than previously assumed, showing the complex relationship between wind factors and safe landing performance.

4.3.3 Training Dataset Generation

Having developed a clearer picture of the dataset's structure, the next step is to consider how it can be leveraged to train the model. The scenarios assessed by the pilots were intended to approximate real-world decision-making tasks, though in a simplified format. Due to practical constraints related to time and resources, only a limited number of scenarios could be directly evaluated. However, relying exclusively on such a small, narrowly defined dataset may limit the model's ability to generalize effectively. To ensure the model can handle a wider variety of decision situations, it should be trained on data that extends beyond the original set of scenarios. To address this, a larger training dataset was developed by generating additional scenarios. This involved sampling from the pool of available airports and their characteristics and combining these into new sets of three-airport scenarios. Care was taken to avoid duplicate airports within each generated scenario. Through this process, and given that $12 \times 3 = 36$ unique airports were available, a total of 988 new scenarios were created. These were then combined with the initial 12 scenarios, resulting in a final training dataset of 1,000 scenarios.

Unlike the original evaluated scenarios, this expanded set initially contained only unlabeled data. In this context, "unlabeled" means that no ground-truth ranking or correct choice had been assigned to these new scenarios. The original idea was to treat this as a reinforcement learning problem, where an agent learns optimal behavior through iterative feedback in the form of rewards and penalties. However, since this task is a single-step problem—meaning the agent needs to make only one decision per scenario—it was more practical to convert the dataset into a labeled format.

This labeling was achieved by applying a weighted similarity measure to each airport option within the new scenarios. The evaluation score for a given option a in scenario S was defined as:

$$EV(a_i, S_n) = \frac{1}{d_{min}(\vec{x}, \vec{b_j})} \times SC(a_i) \tag{4.9}$$

where, n represents the index of the scenario, while $d_{min}(\vec{x}, \vec{b_j})$ denotes the minimum Euclidean distance between the current input scenario $\vec{S}$ and $\vec{b_j}$, which is the jth row in the reference matrix b. This reference matrix is composed of the original twelve scenarios, each flattened so that one scenario corresponds to a single row.

This approach enables a comparison of the new, unevaluated scenarios with the reference scenarios based on similarity. Each action (airport choice) in the new scenario is scored using the original scenarios' ranking scores, adjusted by their similarity. The more similar a generated scenario is to an existing evaluated one, the more its ranking should align with the original pilot responses. This is quite similar to the case-based approach employed in [164]. Using this method, each new scenario was assigned a "correct" label, which is simply the airport with the highest evaluation score. In this way, the initially unlabeled dataset is transformed into a fully labeled training set, ready for use in model training.

4.3.4 Result analysis

The experiments were implemented in Python version 3.9.6 using the XCSF Python package [179]. Due to the large number of hyperparameters that need to be tuned LCS, hyperparameter tuning was performed with a Bayesian optimization approach using the Optuna tuning library [14]. The tuning process employed twenty initial random samples and fifty additional evaluations. To account for the model's stochastic behavior, each parameter configuration was evaluated over thirty-one independent runs, using 10% of the dataset for validation. The weighted F1 score served as the primary metric for selecting the best hyperparameter combination. The LCS performance was compared to that of a random agent. The random agent randomly selects from one of the three available classes for its prediction. The comparison can be seen on Figure 4.11. Upon examining the figure, we an observe that the average accuracy of the LCS lies somewhere between 0.55 and 0.6, which would indicate that it obtains the correct answer approximately 60% of the time. Although the LCS has a wider distribution accuracy values, even at its worst, it would still outperform the random agent. Despite being able to outperform a random agent, the LCS still has a mildly impressive accuracy.

4.3.5 Lessons Learned

This paper explored how data could be gathered directly from decision-makers and later used to train an interpretable AI model to support other future decision-makers in MCDM scenarios. Although the results were not entirely in line with the initial expectations, the exploration highlighted two important points that should be taken into account.

Preference inconsistency

One of the more apparent observations from the paper was the inconsistency between responses for certain scenarios. It is worth noting that there were no initial expectations set before the investigation regarding the degree of variation

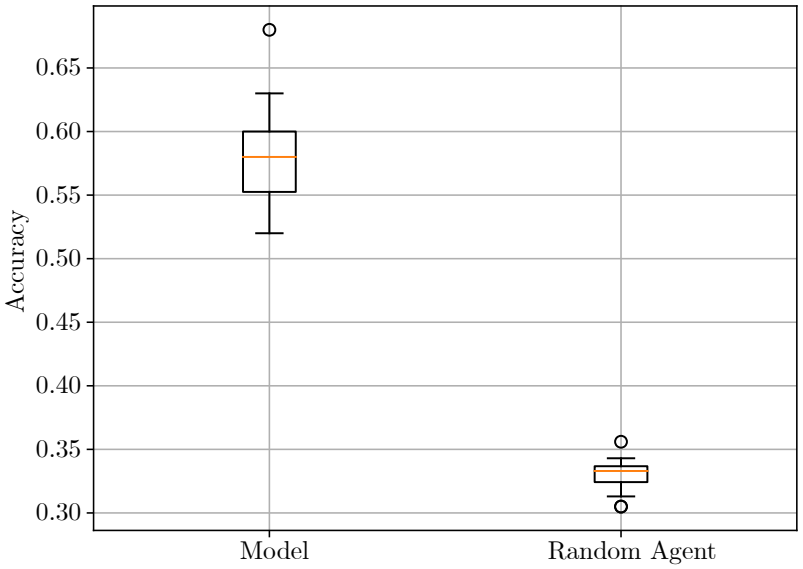

Figure 4.11: LCS accuracy versus random agent accuracy taken from [80]

in the answers; however, the results were surprising. This outcome was understandable, as in a complex MCDM scenario like the DAAS problem, different aspects are likely to be prioritized by different decision-makers, especially when they are asked to evaluate scenarios individually rather than as part of a group. This highlights some of the inherent challenges of using data collected as an amalgamation of many decision-makers rather than from a single, consistent decision-making process. It also further reinforces the difficulties associated with data collection, given that having groups of experts treat each scenario as its own MCDM problem is often too costly and time-consuming.

4.4 Expert considerations

The following is largely based on the author's paper "Through the Psychological Lens: Unveiling Biases in Multi-Criteria Decision-Making" [79].

4.4.1 Main Idea

Inspired by the previous revelations about how decision-makers may differ when confronted with an MCDM problem, a more detailed investigation was performed to more deeply explore the certain psychological considerations that one should be aware of for MCDM problems involving human decision-makers.

4.4.2 Preference expression and logical consistency

Preference expression

In the paper [79], a greater focus was placed on MADM methods, which focus more on selecting from a finite set of options and are more in line with the topics of this thesis and the practical DAAS problem. One of the key considerations when it comes to classical methods hybrids that use of AHP style factor pairwise compressions methods is their preference-elicitation basis. However, the concept of preferences itself often remains ill-defined. Depending on the context, preference can refer to the perceived importance to stakeholders, a decision-maker's personal valuation, or simply a choice between alternatives [152]. This lack of clarity complicates measurement, as different interpretations may call for different instruments, yet everyday language frequently blurs these distinctions. As a result, the validity of preference measures can be questioned, as demonstrated by research highlighting that different methods used

to assess attribute importance often yield inconsistent results, implying that they capture different constructs [263].

Logical inconsistencies

Another significant challenge related to the consistency of judgments expressed by decision-makers. Even when preferences appear logically transitive, real-world assessments often reveal contradictions. For example, a decision-maker may prefer option A over B and B over C but then rank C over A, violating the basic transitivity rule. Such inconsistencies can arise from the inherent variability in how individuals discriminate between alternatives, which is why Thurstone's law of comparative judgment proposes repeated evaluations and mean preference frequencies to construct a coherent ranking [253]. Despite such frameworks, logical inconsistencies still occur due to factors such as irrelevant alternatives, uncertainty, or structural dependencies. A body of work has attempted to address the logical inconsistencies in preference matrices. Papers such as those by Bozóki et al. [40] treat the issue as an optimization problem, where the goal is to obtain a perfectly consistent preference matrix or one that is as close to consistency as possible.

4.4.3 Expert and expertise

What constitutes an experts

Many of the MCDM and MADM methods rely on the experience of experts to provide preferences and rankings that ultimately guide the decision-making process. However, there is no universal agreement on what defines an expert in the first place [273, 67]. Defining expertise is inherently complex and varies greatly across disciplines [122, 262]. Even when a decision-maker has expertise, it remains uncertain whether their judgments consistently outperform those of non-experts. One must also consider the biases that experts can exhibit, such as outcome bias, which experienced professionals like pilots are not immune to [183]. The reliability of an expert's judgment heavily depends on the decision context. If future outcomes can be reliably forecast using past information, expert judgments tend to be more accurate. Otherwise, non-experts may perform just as well [124, 110]. Interestingly, research shows that experts do not always use more information than non-experts; instead, they tend to weigh the criteria differently [237]. This suggests that using an extensive list of criteria is not always necessary to benefit from an expert's knowledge. Experts seem to be much easier to identify in fields with many repetitions where feedback on their performance can be provided within a short time, such as in chess games. The short duration, clear win conditions, and closed system of chess make it an ideal context for developing expertise in the game.

Expert knowledge embedding

An equally important consideration is whether the mechanisms employed in various MCDM methods can genuinely capture and use the experience of the decision-maker. Harries and Harvey [115] argue that participants' responses often reflect how they believe the criteria should have been applied rather than how they were actually used in practice. This raises doubts about the extent to which MCDM techniques can effectively draw on accumulated experience. However, Riquelme [218] showed that people can articulate their judgment policies. In an experiment, participants asserted a holistic judgment on whether they would buy a mobile phone plan and then indicated how they weighted the criteria they had considered. The comparison between weights derived from the holistic assessments and the explicit weights provided by participants suggested that people can reliably describe their own judgment policies. On the other hand, some MCDM and MADM methods do not even claim to incorporate the experience of the decision-maker but instead focus on reflecting the interests and priorities of stakeholders.

4.4.4 Lessons Learned

The key takeaway is that there are serious considerations when employing many of the traditional MCDM mechanisms and that one should exercise caution regarding who the group of experts is. To relate this to the DAAS problem, it helps explain the results obtained from the data generation step but also raises

some very important questions. Namely, can pilots or broader stakeholders truly provide reasonable expertise for all emergency situations? Given that emergencies do not occur frequently and are not part of typical operational procedures, do the experts have enough relevant experience to be useful? Generally speaking, we cannot always obtain the ideal data or the perfect group of experts, so we should attempt to make do with what is available, but it is important to keep these considerations in mind and make adjustments to try to address them. Simply stating that this is all the data there is or that these are the only experts available should not dismiss the very real issues that could arise from it.

4.5 Reinforcement learning-based approach

The following is largely based on the author's paper "A Learning Classifier System Approach to Time-Critical Decision-Making in Dynamic Alternate Airport Selection" [78].

4.5.1 Main Idea

Having examined the use of experts to directly gather data and employing a more classical approach with the weighted-sum approach, a reinforcement learning approach was attempted. The approach is undertaken as a scenario-based formulation in which every scenario also contains information about the context, i.e., the lens through which the options are viewed lends itself well to ideas from reinforcement learning. The lack of data, along with some of the difficulties and problems apparent in data-gathering methods, leads to the investigation of how reinforcement learning could be used—allowing the model to learn through trial and error to make the best decisions instead of trying to simply emulate decision-makers. The idea was to reformulate the problem as a contextual bandit problem. The inclusion of a context aspect that colors the decision-making process was evident in both, and the single-step nature of the contextual bandits aligns well with the more decisive and key MCDM decision-making focus of the DAAS problem.

4.5.2 Implementation

Learning Environment

For the reinforcement learning approach, the 2,600 scenarios generated for the bi-objective approach were employed, where each scenario was a snapshot of a situation in which a diversion decision must be made. Twelve airports in Germany were considered, one of which is always unavailable. Each airport was characterized by the following attributes:

- Longest runway length available $[m]$.
- Crosswind speed at the location of the airport $[kn]$ 4.2.
- Tailwind speed at the location of the airport $[kn]$ 4.2.
- Fuel needed to reach the airport $[kg]$.
- Status of the runway, expressed as a value from 0 to 6 4.1.
- Distance from current aircraft position to planned destination $[nm]$.
- Distance from current aircraft position to appropriate alternate airport $[nm]$.

Utilizing random Latin hypercube sampling, 100 variable combinations for wind and runway conditions were sampled, along with 26 flight plans for realistic airport positioning in the sky, resulting in a dataset represented as a tensor $D_{k,n,m}$.

Objective function

Employing reinforcement learning necessitates an objective function that would

provide evaluations on how the chosen recommendations by the model. This is one of the disadvantages regarding reinforcement learning methods as they tend to be problem-specific and somewhat of an art form to get right. Taking the objective functions from the bi-objective approach, we can obtain groups of objectives according to physical safety considerations and financial costs, expressed as:

$$Risk_{avg}(\vec{w}) = \frac{1}{k}\sum_{i=1}^{m} Risk_i \tag{4.10}$$

$$Cost_{avg}(\vec{w}) = \frac{1}{k}\sum_{i=1}^{m} Cost_i \tag{4.11}$$

where the Risk was calculated as:

$$Risk = \frac{1}{(ld_{have} - ld_{need})} + cw + \frac{1}{fr} \tag{4.12}$$

However, for various critical situations, the risk was calculated as follows:

$$Risk = \begin{cases} 12 & \text{if} \quad cw > 38kn \\ 15 & \text{if} \quad ld_{need} > ld_{have} \\ 20 & \text{if} \quad f_r < 1600kg \\ 25 & \text{if} \quad ld_{need} > ld_{have} \ \& \ cw > 38kn \\ 30 & \text{if} \quad ld_{need} > ld_{have} \ \& \ f_r < 1600kg \\ 35 & \text{if} \quad ld_{need} > ld_{have}, f_r < 1600kg \ \& \ cw > 38kn \end{cases} \tag{4.13}$$

and the cost was calculated as:

$$Cost = (d_{planned} - d_{actual}) * cf \pm f_u \tag{4.14}$$

The objective functions were very suitable for a very narrow use case, such as that of the A321 Airbus aircraft, and were limited in the number of factors they took into account. Additionally, the bi-objective approach always took risk into account given that it always calculated the runway length needed and the crosswind. However, many of these factors had hard constraints where if the runway was sufficiently long, it made no sense to take it into further consideration. Now, crosswind is always undesirable, and its best value would be 0; however, there are also technically defined and legal limits for it, so one could make the same argument for it as well.

The new objective devised was a single objective that essentially combined the previous objective function. The function that provided the reward for the model was represented as:

$$R(\vec{x}) = \frac{1}{\sum_{i=1}^{n} \mathbf{b}_i * c + \sum_{j=1}^{m} \mathbf{u}_j} \tag{4.15}$$

Here, $\vec{x}$ denotes the input scenario unrolled as a vector, ordered by the context and then the features of each options one after the other. The reward was defined in terms of bounded and unbounded attributes, more comonly expressed as attributes with certain constraints associated with them that must be satisfied and unconstrained attributes that should be optimized as much as possible. The values b_i and u_j represent the ith and jth elements within the bounded and unbounded dimensions, respectively. The vector $\vec{b}$ collects all bounded attributes, which correspond directly to the active constraints. If any constraints are violated, the magnitude of the deviation from the permissible value is recorded within $\vec{b}$, meaning its cardinality variable is dependent on the

number of constraints breached. The scaling factor $c = 1 + |\vec{b}|$ thus equals the count of constraint breaches plus one. In parallel, $\vec{u}$ represents all attributes without hard constraints that should ideally be optimized.

This reward function formulation is intended to address some of the shortcomings identified in the bi-objective formulation of risk and cost. It maintains an emphasis on integrating both physical risk elements and economic considerations. This revised structure makes it possible to add as many factors as one like; thus, thus, for the DAAS problem, greater flexibility is granted to airlines or aircraft manufacturers. Additionally, this form of reward is robust across a broad spectrum of conditions. For example, in scenarios in which most candidate airports are unsuitable from a safety perspective, the bounded term will dominate the equation, pushing the model to select actions that limit safety risks as much as possible. The inclusion of the factor c achieves two aims: it amplifies the influence of constraint violations, reinforcing the priority of safety, and it echoes the "Swiss Cheese" model of accident causation [90], which recognizes that multiple small deviations can collectively lead to undesirable outcomes. As a result, the system applies stronger penalties to choices with multiple minor breaches, helping mitigate compounded risk factors. On the other hand, should there be a decision-making scenario in which safety constraints are satisfied, the model would learn to prioritize optimization of the economic factors.

Ultimately, this design seeks to reduce the likelihood of mission failure by balancing operational safety with cost considerations. Conceptually, the objective is to reduce potential hazards and burdens for pilots. Practically, given that the original formulation was a minimization objective, it was inverted into a maximization task by obtaining its reciprocal. Additionally, for clarification, all the factors and their respective constraints were normalized from 0 to 1. In the current setup, four main constraints were imposed:

- Required landing distance must be less than or equal to the available landing distance.
- Fuel required to reach the airport must not exceed fuel on board.
- Tailwind at the chosen airport must not exceed 15 kn.
- Crosswind at the chosen airport must not exceed 30 kn.

Finally, the distance between the original destination and the diversion airport, along with the fuel required to reach it, should both be minimized. Wind conditions and runway status are integral for calculating the required landing distance.

Regarding the objective function formulation, we also note that it needs information from experts to provide the constraints, but can also take on variables as constraints. This is best showcased using the required landing distance and the fuel needed, which are variable and depend on the scenario itself, requiring the algorithm to also take into account more variable constraints that are not specifically encoded as values in the objective function itself.

4.5.3 Results analysis

The model used was an LCS that uses hyperrectangle conditions, linear approximation, and integer actions. The majority of the structure is based on [49] and makes use of the enhanced Michigan-style learning classifier, often called XCSF [276] [277]. The LCS was employed due its rule-based structure, which seemed to align better with explainability, and its close association with the multiplexer problem, which has a similar single-step classification, context-dependent nature as the DAAS problem. The results are shown in Figures 4.12 and 4.13, which depict the average reward and the corresponding number of broken implications, averaged across 31 runs of the algorithm. As can be

seen, the algorithm does indeed learn to make better choices and subsequently reduces the average number of broken implications. The decrease in broken implications near the beginning, followed by the plateau toward the end, suggests that the model correctly identified this as the first priority.

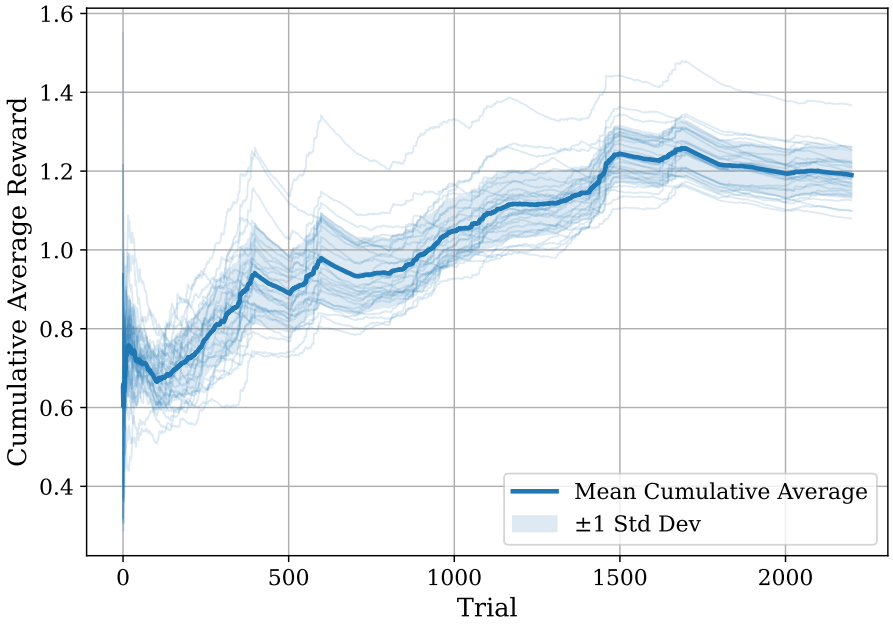

Figure 4.12: Reward Improvement Trend Over Trial, adapted from [78]

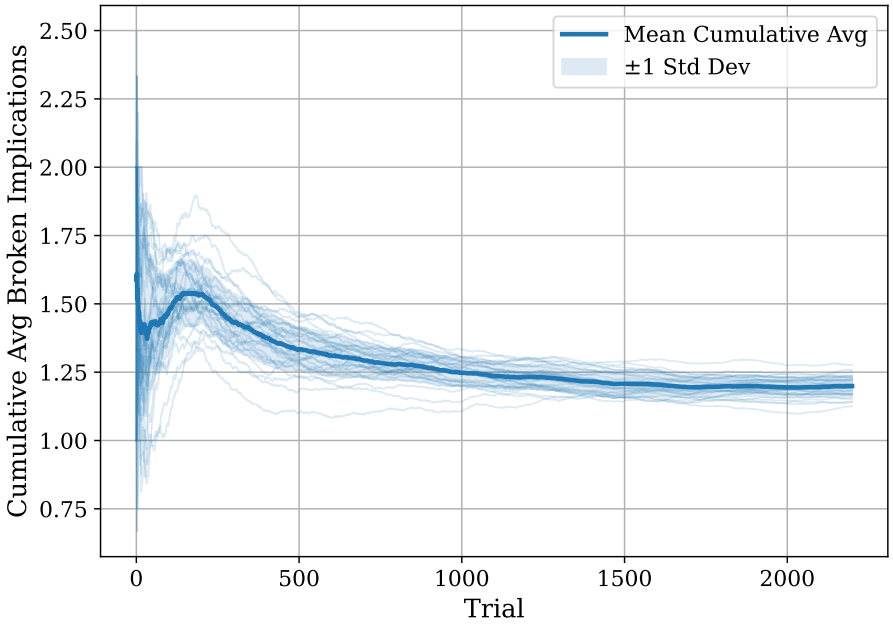

Figure 4.13: Cumulative broken applications,a dapted from [78]

Here an important point to consider is variability in the factors—namely, many MCDM methods assume that the values presented are fixed and not subject to change. However, in the real world, significantly more variability exists. Taking the DAAS problem as an example, we note that weather phenomena cannot be entirely accurately defined, with the system observing estimations for the weather condition for each option. This could be subject to change depending on the volatility of weather—in a storm, it is expected that wind and weather phenomena would behave more erratically. For the purposes of the experiments, it was assumed that the approximations are indeed exact and variability would be considered another time.

4.5.4 Lessons learned

The paper explored how to tackle an emergency MCDM problem by employing an explainable LCS model and converting the problem to a reinforcement-learning classification problem.

Encoding domain experts knowledge

The key insight gathered here was the use of an objective function in order to encode an expert's knowledge. Specifically, the presence of factors that need to be satisfied for the safety aspect lends itself well to constraint-based approaches, in which the decision-makers should be focused on providing the constraints and determining how to deal with them rather than just comparing alternatives. This focus on encoding knowledge in an objective, constraint-based function, in addition to dealing with some of the previous issues outlined in the bi-objective problem, serves as a much easier way for experts and stakeholders to provide estimates for options and decisions that can span many decision-making scenarios. While it may be more difficult than simply specifying which factor or alternative they prefer in a pairwise comparison fashion, it offers additional transparency and rigidity in the way their knowledge and insight is expressed, which is valuable for transparency and retrospective analysis.

Implementation considerations

The initial exploration of transforming the problem into a reinforcement-based multi-class classification problem, in a similar vein to the multiplexer problem, was not without faults. The understanding and grouping of factors based on their impact on the decision presents as a useful idea; however, the objective function formulation and its adaptation into a reward function were performed in a very cumbersome manner. Additionally, the idea was for the LCS to treat the problem as a more difficult, real-valued version of the multiplexer problem due to certain identified similarities. However, the reward signal was not normalized properly and came from an objective function with sharp constraints. A more appropriate formulation would have been to use the objective function to, in essence, identify the best option and provide a 1 or 0, similar to the process in the multiplexer problem. This labeling of the dataset leads into the next consideration of variability. The scenario generation and model learning cannot be performed on static scenarios and must take into account the variability of the factors, allowing the model to learn how to make the best decision under those variable conditions, balancing trade-offs and trying to minimize constraint violations while maximizing other factors.

4.6 Summary of this Chapter

This chapter presented a comprehensive overview of the research trajectory and key insights gathered from the author's publications related to MCDM in time-critical emergency applications. It traced the evolution of methods and ideas employed to overcome the inherent limitations of traditional MCDM techniques when applied to highly uncertain, high-stakes, and time-constrained contexts such as the DAAS problem.

A recurring theme emerging from this investigation is the recognition that classical MCDM frameworks are not particularly well suited to emergency decision-making with time considerations and in stressful situations. Another important characteristic identified, which stemmed from a real-world problem, is the occurence of a situation in which the decision-makers are embedded in the situation and will directly experience the outcome of their decision. This is somewhat different from typical MCDM methods and those applied to emergency management in which the stakeholders, although responsible for the outcome, seldom seem to bear the full implication of their choices directly. Thus, the identified need for support systems that can replicate expert-like reasoning under these conditions prompted a shift towards hybrid AI–MCDM approaches.

The investigation emphasized the challenges of data availability in emergency settings, noting that clean, structured datasets are rare. This necessitated the

use of scenario-based synthetic data generation, guided by expert input and operational knowledge. The investigation attempted a more classical weighted-sum approach based on domain knowledge, which was found lacking regarding adaptability. Additionally, an attempt was made to gather data directly from prospective decision-makers and stakeholders, specifically pilots who were presented with various DAAS problem scenarios for use in training AI-MCDM models. These efforts highlighted the inherent variability and inconsistency in human responses, especially when scenarios are presented in isolation and without group deliberation. This also showcased the difficulty of using data gathered in this manner to train MCDM-AI hybrid models.

A key insight gained from the studies is the centrality of constraints in emergency MCDM problems. Rather than treating all factors equally, those related to safety were identified as constraints necessitating satisfaction, while others served as optimization goals. This naturally led to the use of constraint-based formulations in which decision-making is framed as the identification of viable options under hard and soft constraints. It was found that in such contexts, encoding expert knowledge through an objective function was both practical and transparent, allowing domain experts to specify rules rather than articulate fuzzy preferences or weights. This, along with the investigation of explainable rule-based models such as the LCS, seemed to be a useful direction for further exploration.

In summary, this chapter showcased how the research evolved from a bi-objective approach more in line with traditional MCDM formulations toward a learning-based, constraint-driven AI-MCDM hybrid approach. It showcased how a time-critical emergency MCDM problem could be reformulated as an optimization problem more suitable for AI models, which could then be used in a decision-support system.

5 Explainable Adaptive, Context-aware Time-critical MCDM - ExACT-MCDM

This chapter presents the methodology specifically developed to address several challenges associated with time-critical decision-making in emergency applications. By combining a wide range of context-aware, synthetically generated or data-augmented decision scenarios with domain knowledge encoded through a structured labeling function in a weak supervision fashion, the methodology enables the transformation of complex MCDM problems into a supervised learning task. This facilitates the training of various AI models, which can then be deployed as core components within an intelligent decision-support system. The proposed methodology also aims to provide greater transparency, explainability, and interpretability due to its complex nature and use of AI models. It is titled Explainable, Adaptive, Context-aware, Time-critical Multi-Criteria Decision-Making (ExACT-MCDM) and is designed for atypical, high-stakes decision environments in which important and decisive actions need to be taken.

5.1 Main challenges and ideas

The methodology is based on the assumption that an Artificial Intelligence–Multi-Criteria Decision-Making (AI-MCDM) hybrid approach used as part of a DSS can provide suitable assistance in decision-makers embedded in these time-critical emergency situations. The use of AI models occurs, in part, due to their ability to generalize, handle a wide array of possible decision-making scenarios, and examine the extent to which AI-based models are useful in applications of time-sensitive emergency MCDM.

5.1.1 Emergencies and Data Scarcity

The main difficulty when using contemporary machine learning methods is regarding data availability. Should a suitable, well-defined, and distributed dataset exist, one can simply train a model, employ hyperparameter tuning, and determine its incorporation into a DSS system. However, one of the most difficult problems is procurement of this data and assuring that it is of sufficient quality to be used in this manner. The data availability problems typically result from two characteristics.

Emergency scenarios, such as those found in the DAAS problem, are inher-

ently edge cases and fall outside the scope of normal operations. As a result, their occurrence is rare, poorly distributed, and heavily biased toward more typical cases, making it difficult to gather a well-representative dataset. This is particularly problematic, as the need for decision support increases precisely when the decision-making scenario deviates from standard procedures and the experience of the decision-makers. This necessitates that the model be exposed to as many edge cases as possible in order to provide assistance at the moment it is needed most.

Traditional labeling approaches for decision data in emergency MCDM settings present another set of challenges. The process is not only time-consuming and dependent on expert availability but is also not a simple labeling task, as each input that needs to be labeled is its own MCDM problem requiring contemplation and careful consideration, if not the outright use of traditional MCDM methods. This is additionally compounded by the diverse, edge-case nature of the decision-making problems, in which the expert involved may not have sufficient experience to properly assess the situation or recommend an appropriate course of action. As a result, conventional labeling pipelines are not practical or well suited.

5.1.2 Components

The ExACT-MCDM methodology makes use of three key components to overcome the data availability issues and provide a suitable dataset for AI models.

1. Scenario-based data.
   In order to train the models, scenario-based generated data is used. This can encompass a wider range of possible decision-making scenarios, including the most difficult and extreme ones. The use of generated data to overcome data limitation problems is not especially radical and has been employed in much of the literature, particularly in the Aerospace context, as shown in [232] and [139]. An important point to consider here is that the focus should be on providing suitable inputs for the model to learn from, inputs that accurately reflect those it will encounter when in use in the DSS. Essentially, many real-world problems are unlikely to have suitable representative datasets readily available on which to train; rather, the data need to be adapted and tailored by aggregating different data sources, consulting experts, and using a variety of sampling techniques based on relevant information. The scenario-based approach allows for addressing the data limitations and exploring scenarios of interest.

2. Weak supervision-based labeling mechanism.
   Although the scenario-based synthetic data generation approach is able to theoretically provide a rich and diverse dataset, there is the problem of providing labels or evaluation for possible decisions within the scenarios themselves. Typically, despite MCDM processes', their evaluations tend focus on cyclical refinement, initially engages with a mechanism to produce to be a priori-focused, where one engages initially in a mechanism for producing evaluations, be it direct weight assignment or something like AHP in which weights are derived. On the other hand, the idea proposed here is to borrow from reinforcement learning, as well as strategies such as the more well-known Best–Worst Method and TOPSIS. Essentially, the goal is ascertain how good the outcome is after the fact, when a decision has been made; in essence, focusing more on how to evaluate a specific choice than determining how to weight and prioritize its factors. Although subtle in nature, this shift in thinking is important. Following from this, and inspired by methods of distance learning, an evaluation function is used, similar to that proposed in Chapter 4.5, which can incorporate soft and hard constraints, as well as other domain knowledge. Then, using this evaluation function, scores for each of the options can be ascertained and, thus, a labeled dataset can be derived. Using this

approach, domain experts can impart their expertise and label a large dataset without having to engage in labeling each option one by one.

3. Scenario simulation robustness
   An important aspect occasionally overlooked in MCDM problems is robustness. Given the variability of aspects such as wind in DAAS problems, as well as general variability common in many real-world problems, the incorporation of robustness and variance in the factors must be considered. To this end, the idea is to incorporate additional variability in the factors so as to reflect the real world on a scenario basis. For example, in the DAAS problem, this would mean varying the wind and weather phenomena accordingly to simulate real possible deviations in the scenario, whilst not varying distances as airport positions do not vary. Incorporating scenario-based variability of the factors along with the labeling function for each option in a scenario allows n number of evaluations based on the number of perturbations or variabilities employed. In this way, we incorporate robustness into the final labeling of the dataset.

Core idea The core idea is thus to obtain scenarios which are their very own MCDM problems, derive an evaluation function in order to score the options, and incorporate robustness via controlled perturbations and simulation on a per-scenario level. This would then be used to train an AI model. The AI model acts as a surrogate and can be called up instantly in an emergency without the need to use traditional methods, or, if this approach is considered, can bypass the need for time-consuming simulations and with varying simulation times depending on the use case. For example, for the DAAS problem, weather simulation and stepwise simulation are needed to determine factor values, which can be rather slow and vary in duration given the stepwise nature, as some scenarios may require more simulation if, for example, airports are further away and more time is needed to simulate the flight to them. A generalizable ML model can simply obtain the best estimates for each of the values and, based on what it has learned, provide a course of action almost instantaneously, which is important in time-critical and stressful situations where each second counts. Essentially, this approach can be likened to casting a huge net—the strings are the scenarios generated and considered and the AI models function as a mechanism to fill in the gaps in the net.

5.2 ExACT-MCDM methodology description

Having described the three key components, this section will focus on showcasing how they fit together to build a cohesive and usable whole. It is important to specify the distinction between the various types of stakeholders involved in the process. Essentially, stakeholders can be divided into interested parties and decision makers. Taking the DAAS problem as an example, interested parties would include airlines, aircraft manufacturers, the DSS designers, airport management, and crews. This subset of stakeholders is primarily engaged in the development stage of the DSS, i.e., determining the objective function, scenarios considered, evaluation criteria, etc. The decision makers represent the final decision makers and, in most cases, the responsibility holders for the decision—the pilots in the DAAS problem.

These two groups have somewhat different priorities and needs. Explainability is needed in both groups but varies in extent. The interested parties require a greater focus on more general evaluations, a clearer understanding of the model's decision capabilities, and clarity on how experts embed their domain knowledge. The final decision makers, on the other hand, want fast recommendations and a good breakdown of the recommendation on a single case basis, namely the current scenario they are in. Given the focus of this PhD on AI models and their integration into the DSS, the ExACT-MCDM method is prioritized, regarding how one can ingrain domain knowledge, address the data

scarcity issue, and train suitable AI models.

ExACT methodology steps

The ExACT-MCDM methodology is envisioned as a cyclical process consisting of the aforementioned key ideas brought together in the following way:

1. Problem description
   As in many MCDM methods, the initial step is to determine the problem and its scope. Here, the factors considered need to be taken into account alongside the experts involved and the planning of the next steps. This is the preparatory step and should encompass planning and taking into account all actions needed for the execution of the following steps. Key in this stage is defining the problem and scope as much as possible, determining the number of options considered, defining the relevant factors per option,and exploring specific contextual information that might affect the decision in a given scenario.

2. Scenario generation
   This is a key step in which the possible breadth and depth of the decision-making scenarios need to be considered and generated. Key in this step is determining how many scenarios will be generated, examining the possible cases they would encompass, and providing a detailed definition of the simulation and/or sampling methods used to derive the scenarios.

   Equally important is that scenarios not only include the decision options and their associated factors but also the broader context or viewpoint through which they are evaluated. In this sense, the construction of scenarios follows a logic similar to that of contextual bandits, where both the options and the situational context jointly define the decision space. This is a pivotal step as it amounts to selecting and generating the dataset from which the models will learn. Great care should be taken to ensure that the generated scenarios align with the goals of the stakeholders and provide ample instances that are of real interest. It is important to, again, highlight that generating the decision-making scenarios is based on expertise, data analysis, and already-available data, which is transformed and augmented to serve as training input for later model training.

   Generating the scenarios and subsequent dataset is therefore one of the initial steps in encoding domain knowledge and focus, along with additional information ideally gathered from extensive data analysis, experience, and educated guesses. One should then consider is the cost per scenario generated and the trade-offs between detail and accuracy compared to producing a larger number of lower-fidelity scenarios. Using the example of the DAAS problem, this could reflect the level of detail one applies to weather modeling and fuel calculation, as these play influential roles in the final decision-making process and can be quite expensive. Therefore, one also needs to consider the trade-off between scenario accuracy and cost per scenario generation.

3. Evaluation function creation
   The evaluation function is the most direct method of encoding expert domain knowledge. The idea is to use this as a proxy for the group of experts labeling each scenario separately.

   It must be noted that, as the approach is inspired by the DAAS problem and the lack of suitable methods, the assumption is that one would be dealing with real-valued factors or a mixed arrangement consisting of real values and categorical or ordinal values. This does not imply that all factors must be real values. Factors might also take categorical values, and this method becomes useful when the number of possible combinations is astonishingly large. In short, it would be applied when simply producing a small number of scenarios is not sufficient for the scale of the problem. Hence, the evaluation function provides a score for each option, should

it be chosen. The evaluation function is, in a sense, the most important and difficult aspect of the methodology.

Although the involved experts may choose to craft whatever function they deem appropriate, a suggested hierarchical and constraint-aware evaluation function is presented, inspired by and used to tackle the DAAS problem, which can also be employed in other fields with similar requirements. The goal was to provide a technique akin to a flexible function capable of being adapted to a wide array of problems and further augmented depending on the needs of the problem. The idea was also inspired by the notion of having a go-to loss function, like RMSE, that could be both used out of the box and modified to fit one's needs.

4. Scenario simulation and option estimation.
   The goal of this step is to introduce variations in the factors for each option in every scenario. This is performed to incorporate uncertainty and to enable a better, more robust evaluation of the options in each scenario. Within the DAAS problem, this would involve introducing changes to wind and wind direction, as well as other weather phenomena that cannot be exactly estimated upon arrival at the chosen airport, but for which we only have the best estimates. However, like weather, certain factors may be unpredictable, and it is therefore necessary to take their variability into account when evaluating the options.

   An additional point to consider is that the variations/perturbations should not simply be the random introduction of noise. Rather, they should reflect real changes that would occur and, if necessary, be re-simulated. To harken back to the DAAS problem, this would involve not only varying the weather accordingly but also re-simulating the fuel requirements and runway usage, as these are influenced by weather. Thus, perturbations should not be performed carelessly and uniformly across all factors but should instead be conducted in a manner that reflects reality.

   Having incorporated realistic perturbations, the previously defined labeling function can be used to obtain an estimate for each of the options in the scenario. To illustrate it simply, variations would be introduced from the original scenario; then, using the labeling function, all options would be assigned their score for this set of variations. Continuing this pattern, scenarios would be varied accordingly and new scores for each variation run would be produced. The final scoring of the options would involve using their scores to produce one aggregate evaluation for the option, which can then be used to derive the best action and, in so doing, label the scenario. The aggregation can be performed using a more naive approach such as averaging. However, a slightly more sophisticated technique for aggregating the various simulation runs will be presented, one that is based upon concepts and ideas from stochastic dominance.

5. Model training
   The model training logically follows the generation of a labeled dataset. Here, the process is quite straightforward, involving the training and hyperparameter tuning of one or multiple models of interest. Given the focus on explainability and interpretability, along with aiding in the evaluation of model performance, preference should be given to models that align better with these requirements. Models such as learning classifier systems, XGBoost, and similar approaches are preferred over neural networks.

6. Evaluation
   The final step in the cyclical process is the examination of the steps that came before and the consideration of if, and what, changes and adjustments need to be made. In this step, the experts involved can examine the model performance, determine if it is suitable, decide whether more scenarios need to be generated and of what type, and consider whether

certain adjustments should be made to the evaluation function to better capture their expertise.

As can be noted, the proposed methodology is not a radical departure from conventional approaches in either MCDM or AI, but rather identifies and makes use of various ideas and concepts from both fields and combines them appropriately in order to tackle MCDM in time-critical emergency applications where ML models are used as a part of DSS.

The very cyclical nature of the process is derived from the general MCDM framework. The derivation of scenarios, along with the incorporation of variations and simulations, is extracted from multi-scenario many-objective robust decision-making. The inclusion of context information that changes how the options are considered is borrowed from reinforcement learning states and, more specifically, contextual bandits, along with the idea of using an evaluation or objective function to score the performance of the choice. Finally, the incorporation of a hierarchical, constraint-aware function and the use of stochastic dominance additionally fit well within common MCDM approaches.

Combining all of these concepts yields an approach geared towards tackling time-constrained emergency MCDM problems with the use of an ML-based DSS. It could also be used for a wide array of problems that meet similar requirements, especially where decision makers need support in one decisive and critical step while under great time pressure and stress.

A visual representation of the methodology can be seen in Figure 5.1.

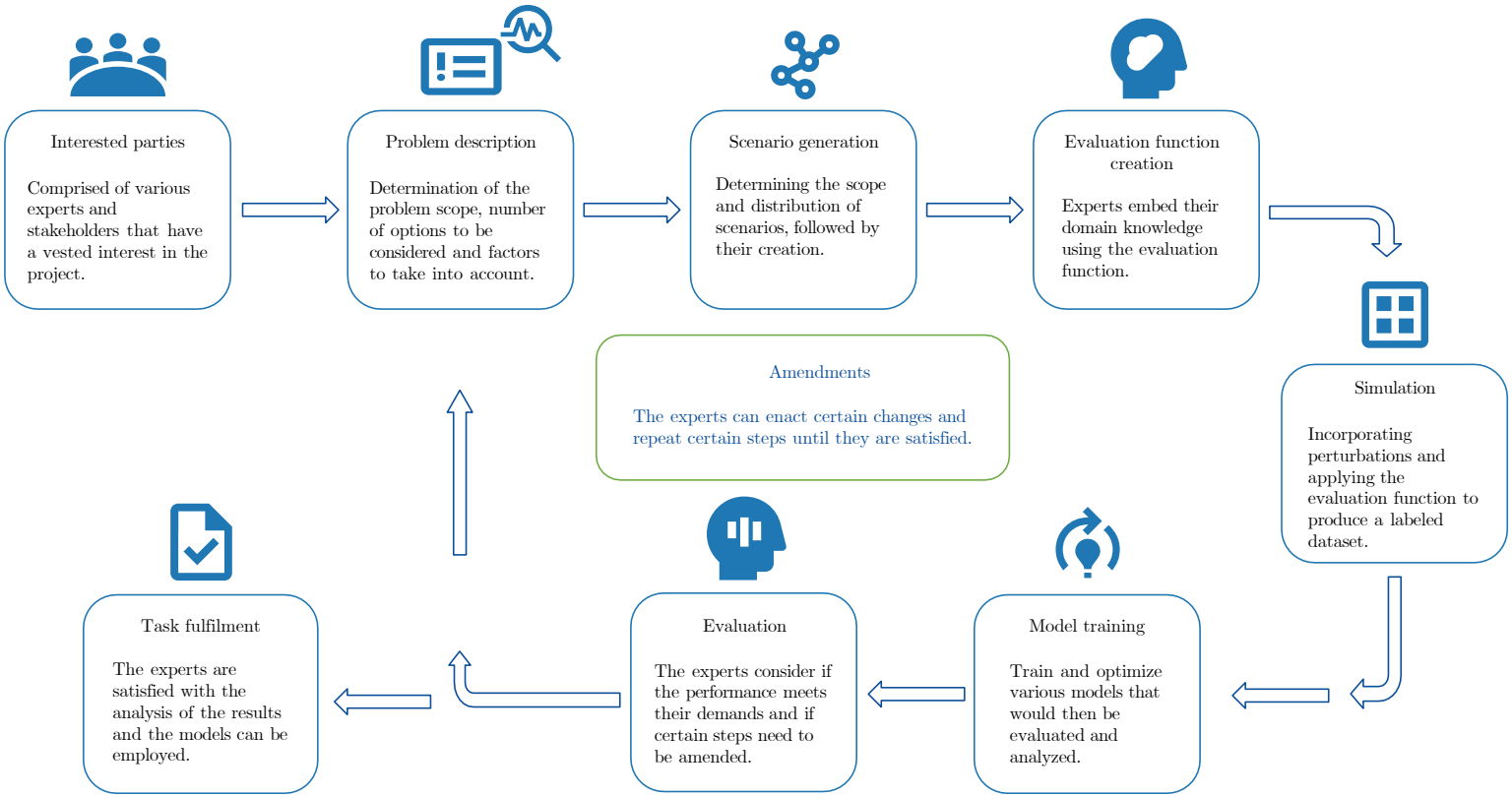

Figure 5.1: ExACT-MCDM methodology

5.3 Hierarchal Constraint-aware Evaluation function concept

The evaluation function is the key aspect of this approach, in whereby the experts can directly encode their domain knowledge and produce evaluations for each option. Great efforts were made to provide a general concept of how to create a function that could be applied to problems beyond the DAAS problem. The proposed ideas make certain assumptions, and it is the duty of the relevant parties to determine if this structure is usable for their own needs. The key assumption is that the problem resembles the DAAS problem in that it has multiple factors, some of which need to be satisfied and others that need to be optimized as much as possible. This is in line with emergency decision-making in which there are certain aspects that one aims to satisfy, often pertaining to safety and well-being, and others that incorporate secondary and tertiary aspects like economic considerations.

Function formulation

Let $\mathcal{A} = \{a_1, a_2, \ldots, a_n\}$ be the set of alternatives. For each $i \in \{1, \ldots, n\}$, let $\mathbf{x}_i = (x_{i1}, \ldots, x_{ip})$ denote the feature vector describing a_i.

The factors of each option can be grouped into two categories:

- $\mathcal{C}$: constraint-critical features (e.g., safety, feasibility),
- $\mathcal{P}$: preference-based features (e.g., efficiency, cost).

This grouping is akin to deriving a more general and flexible form of the risk and cost objectives expressed in Section 4.1 and further expanded upon in the reward function in Section 4.5.

Thus, for each group, we define a transformation function that is applied to every feature within the group. For constraint-critical features, this function serves to encode constraints that can be both soft and hard.

For example, for a constraint feature $j \in \mathcal{C}$, we define

$$\phi_j(x_{ij}) = \begin{cases} 0, & \text{if } x_{ij} \leq s_j, \\ x_{ij} - s_j, & \text{if } s_j < x_{ij} \leq h_j, \\ x_{ij} - s_j, & \text{if } x_{ij} > h_j, \end{cases} \tag{5.1}$$

where s_j and h_j denote the soft and hard thresholds for feature j. If the third case applies, the violation counter is incremented ($\lambda \leftarrow \lambda + 1$).

The example above represents a simple linear transformation. However, the functions and their shapes can be adjusted depending on the problem. This can also be conducted independently for each feature, so each may receive a different function.

For preference-based features $j \in \mathcal{P}$, an analogous transformation $\psi_j(x_{ij})$ may be defined, depending on how preferences are expressed.

Each transformed feature should be normalized to the interval $[0, 1]$ before aggregation, ensuring comparability within each group. The aggregated values for the two groups are thus:

$$S_i^{\text{constr}} = \sum_{j \in \mathcal{C}} \phi_j(x_{ij}), \qquad S_i^{\text{pref}} = \sum_{j \in \mathcal{P}} \psi_j(x_{ij}). \tag{5.2}$$

Let $\lambda \geq 0$ denote the number of hard-constraint violations for option i. The total score for option i is then computed as

$$\text{Score}_i = \underbrace{\text{Norm}\big(S_i^{\text{constr}}\big) + \lambda}_{\text{Constraint group (penalty-dominant)}} + \underbrace{\text{Norm}\Big(S_i^{\text{pref}}\Big)}_{\text{Preference group}}, \tag{5.3}$$

where Norm$(\cdot)$ denotes a global normalization function such as min–max normalization applied across all options:

$$\text{Norm}(z_i) = \frac{z_i - \min_k z_k}{\max_k z_k - \min_k z_k}. \tag{5.4}$$

This structure guarantees that constraint-breaking options receive higher scores (and are ranked lower), while still allowing preference-based features to differentiate between feasible options.

5.3.1 Illustrative example

The following illustrative example showcases how the evaluation function is used to devise scores for three possible alternatives in a simplified alternate-airport selection scenario.

Options and Features

Let $\mathcal{A} = \{a_1, a_2, a_3\}$ represent our three diversion airport options.

Each option is described by the following features, which are grouped accordingly:
Constraint-critical features $\mathcal{C}$:

- x_1: Visibility (meters)—higher is better.
- x_2: Cloud ceiling (feet)—higher is better.
- x_3: Crosswind (knots)—lowerr is better.
- x_4: Runway margin (meters)—higher is better.
- x_5: Fuel reserve margin (kg)—higher is better.

Preference-based features $\mathcal{P}$:

- y_1: Distance to original destination (km)—lower is better.
- y_2: Distance to aircraft's current position (km)—lower is better.
- y_3: Distance to airline hub (km)—lower is better.

For constraint-critical features, the following soft and hard thresholds are defined:

Feature	Soft Threshold	Hard Threshold	Type
Visibility (x_1)	1600 m	800 m	Lower bound
Cloud ceiling (x_2)	500 ft	100 ft	Lower bound
Crosswind (x_3)	10 knots	38 knots	Upper bound
Runway margin (x_4)	N/A	0 m	Lower bound
Fuel reserve margin (x_5)	1500 kg	300 kg	Lower bound

Table 5.1: Threshold values for each constraint feature

For the example given, the options have the following feature values:

Step 1: Calculate Constraint Violations

For each constraint-critical feature, we calculate the violation magnitude using the transformation function $\phi_j(x_{ij})$.

For lower-bound constraints (visibility, cloud ceiling, runway margin, fuel reserve), the violation is calculated as:

$$\phi_j(x_{ij}) = \begin{cases} 0, & \text{if } x_{ij} \geq s_j \text{ (no violation)}, \\ s_j - x_{ij}, & \text{if } h_j \leq x_{ij} < s_j \text{ (soft violation)}, \\ s_j - x_{ij}, & \text{if } x_{ij} < h_j \text{ (hard violation)}. \end{cases} \tag{5.5}$$

Feature	Option a_1	Option a_2	Option a_3
Constraint-critical features			
Visibility (x_1)	1200 m	600 m	2000 m
Cloud ceiling (x_2)	400 ft	80 ft	600 ft
Crosswind (x_3)	15 knots	40 knots	8 knots
Runway margin (x_4)	150 m	-50 m	200 m
Fuel reserve margin (x_5)	1200 kg	200 kg	1800 kg
Preference-based features			
Distance to original destination (y_1)	120 NM	80 NM	200 NM
Distance to aircraft position (y_2)	50 NM	30 NM	100 NM
Distance to airline hub (y_3)	150 NM	100 NM	80 NM

Table 5.2: Raw feature values for the three options.

For upper-bound constraints (crosswind), the violation is calculated as:

$$\phi_j(x_{ij}) = \begin{cases} 0, & \text{if } x_{ij} \leq s_j \text{ (no violation)}, \\ x_{ij} - s_j, & \text{if } s_j < x_{ij} \leq h_j \text{ (soft violation)}, \\ x_{ij} - s_j, & \text{if } x_{ij} > h_j \text{ (hard violation)}. \end{cases} \tag{5.6}$$

Violation Calculations

Visibility (x_1):

$$\phi_1(x_{11}) = 1600 - 1200 = 400 \text{ (soft violation)}$$
$$\phi_1(x_{21}) = 1600 - 600 = 1000 \text{ (hard violation)}$$
$$\phi_1(x_{31}) = 0 \text{ (no violation)}$$

Cloud ceiling (x_2):

$$\phi_2(x_{12}) = 500 - 400 = 100 \text{ (soft violation)}$$
$$\phi_2(x_{22}) = 500 - 80 = 420 \text{ (hard violation)}$$
$$\phi_2(x_{32}) = 0 \text{ (no violation)}$$

Crosswind (x_3):

$$\phi_3(x_{13}) = 15 - 10 = 5 \text{ (soft violation)}$$
$$\phi_3(x_{23}) = 40 - 10 = 30 \text{ (hard violation)}$$
$$\phi_3(x_{33}) = 0 \text{ (no violation)}$$

Runway margin (x_4):

$$\phi_4(x_{14}) = 0 \text{ (no violation)}$$
$$\phi_4(x_{24}) = |(-50)| = 50 \text{ (hard violation)}$$
$$\phi_4(x_{34}) = 0 \text{ (no violation)}$$

Fuel reserve margin (x_5):

$$\phi_5(x_{15}) = 1500 - 1200 = 300 \text{ (soft violation)}$$
$$\phi_5(x_{25}) = 1500 - 200 = 1300 \text{ (hard violation)}$$
$$\phi_5(x_{35}) = 0 \text{ (no violation)}$$

The number of hard violations are also determined (λ) for each option:

$$\lambda_1 = 0 \text{ (no hard violations)}$$
$$\lambda_2 = 4 \text{ (visibility, cloud ceiling, crosswind, runway margin)}$$
$$\lambda_3 = 0 \text{ (no hard violations)}$$

Feature	Option a_1	Option a_2	Option a_3
Visibility	400 (soft)	1000 (hard)	0
Cloud ceiling	100 (soft)	420 (hard)	0
Crosswind	5 (soft)	30 (hard)	0
Runway margin	0	50 (hard)	0
Fuel reserve margin	300 (soft)	1300 (hard)	0
Raw sum	805	2800	0
Hard violation count (λ)	0	4	0

Table 5.3: Constraint Violation Summary

Step 2: First-level Normalization (Dataset-wide)

Each feature's violations are normalized across all options to a range of $[0, 1]$ using min–max normalization. For each feature j:

$$\text{Norm}(\phi_j(x_{ij})) = \frac{\phi_j(x_{ij}) - \min_k \phi_j(x_{kj})}{\max_k \phi_j(x_{kj}) - \min_k \phi_j(x_{kj})} \tag{5.7}$$

Feature	Option a_1	Option a_2	Option a_3
Visibility	0.20	1.00	0.00
Cloud ceiling	0.25	1.00	0.00
Crosswind	0.14	0.86	0.00
Runway margin	0.00	1.00	0.00
Fuel reserve margin	0.19	0.81	0.00
Normalized sum	0.78	4.67	0.00
Hard violation count (λ)	0	4	0

Table 5.4: Normalized Constraint Violations

Step 3: Process Preference Features

For preference-based features, given that they are distances, they should simply be minimized as much as possible. First, each feature across all options is normalized to the range, as was performed for the constraint features $[0, 1]$ and depicted in Table 5.5.

Feature	Option a_1	Option a_2	Option a_3
Distance to original destination (y_1)	0.25	0.00	1.00
Distance to aircraft position (y_2)	0.20	0.00	1.00
Distance to airline hub (y_3)	0.88	0.38	0.00
Normalized sum	1.33	0.38	2.00

Table 5.5: Normalized Preference Features

Step 4: Calculate Aggregated Sums

The normalized values of each group are summed.
Constraint Group Sum:

$$S_1^{\text{constr}} = 0.4 + 0.238 + 0.167 + 0.0 + 0.231 = 1.036$$
$$S_2^{\text{constr}} = 1.0 + 1.0 + 1.0 + 1.0 + 1.0 = 5.0$$
$$S_3^{\text{constr}} = 0.0 + 0.0 + 0.0 + 0.0 + 0.0 = 0.0$$

Preference Group Sum:

$$S_1^{\text{pref}} = 0.333 + 0.286 + 1.0 = 1.619$$
$$S_2^{\text{pref}} = 0.0 + 0.0 + 0.286 = 0.286$$
$$S_3^{\text{pref}} = 1.0 + 1.0 + 0.0 = 2.0$$

Step 5: Second-level Normalization

The aggregated sums are normalized across all options.
Normalized Constraint Sum:

$$\text{Norm}(S_1^{\text{constr}}) = \frac{1.036 - 0.0}{5.0 - 0.0} = 0.207$$
$$\text{Norm}(S_2^{\text{constr}}) = \frac{5.0 - 0.0}{5.0 - 0.0} = 1.0$$
$$\text{Norm}(S_3^{\text{constr}}) = \frac{0.0 - 0.0}{5.0 - 0.0} = 0.0$$

Normalized Preference Sum:

$$\text{Norm}(S_1^{\text{pref}}) = \frac{1.619 - 0.286}{2.0 - 0.286} = 0.778$$
$$\text{Norm}(S_2^{\text{pref}}) = \frac{0.286 - 0.286}{2.0 - 0.286} = 0.0$$
$$\text{Norm}(S_3^{\text{pref}}) = \frac{2.0 - 0.286}{2.0 - 0.286} = 1.0$$

Step 6: Calculate Final Scores

The final score for each option is calculated as:

$$\text{Score}_i = \text{Norm}(S_i^{\text{constr}}) + \lambda_i + \text{Norm}(S_i^{\text{pref}}) \tag{5.8}$$

$$\text{Score}_1 = 0.207 + 0 + 0.778 = 0.985$$
$$\text{Score}_2 = 1.0 + 4 + 0.0 = 5.0$$
$$\text{Score}_3 = 0.0 + 0 + 1.0 = 1.0$$

Ranking of options

Since lower scores are better (representing fewer violations and better preferences), the options are ranked as follows:

Rank	Option	Score	Explanation
1	a_1	0.985	Best overall balance of constraints and preferences
2	a_3	1.000	No constraint violations but worse on preferences
3	a_2	5.000	Multiple hard constraint violations

Table 5.6: Final ranking of options based on aggregated score.

This example demonstrates two key aspects of the evaluation function:

1. Hard constraint dominance: Option a_2 has four hard constraint violations, resulting in a significantly higher score despite having the best preference features. In short, the evaluation function prioritizes safety by heavily penalizing hard constraint violations through the λ counter.
2. Balancing trade-offs: Option a_1 has several small soft constraint violations but performs sufficiently better regarding preferences than Option a_3, resulting in a slightly better overall score. An important consideration here is how the soft constraints are defined and viewed. In this specific case, they are viewed as areas where pilots require more attention and are deemed to be safe enough that the operational aspects are considered on the same level; however, this could be enacted differently.

In this aviation context, Option a_1 represents the best diversion airport because it balances safety constraints, with only soft violations and operational preferences. Option a_2 would be unsafe due to multiple hard violations of critical safety parameters, while Option a_3 is safe but less optimal from an operational perspective.

5.3.2 Final note on the evaluation function

Grouping

The idea of the evaluation function is based on grouping features that will be treated equally within the group, relating to the idea of the risk and cost metrics from the initial approach. The benefit is that this new grouping mechanism allows for the inclusion of as many factors as one wishes, which solves a major limitation of the original approach where the risk and cost metrics were, in a sense, fixed and defined for a specific type of aircraft and use case. This new proposed evaluation function mechanism enables different airlines or aviation producers to tailor the evaluation function and factors considered based on their judgment, while also allowing for even further flexibility in other problems or domains.

Inclusion of transformation functions and aggregation

The transformation functions allow for the inclusion of soft and hard constraints, along with a finer expression of one's preferences. The aggregation on a per-group basis can be performed using simple sums or even a weighted sum. Another possibility is the inclusion of the Choquet integral in order to additionally express the synergies between features.

However, although weights can be used both within groups and between groupings to obtain the final score, they are discouraged. The idea is to rely on the more expressive and clear transformation functions and groupings, as they are more straightforward and transparent. Weights, on the other hand, can be difficult to derive and even harder to justify regarding why certain values were chosen. Thus, while they could be applied, they should be considered a last resort once groupings, transformation functions, and the Choquet integral prove insufficient.

5.4 Stochastic dominance-based score

The methodology involves varying certain factors across a scenario in ordered to obtain a better estimate and provide a better labeling. Although a simple mean could be used, it may also obscure risk-related properties and fail to provide the best estimate. Given that we obtained a distribution of values, the approach makes use of ideas from Stochastic dominance (SD) to compare options by considering the entire distribution of outcomes. In particular, first-order stochastic dominance (SD1), second-order stochastic dominance (SD2), and third-order stochastic dominance (SD3) are well-established tools in decision theory and finance for ranking risky alternatives [157, 158].

5.4.1 Classical Stochastic Dominance

Let X and Y be two random variables with cumulative distribution functions (CDFs) F_X and F_Y.

First-order stochastic dominance (SD1)

X first-order dominates Y (denoted $X \succeq_1 Y$) if

$$F_X(z) \leq F_Y(z) \quad \text{for all } z, \tag{5.9}$$

with strict inequality for some z. This means that X yields higher values with greater probability across all thresholds, and represents the preferences of all decision makers who prefer "more to less." A visual illustration is showcased in Figure 5.2, in which A is better than B.

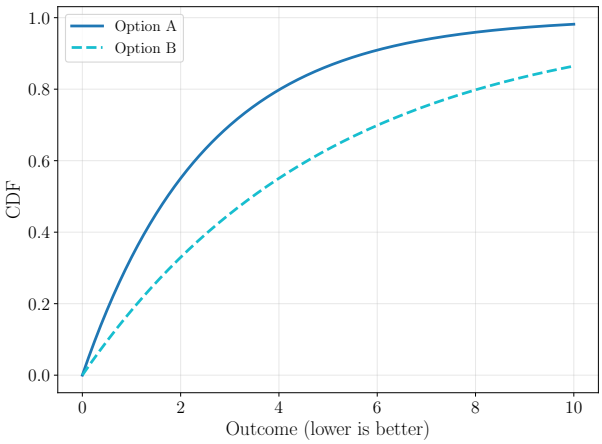

Figure 5.2: Stochastic dominance type I

Second-order stochastic dominance (SD2)

X second-order dominates Y (denoted $X \succeq_2 Y$) if

$$\int_{-\infty}^{z} F_X(t)\,dt \;\leq\; \int_{-\infty}^{z} F_Y(t)\,dt \quad \text{for all } z, \tag{5.10}$$

with strict inequality for some z. This corresponds to the preferences of all risk-averse decision makers, since it penalizes distributions with more mass in the lower tail. Visually, SD2 can be seen in Figure 5.2, in which A is better than B.

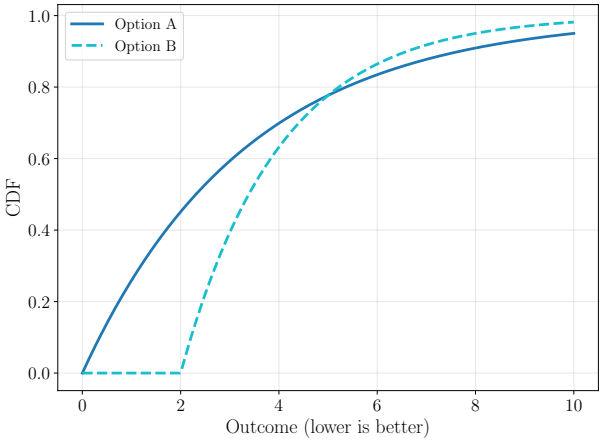

Figure 5.3: Stochastic dominance type II

Third-order stochastic dominance (SD3)

X third-order dominates Y (denoted $X \succeq_3 Y$) if

$$\int_{-\infty}^{z}\int_{-\infty}^{t} F_X(u)\,du\,dt \;\leq\; \int_{-\infty}^{z}\int_{-\infty}^{t} F_Y(u)\,du\,dt \quad \text{for all } z. \tag{5.11}$$

This captures the preferences of all decision makers who are risk-averse and also prefer positive skewness (i.e., they dislike downside risk but appreciate upside potential). Figure 5.4 presents this visually.

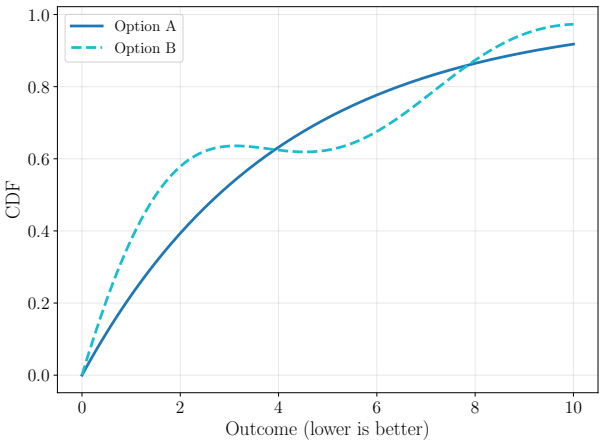

Figure 5.4: Stochastic dominance type III

5.4.2 Two-Stage Stochastic dominance-based ranking

Classical stochastic dominance relies on pairwise comparisons and thus could result in incomplete ordering. To mitigate this problem, a two-stage process may be implemented, which would result in complete ordering.

In the first stage, a pairwise comparison is carried out in which each option receives a value based on the number of other options it dominates. Should multiple outcomes have the same value, they are placed together in a group, an idea borrowed from the fronts implementation in the famous NSGA-II algorithm. Within each of these groupings, a score is calculated based on the within-scenario ranking the options received. Thus, for each scenario, all options are ranked from best (rank 1) to worst (rank n) based on their performance scores. These are then aggregated using a consistency-adjusted scoring function:

$$S_i = \beta \cdot \frac{1}{m} \sum_{j=1}^{m} r_{ij} + (1 - \beta) \cdot \sqrt{\frac{1}{m-1} \sum_{j=1}^{m} \max(0, r_{ij} - \bar{r}_i)^2} \tag{5.12}$$

where $\bar{r}_i$ is the mean rank for option i, and the consistency term only penalizes ranks that are worse than the mean. Thus, with this method, additional scores for each option within the grouping can be determined that function similarly to the stochastic dominance method and enable complete ordering of the options within a scenario.

On the usage of aggregation methods

The use of this stochastic-dominance-inspired ordering results from the fact that it is a well-established method that aptly aligns with the goals defined for this type of decision-making. However, the presented method is merely one possible way to obtain the final ordering, with other methods and approaches that could be considered based on the needs of the interested parties.

5.5 Benefits and Considerations

The proposed methodology for tackling time-constrained MCDM problems aligns well with the needs these problems present and attempts to fill the gap regarding the current methods from the literature.

Benefits

The core advantage of this methodology stems from its manner of addressing the problem of limited data. The use of scenario-generated datasets, along with the described evaluation function, means that many varied labeled data inputs can be produced. The evaluation function in particular allows experts to encode domain knowledge in order to generate better labels and evaluations for the newly created datasets.

An argument could be made that there is a large degree of complexity in the creation of the evaluation function, and that the methodology is not as plug-and-play as methods such as AHP, which follow more straightforward and easy-to-understand steps. However, the opposite argument can also be made. The nature of these classical methods may deceptively appear clearer and simpler because much of the difficult judgment is left to the decision makers and stakeholders themselves, who must contemplate these challenging processes and produce answers in the form of pairwise rankings or weight vectors.

Here, the method demands more clarity and precision from the experts, both for the sake of both interpretability and explainability, but also in order to tackle the more elaborate nature of the evaluation and its objectives. Thus, although classical methods may seem simpler, one could argue that they obscure highly complex decision-making processes behind the ready-made preferences of experts, thereby reducing explainability and interpretability and causing confusion and inconsistencies.

Having this dataset enables the training of a model that can then be deployed as part of a DSS. It acts as a surrogate model, providing near-instant decision support without the need to run lengthy and potentially less predictable simulations at the time of the emergency. The model also enables generalization to scenarios that were not directly seen or included in the dataset.

In a sense, the model also addresses the problem of similarity. Attempts to create an explicit similarity function for the scenarios proved difficult due to the associative and cumulative properties of addition. Currently, well-established similarity functions seem ill-suited to address this issue, as they treat each of the factors the same, but their importance varies with context, with some features always being of considerable importance. In addition, scenarios are made up of many options, and unrolling them into a vector and using them for comparison results in uneven and unfair comparisons, where things such as the ordering of the options can prove challenging. Therefore, the model can be understood as a mechanism to implicitly encode how scenarios should be scored and related to one another. This allows it to approximate which scenarios from the dataset are most closely aligned with the one the decision makers are facing and to use that information to produce a suitable suggestion for a course of action.

Finally, one of the common problems associated with using MCDM methods in emergency situations is the fact that most methods cannot easily adapt to a changing situation. Although the focus of this method is on making a single decisive multi-criteria decision, that does not mean it cannot be employed multiple times should the situation change. The models and the DSS they are part of can simply be employed once when initial assistance is required, and then again if new information arrives that changes the scenario. In such a case, the model can reassess and provide new decision support for the updated situation. In a way, it behaves similarly to memoryless contextual bandits and allows continual decision support while also recognizing that the suggestions need to remain valid and stable for at least some determined amount of time so that the decision-makers can develop an understanding of them. This mechanism can also be set up to continuously update the ordering and options in the background, only presenting the new situation to the decision-makers if there is an order change or a change in predetermined magnitude. This ensures that the system is not overly sensitive and does not overwhelm the decision-makers.

Considerations

Although the methodology provides options for addressing certain issues, there are some aspects that should be taken into account when employing it. Foremost is the data generation, as this is the most crucial step and, ultimately, what the model learns from. The data generation process, along with its subsequent variation and perturbation, is vital—as the old adage goes, garbage in, garbage out. Therefore, great care must be taken. Of particular note is the importance of deciding which types of scenarios are generated and how, given that the sampling process is incredibly important for the generalization capabilities of the model.

While making use of simulations and synthetic or augmented data, this does not mean that the data is unlimited, as there could be high computational costs in generating the scenarios. However, the assumption here is that the process can be scaled and parallelized, and that computational time is less important than human time and effort, which should be carefully planned for and used effectively. Furthermore, arguments could be made for scenarios in which the model training process is entirely reframed as a single-step, bandit-style, reinforcement learning process. In cases where the number of options is equal to or significantly greater than the number of perturbations per option required for a robust evaluation, it may be more computationally efficient to model the problem this way, thereby reducing the number of simulations and evaluations required. If such an approach is adopted, the exploration–exploitation trade-off inherent in reinforcement learning should also be carefully considered.

Another important aspect pertains to the cyclical nature and evaluation of the entire process. Although the evaluation function enables the labeling of the entire dataset, depending on its size, it may be difficult to ascertain whether the function is working exactly as intended or whether there are scenarios in which the labeling does not align perfectly with the experts' preferences and guidelines. However, in many real-world datasets, mislabeling is inevitable and should only be of concern if there are vast swaths of scenarios in which this occurs. The real issue is the impracticality of checking each and every scenario. A possible workaround is to ensure extreme edge cases and scenarios of particular interest are prepared and inserted by the experts so that they can validate whether everything is working as intended, supplemented with random sampling of scenarios to be checked. Although this requires direct involvement of the experts, the assumption is that checking a sample of scenarios is a far less demanding task than having to provide evaluations for all of them.

The methodology has many moving parts. However, these make the process more explicit and direct, and they provide mechanisms for the experts to clearly state their preferences and intentions. Ideally, these would then be captured by the models and later prove helpful in the deployed DSS.

5.6 Summary of this Chapter

This chapter presented the ExACT-MCDM methodology (Explainable, Adaptive, Context-aware, Time-critical Multi-Criteria Decision-Making), which was designed to address the challenges of emergency decision-making in contexts where data is scarce and time is limited. The approach combines scenario-based data generation, a structured evaluation function, and robustness through simulation in order to create suitable training data for machine learning models that can later act as part of a decision-support system.

The key idea is to encode expert domain knowledge into an evaluation function that distinguishes between constraint-critical and preference-based features while incorporating both soft and hard thresholds. Scenario generation and the introduction of variability provide robustness by simulating real uncertainty, after which the evaluation function is applied to produce scores for all options. This results in a labeled dataset that can be used to train interpretable machine learning models. The cyclical nature of the methodology resembles human-in-the-loop processes that can be refined continuously, aligning it with general MCDM principles while making it suitable for the demands of time-critical emergency decision-making.

Part III

Methodology Testing

6 Data Creation Pipeline

The following chapter describes the entire data generation pipeline for the DAAS problem version that will be used as a testing ground for the newly proposed ExACT-MCDM methodology. Although an existing dataset was used in the process of developing the methodology, here, a more advanced version of the problem will be used so that it more closely reflects the reality of alternative-airport selection. What follows is a listing of the factors considered and with their grouping, as well as the transformation functions and their soft and hard constraints. The chapter also showcases how the scenarios are created, including a more detailed description of the more sophisticated simulation used to generate them.

6.1 Scope and Assumptions

In order to test the methodology in practice, the author assumes the role of a unified expert panel, effectively acting as an aggregation of expert judgment. This choice was made as a simplifying assumption to allow for systematic development and evaluation of the methodology without the logistical complexity of assembling a full expert group. Nevertheless, the process was not carried out in isolation. Throughout the development, feedback was gathered from colleagues working on the IPAS system and pilots that were at the DLR, in order to better capture the operational perspective. Furthermore, technical documentation and manuals were consulted to ensure that the assumptions, scenario definitions, and evaluation functions aligned with established aviation practices. This combined approach allowed the methodology to balance practical realism with methodological clarity, while acknowledging that in a real-world deployment, the role of expertise would be distributed across multiple interested parties rather than concentrated on a single analyst.

Although the dataset and the respective simulations and mechanisms for their creation are improved compared to the original ones used in the methodology, similar assumptions are still present. Namely, for the experiments, only the Airbus A321 aircraft is considered, which is one of the most widely used narrow-body aircraft worldwide and particularly popular in Europe [129]. The choice of focusing on a single aircraft type is an arbitrary but simplifying assumption that allows for clearer analysis. At the same time, it reflects a realistic operational context, as the A321 is highly representative of short- and medium-haul commercial aviation.

In the experiments, the scenario represents a single snapshot at the moment when a decision needs to be made. At this point, the models receive their input and produce recommendations regarding which options to take. It is assumed that the model output remains generally valid for approximately ten minutes, providing a reasonable buffer for the final decision makers to consider

the evaluations produced.

Based on information from pilots indicating their desire for approximately five options to be shown to them, the opportunity was taken to reduce the problem by considering only the five closest airports to the aircraft position as possible options.

Having clarified the assumptions, a more structured foundation for the data generation process is established. The following sections build on this foundation by specifying the factors considered and detailing the framework used to create realistic decision scenarios.

6.2 Factors considered

The factors to be considered were determined through various information gathering methods, primarily by consulting the flight manual [13] and interviews with pilots [281, 280, 279].

The factors that were chosen are:

Context factors:

1. Airline—Different airlines may have contractual obligations, preferred hubs, or operational restrictions influencing alternate choices.
2. Altitude—Determines reachable airports within remaining fuel and time constraints.
3. Initial fuel—Critical for estimating endurance and feasibility of diversion options.
4. Status—Captures the nature of the emergency (technical, medical, weather-related), which can alter the priority of selection criteria.

Airport-specific factors:

1. Headwind—Increases fuel consumption and time required to reach an alternate.
2. Tailwind—Reduces fuel consumption and travel time, making an alternate more favorable.
3. Crosswind—Affects landing safety; strong crosswinds can render a runway unusable.
4. Visibility—Determines whether safe landing procedures can be performed.
5. Cloud ceiling—Restricts approach types and can limit accessibility to airports with less advanced navigation aids.
6. Runway length—Physical limitation for landing; must be sufficient for the aircraft type.
7. Runway length required—Minimum safe distance needed by the current aircraft configuration and weight for landing.
8. Runway width—Necessary for safe maneuvering and rollout, especially for larger aircraft.
9. Runway condition—Surface state (dry, wet, or icy) directly influences braking performance and safety.
10. Fuel needed—Estimated fuel consumption to reach the alternate.
11. Average distance to closest airports—Reflects regional airport density and potential backup options if conditions change.

12. Distance to destination—Showcases the displacement in passengers and goods with regards to the original plan
13. Distance to aircraft—Direct measure of time and fuel required to reach the alternate.
14. Distance to hub—Indicates whether the alternate is near a hub with support facilities.
15. Hub type—Classifies airports (major hub, regional, or maintenance base), affecting available services during diversion.

Context factors versus options factors

The distinction between context factors and airport-specific factors is important when it comes to dynamic alternate-airport selection. Context values describe the initial situation of the aircraft and its operator, such as altitude, fuel state, or airline policies. These values are relatively fixed for the decision at hand and provide the boundary conditions within which diversion decisions must be made. In contrast, airport factors describe the characteristics of the available alternatives and their operational feasibility, such as weather, runway conditions, and distances. These values can vary significantly across different options and determine which airports are practically suitable in the given context. Considering both categories together ensures that the decision-making process remains suitable.

Based on the methodology, the factor groupings are showcased in Table 6.1.

Feature	**Unit**
Constraint-critical features	
Visibility (x_1)	meters (m)
Cloud ceiling (x_2)	feet (ft)
Crosswind (x_3)	knots (kt)
Tailwind (x_4)	knots (kt)
Runway margin (x_5)	meters (m)
Fuel needed (x_6)	kilograms (kg)
Fuel reserve margin (x_7)	kilograms (kg)
Distance to aircraft position (y_2) *	nautical miles (NM)
Preference-based features	
Distance to original destination (y_1)	nautical miles (NM)
Distance to aircraft position (y_2) *	nautical miles (NM)
Distance to airline hub (y_3)	nautical miles(NM)
Average distance to closest airports (y_4)	nautical miles (NM)

Table 6.1: Grouping of features into constraint-critical and preference-based categories, with units of measurement.

The transformation functions were defined in a similar manner to the exmaple consisting of linear piecewise functions, specifically:

$$\phi_j(x_{ij}) = \begin{cases} 0, & \text{if } x_{ij} \geq s_j \text{ (no violation)}, \\ s_j - x_{ij}, & \text{if } h_j \leq x_{ij} < s_j \text{ (soft violation)}, \\ s_j - x_{ij}, & \text{if } x_{ij} < h_j \text{ (hard violation)}. \end{cases} \tag{6.1}$$

for lower-bound constraints, such as visibility, cloud ceiling, runway length needed, and fuel reserve.

$$\phi_j(x_{ij}) = \begin{cases} 0, & \text{if } x_{ij} \leq s_j \text{ (no violation)}, \\ x_{ij} - s_j, & \text{if } s_j < x_{ij} \leq h_j \text{ (soft violation)}, \\ x_{ij} - s_j, & \text{if } x_{ij} > h_j \text{ (hard violation)}. \end{cases} \tag{6.2}$$

for upper-bound constraints such, as crosswind. Additionally, we used the violation counter λ to keep track of the hard constraint violations $\lambda \leftarrow \lambda + 1$

The soft and hard constraints for each factor can be seen in Table 6.2.

Feature	Soft Threshold	Hard Threshold	Type
Visibility (x_1)	1600 m	800 m	Lower bound
Cloud ceiling (x_2)	500 ft	100 ft	Lower bound
Crosswind (x_3)	10 knots	38 knots	Upper bound
Tailwind (x_4)	5 knots	10 knots	Upper bound
Runway needed (x_5)	N/A	Runway available m	Lower bound
Fuel reserve margin (x_6)	1500 kg	300 kg	Lower bound

Table 6.2: Threshold values for application constraint-critical features

Caveats

Worthy of special mention is the distance to aircraft position, which is essentially the distance from the airport to the considered alternative airport. This distance is present in both groupings and makes them non-disjoint sets. The value of this distance in the constrained factors is set to zero unless there is a medical emergency on board, encoded in the status factor, which necessitates the inclusion of this distance in both groupings to additionally highlight the importance of ensuring timely medical attention for a passenger. Additionally, within the preference groupings, the distance to hub airports is also slightly adjusted, being halved when the closest hub is a primary hub, as such facilities can offer ample support to both the crew and passengers.

6.3 Scenario Framework

Having established the factors to be considered and the constraints, the focus now shifts toward the scenario generation step. A scenario consists of the context values and five options, each representing the possible alternatives (in this case, airports). The scenario generation process is carried out by means of a simulation with multiple components in order to simultaneously account for wind and weather phenomena and simulate the technical aspects of the aircraft. What follows is a detailed description of the various key elements of the simulation, along with how they integrate into a final whole in order to generate the data needed for training of the models.

6.3.1 Flight plans and Airports

Ensuring an accurate assessment of the possible positions in which an aircraft might find itself when needing to decide on an alternate airport is of great importance, as this directly influences the distance factors and, by extension, the fuel calculations. Therefore, having accurate approximations of where such a decision is most likely to occur in the airspace increases the real-world applicability of the scenarios. To this end, in the scenario generation process, multiple flight plans were taken from EUROCONTROL flight plan databases [3, 4].

A flight plan is the structured route that an aircraft follows between its departure and destination airports. It consists of a sequence of predefined navigation points, called waypoints, which are used by air traffic control and onboard systems to ensure safe and efficient routing. Each waypoint is defined by geographic coordinates and often associated with airways that connect different parts of the airspace network. In practice, flight plans balance operational considerations such as air traffic regulations, weather conditions, and fuel efficiency. The example in Figure 6.1 illustrates a typical route between Berlin (EDDB) and Frankfurt (EDDF), highlighting the intermediate waypoints that structure the aircraft's trajectory.

The flight plans used were selected by considering the top destinations from the largest commercial airports in Europe, ensuring reasonable realistic coverage.

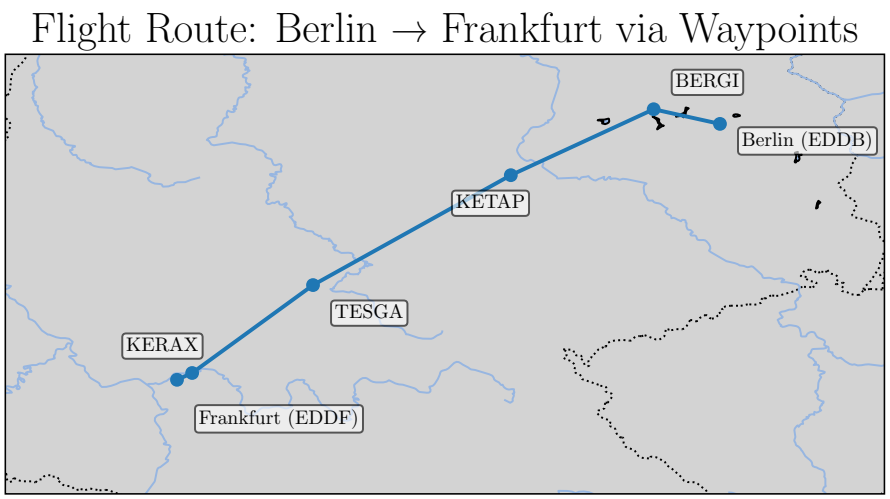

Figure 6.1: Illustrative example of flight plan waypoints from Berlin to Frankfurt

To provide realistic geographical and infrastructural information about airports, data were parsed from the X-Plane flight simulator database [151]. This dataset contains detailed airport layouts, runway information, and navigation aids, making it a valuable source for scenario generation. Although originally intended for flight simulation, the X-Plane database is based on real-world aviation data and is suitable for the intended use. The parsed airports were additionally filtered to include only those with at least one runway 2000 m in length and located roughly within Europe, which was defined by a constraint box with the following coordinates as the vertices:

- Southern edge: 34.0° N.
- Northern edge: 71.0° N.
- Western edge: -24.0° E (24° W).
- Eastern edge: 40.0° E.

A visual representation of the bounding box can be seen in Figure 6.2. A detailed list of the airports considered hubs and their respective airlines is also included in Appendix B.

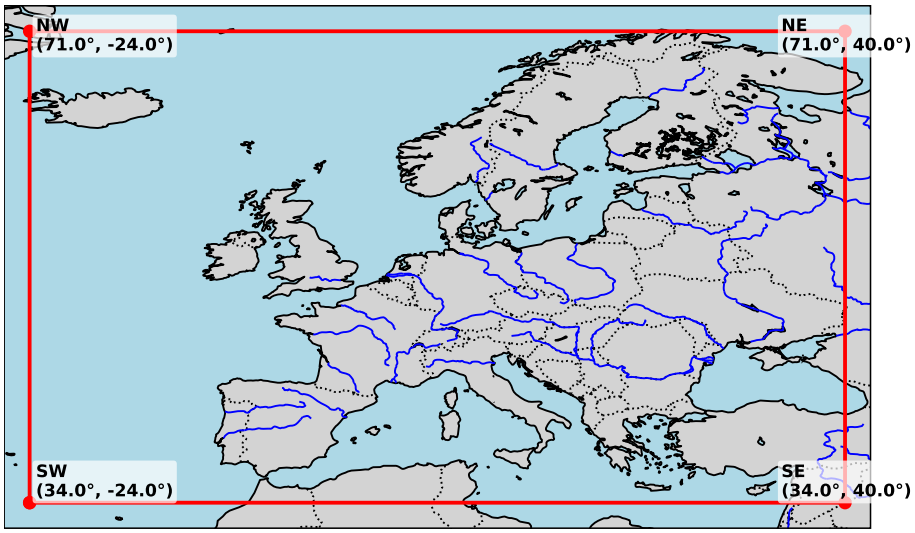

Figure 6.2: Geographical bounding box defining the area of parsed airports across Europe

6.3.2 Wind and Weather phenomena

The wind and weather phenomena play the most crucial role within the simulation, as they have the strongest influence on many of the factors and are outright factors themselves. It is important to note that realistic wind and weather simulation can be extremely difficult and complex, thus for the current use case, the simulation pipeline was somewhat simplified but still erred on the side of providing reasonably realistic approximations that would be useful.

Wind Data

In order to realistically model the influence that the wind has on the decsion-making process, the wind data was taken from NASA's MERRA-2 (Modern-Era Retrospective analysis for Research and Applications, Version 2) reanalysis dataset [2]. MERRA-2 has provided globally gridded wind fields and other meteorological variables since 1980, generated by merging satellite observations with the GEOS model in a consistent framework.

The wind data was obtained for the period between 01.01.2020 and 01.08.2024 as NETCDF4 files. Each file corresponds to a single day and contains 24 hourly time steps. From these, a random hour was selected and the zonal and meridional wind components (U10M and V10M) were extracted for the European bounding box and subsequently employed to generate realistic wind conditions for one scenario. A visual example of a possible wind field from the extracted data can be seen in Figure 6.3. It is important to note one of the simplifications applied here: although an actual wind field corresponding to a realistic real-world distribution was employed, the wind field is constant and does not change beyond the hour selected. However, a strong argument can be made that wind incorporation is not often applied in many air traffic control simulations, and even in many of DLR's internal simulations, wind is only specified as a constant vector rather than as a real wind field obtained from actual atmospheric data.

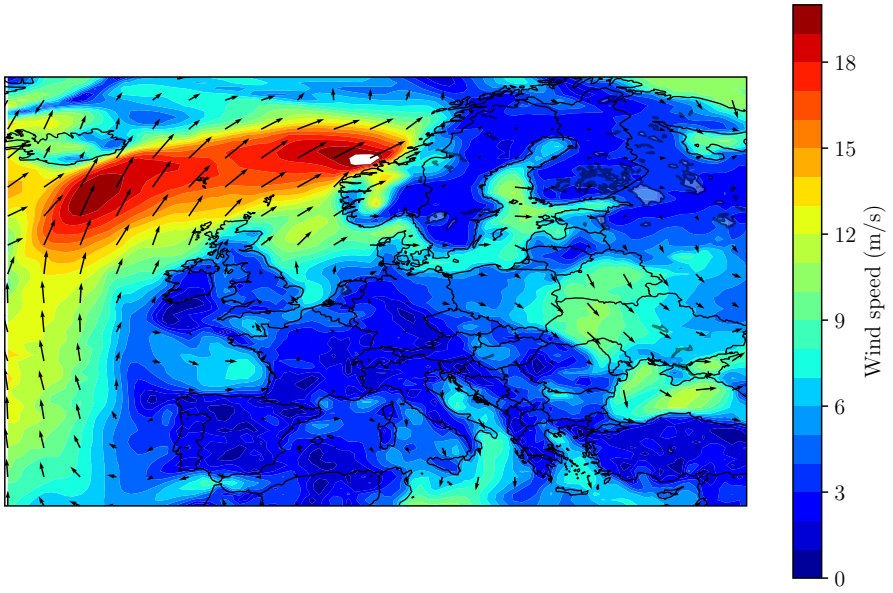

Figure 6.3: MERRA-2 10m Wind Speed and Direction

Wind field

The MERRA-2 dataset provides excellent real-world wind information; however, it only near-surface wind components at 2 m and 10 m above ground level. Additionally, it only includes information about the wind regarding the regular latitude–longitude grid at a resolution of $0.5° \times 0.625°$. Due to aircraft operations and to adress the limitation of the MERRA-2 dataset, the surface values were extrapolated using the commonly applied wind profile power law [121]. This formulation allows wind speeds at arbitrary altitudes to be approximated from a reference measurement height, while also incorporating a gradual

change in direction with altitude. The power law is given by

$$V(z) = V(z_{\text{ref}}) \left(\frac{z}{z_{\text{ref}}} \right)^{\alpha}, \tag{6.3}$$

where $V(z)$ is the wind speed at altitude z, z_{ref} is the reference altitude (2 m or 10 m in the case of MERRA-2), and α is the empirical Hellmann exponent, commonly taken as $\alpha = 0.143$ for neutral atmospheric conditions. This approach allows for the generation of approximate wind profiles at typical flight altitudes such as 10,000 ft and above, thereby ensuring that the simulated conditions more closely resemble real-world flight environments. Additionally, to obtain wind estimates at arbitrary aircraft positions, linear interpolation was applied using the griddata function from the SciPy library [265]. This ensures continuous wind information across the entire bounding box.

Crosswind and Tailwind components

Regarding wind, the crosswind and tailwind components at the airport are factors in their own right due to their safety implications and influence over other considerations, such as the runway length required. However, airports are not built without accounting for these aspects, and most runways are positioned to minimize the negative effects of crosswind and tailwind. Excessive crosswind or tailwind can cause the aircraft to veer off the runway with catastrophic consequences (a situation also referred to as a runway excursion). At the same time, runway orientation is designed to take advantage of headwind, which slows the aircraft down and enables safer landing conditions. A visual representation of the wind components and their relation to the runway can be seen in Figure 6.4; the image is the same as the one used in Chapter 4, Section 4.1, and is included again for additional clarity.

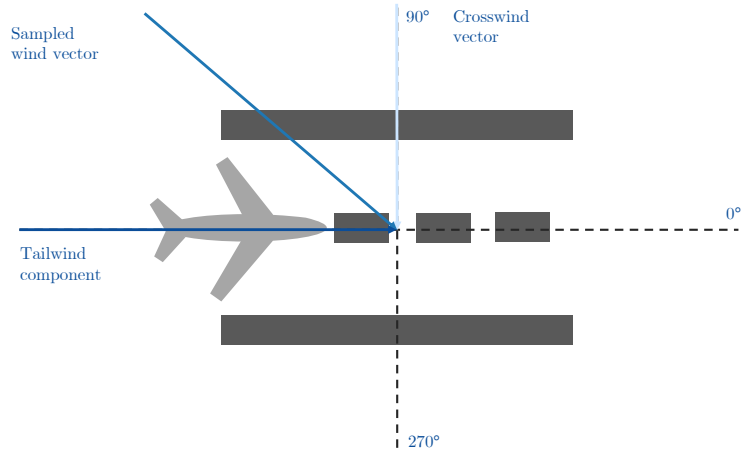

Figure 6.4: Wind direction representation

Since many airports have many runways and two orientations per runway, a method was create to compute a runway suitability score that guides the selection of the best runway for landing under given weather and aircraft conditions, mimicking how real-world ATc would advise the pilots where to land. The overall scoring function combines factors related to the available runway margin, wind components, and special equipment such as an instrument landing system (ILS). Formally, the runway score S for each runway candidate is calculated as:

$$S = M + H - T - X + E + I \tag{6.4}$$

where M denotes the margin bonus, computed as the difference between the available runway length and the total landing distance required,

$$M = L_{\text{available}} - D_{\text{required}}, \tag{6.5}$$

and H represents the headwind benefit,

$$H = 50 \cdot \max(0, W_{\text{headwind}}). \tag{6.6}$$

The term T captures the tailwind penalty,

$$T = 200 \cdot \max(0, W_{\text{tailwind}}), \tag{6.7}$$

while X denotes the crosswind penalty, which increases non-linearly with the magnitude of the crosswind component,

$$X = 10 \cdot |W_{\text{crosswind}}|^{1.5}. \tag{6.8}$$

An additional margin-related bonus is included through E,

$$E = 0.1 \cdot (L_{\text{available}} - D_{\text{required}}), \tag{6.9}$$

and I provides a discrete bonus for the presence of an instrument landing system (ILS),

$$I = \begin{cases} 500, & \text{if ILS is present,} \\ 0, & \text{otherwise.} \end{cases} \tag{6.10}$$

This runway scoring function ensures that the selected runway maximizes safety by favoring sufficient runway length, favorable wind conditions, minimal crosswind, and the presence of advanced landing aids.

6.3.3 Fuel calculations

The fuel calculation combines both technical information from the flight manual [13] and the wind fields and interpolation approaches in order to produce estimates of the fuel needed to reach one of the possible alternate airports. Fuel calculation is not only relevant to the range of the aircraft but also important because it influences the weight of the aircraft, which in turn has implications when calculating the required runway distance, as heavier aircraft need more runway for landing.

BADA performance model

Regarding fuel estimation, two specific mechanisms are employed. The simulation begins with the initial fuel from the obtained flight plans and then uses the Base of Aircraft Data (BADA) 4 performance model [201, 1] to estimate fuel usage up to the decision point. BADA, developed by EUROCONTROL, is a performance model that contains standardized performance parameters such as thrust, drag, fuel flow, and climb and descent profiles for many aircraft types. These values are derived from flight manuals and further calibrated with real-world flight data. For fuel simulation, BADA provides fuel flow values depending on altitude, aircraft weight, and speed, which makes it possible to estimate fuel consumption during climb, cruise, and descent in a consistent manner. Because the model is validated against airline operations, it has become one of the main reference points when carrying out large-scale studies of fuel consumption and traffic behavior.

Stepwise simulation

While BADA provides reliable estimates for fuel consumption, it does not take into account wind field modeling and requires flight plans in order to provide accurate results. Therefore, from the decision point to the selected alternate airport, the assumption was made that the aircraft would fly directly. For this part of the fuel estimation, a custom stepwise simulation with a time step of 10 seconds was implemented. The simulation consists of three main components:

1. Fuel calculation for the cruise part of the flight.
   This is the most complex part of the simulation. In order to estimate fuel burn, the fuel burn tables from the flight manual [13] were used alongside linear interpolation. These tables specify fuel burn for specific weights at specific altitudes. To obtain a better estimate of fuel burn in this phase, additional tables were also used and interpolated to determine the true

airspeed (V_a). This then allowed for the use of the wind field to model the influence of wind on fuel consumption through what is known as the wind triangle. The wind triangle is a fundamental navigation tool in aviation that expresses the vector relationship between the airspeed vector (V_a), the wind vector (V_w), and the resulting ground speed vector (V_g). The diagram also illustrates the angles between aircraft heading (ψ), ground track (χ), and the wind correction angle (β). By categorizing the wind into crosswind and head/tailwind components, the simulation determines the necessary correction to maintain the desired ground track and computes the effective ground speed. Since ground speed governs the distance covered per unit of time, and thus the associated fuel burn, wind is fully integrated into the fuel calculation process. A visual representation can be seen in Figure 6.5.

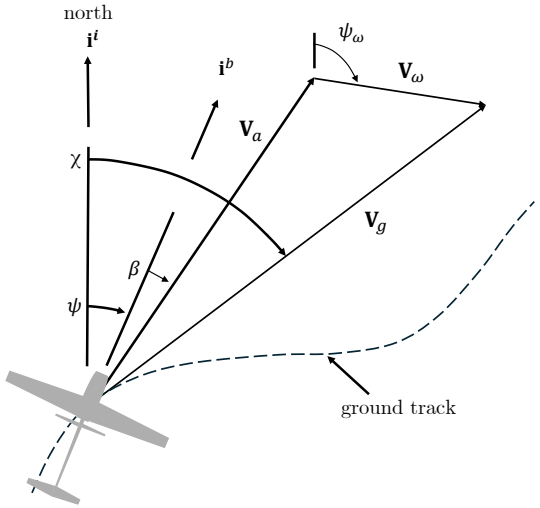

Figure 6.5: The wind triangle. Here, β represents the side-slip angle of the UAV, χ the course angle, ψ the yaw angle, and ψ_w the wind angle, adapted from [94].

2. Fuel calculation for the descent part of the flight.
   This is the fuel needed for the aircraft to perform the descent and land. The descent distance and the fuel required is interpolated from another table based on the flight manual [13].

3. Fuel calculation for fuel penalties and fuel leaks.
   This represents additional fuel usage due to technical failures, which could impose an altitude ceiling on the aircraft, potentially causing it to fly at a lower altitude and therefore consume more fuel. It also includes the calculation of possible fuel leak amounts to be added to the final fuel consumption required to reach the airport.

The fuel estimates from the BADA performance model and the custom stepwise simulation were combined to obtain an accurate estimate of fuel usage.

6.3.4 Technical Problems Modeling

In order to ensure the simulation is realistic and covers a wider range of possible emergency scenarios, a broad set of technical problems was included. Such problems not only represent important safety considerations but also have a direct impact on fuel usage, available altitude, and the overall set of feasible diversion airports. In the simulation, several categories of technical issues were modeled, each reflecting typical operational challenges that can arise in real-world scenarios. The three types of technical failures are as follows:

1. Engine-out condition
   In aviation, the loss of one engine reduces available thrust and imposes strict limits on the maximum cruising altitude. Aircraft performance manuals specify a reduced altitude ceiling for engine-out scenarios in order to maintain safe climb and cruise capability. In the simulation, this is implemented by enforcing the altitude ceiling based on the current aircraft weight, which forces the aircraft to cruise at lower flight levels. Since fuel burn is higher at lower altitudes, this condition directly increases fuel consumption and reduces the reachable set of alternate airports.

2. System degradations (G+B, G+Y)
   Failures of onboard systems, such as degraded hydraulic, pneumatic, or electrical subsystems, can increase fuel usage indirectly by lowering overall efficiency or requiring backup systems to remain active. In the simulation, these are modeled as stochastic percentage penalties applied to the total fuel burn. The presence of multiple degraded systems leads to larger penalties, reflecting real-world operational procedures in which safety margins and redundancy increase fuel demand. These degradations therefore reduce fuel reserves and limit diversion flexibility.

3. Fuel leaks (FL_1, FL_2, FL_3)
   Fuel leaks are a serious safety issue in aviation and can range from minor to critical depending on severity. In reality, leak detection systems and crew procedures mitigate some of the risks, but a leak always reduces the available fuel regardless of performance. In the simulation, fuel leaks are implemented as additional losses on top of normal consumption, with normally distributed leak rates depending on severity. Minor leaks contribute only a small fraction of the normal burn rate, whereas critical leaks can result in significant additional fuel loss, potentially exceeding the planned consumption and greatly reducing the diversion range.

4. Medical emergencies (MED)
   Medical emergencies on board represent time-critical situations requiring expedited landing at a suitable airport with appropriate medical facilities. While not directly affecting aircraft performance or fuel consumption, medical emergencies impose strict time constraints on the decision-making process and may prioritize closer airports over those with better weather or runway conditions.

Together, these technical problems influence both the direct fuel consumption and the set of feasible alternate airports. By increasing burn rates, reducing cruising altitudes, or causing direct fuel loss, they create more restrictive operating conditions that must be reflected in the decision-making scenarios.

Another important aspect is the inclusion of combinations of technical failures, with the most extreme case being an engine-out, a critical fuel leak, two hydraulic failures, and a medical emergency on board. The technical failures are encoded in the technical status factor, which belongs to the group of context factors.

Technical status encoding

In order to ensure the technical failures are usable within the machine learning framework, a hierarchical encoding scheme was implemented. Each technical problem is mapped to a unique numerical representation using a custom encoder that separates fuel leak severities into tiers and encodes the remaining failures as binary flags. For example, fuel leaks are represented hierarchically as FL_1, FL_2, and FL_3 with increasing severity, while non-fuel issues such as engine-out, system degradations (G+Y, G+B), and medical emergencies are encoded as bit flags. The encoder combines these into a single value, ensuring that all possible combinations are uniquely represented and preserving the natural dominance hierarchy of the fuel leak severities (i.e., FL_3 always encodes higher than FL_2 or FL_1). This structured encoding is particularly useful for training the models since it allows the technical status factor to be directly

integrated into the decision-making process while ensuring the space of possible failures is consistent and interpretable. In contrast to a purely categorical one-hot representation, the hierarchical encoding ensures that the values also carry ordinal meaning when normalized. For example, fuel leak severities form a natural progression from FL_1 to FL_3, rather than being treated as unrelated categories. This ensures the encoding is more compact, avoids the overhead of many binary flags, and provides the models with a meaningful numerical structure that reflects the severity and combinations of failures. An example of this can be seen in Table 6.3.

Problem Combination	Encoded Value	Explanation
None	0	No problems
MED	1	Medical emergency only (bit flag)
G+Y	2	Minor system degradation (bit flag)
G+B	4	Major system degradation (bit flag)
ENG_OUT	8	Engine-out condition (bit flag)
FL_1	16	Minor fuel leak (tier 1)
FL_2	32	Moderate fuel leak (tier 2)
FL_3	48	Critical fuel leak (tier 3)
FL_2 + G+B	36	Moderate fuel leak (tier 2) + major system degradation
FL_3 + ENG_OUT + G+Y	59	Critical fuel leak (tier 3) + engine-out + minor degradation

Table 6.3: Examples of hierarchical encoding of technical problem combinations. Fuel leaks are ordered in severity tiers, while non-fuel failures are combined as bit flags.

6.3.5 Scenario Generation Process

The scenario generation process consists of several steps that result in a scenario that can be passed through the scoring function and used as part of the dataset to train the models.

The process can be subdivided into two distinct parts; one pertains to the scenario as a whole and another is performed for each option.

Regarding the first part the steps taken are as follows:

1. Scenario type selection
   This is the initial step undertaken. Scenarios are defined according to two key dimensions:

   - Weather conditions ranging from very calm to catastrophic.
   - Technical failure severity ranging from very light to critical.

   Each scenario type is a combination of these two dimensions, creating 60 possible scenario types (10 weather types and 6 failure severity types), which can be seen in a tabular format in appendix A. Based on this, the weather phenomena are defined alongside possible modifications to the wind speeds and the number and types of technical failures.

2. Flight plan selection
   From the many possible flight plans available, one is randomly selected. This flight plan provides information on the route, waypoints, altitude, and initial starting fuel. From the possible waypoints, one waypoint belonging to the cruise phase is selected as a decision point. The BADA performance model is then used to calculate fuel usage up until that point, providing reasonably accurate weight and fuel-onboard estimates.

3. Possible options determination
   Having identified a decision waypoint, the n closest airports are calculated. It is also important to note that in order for the models to learn

how to rate the options based on their characteristics, the list of the closest possible options was shuffled so that the closest option was not always in the initial position.

Continuing with the second element, which was employed for each option:

1. Options for weather and wind charactersitics determination
   Taking into account the weather aspect from the scenario type and estimating the best runway accordingly, the crosswind and tailwind aspects are defined. The weather type influences if and by how much the wind vector is modified from the original wind file. Some scenarios simply use the original wind field, while others increase the wind intensity but still maintain a reasonably similar wind direction to a real-world wind field. Other scenarios also modify the wind direction in order to showcase the effect of sporadic winds. An important caveat here is that this is another simplification regarding the wind and weather phenomena modeling performed. In reality, wind field simulation, and even simulating augmentation of wind speed, could cause more complicated outcomes across the entire wind field, so it is important to note the simplification used in the scenario generation procedure. Additionally, based on the weather type, appropriate values for the runway condition, visibility, and cloud ceiling at each airport are specified.

2. Fuel needed and runway length required estimation
   An aircraft object is created with the weight and fuel established by the BADA performance model, and is placed at the altitude and position of the decision waypoint from the flight plan. For each option, a direct flight is simulated to estimate the fuel needed using the stepwise simulation. Here, it is important to note that the scenario weather changes that influence the wind at ground level for the options, which may have been increased or altered from the original wind field, are additionally adjusted. The adjustment involves calculating an adjustment factor based on the ratio of change across all airports. This factor is then used to augment the wind field across all flight levels, which corresponds to a more accurate wind representation and better estimates of the fuel needed to reach the destination. After the fuel burn is calculated, the weight is adjusted based on the fuel usage, and this information, along with the crosswind, tailwind components, and runway status, is used to estimate the runway length required.

3. Key distances estimation
   Distances to the aircraft, the original destination, and the nearest hub are also estimated in nautical miles. Additionally, a contingency factor is estimated, which is the average distance of the three closest airports. This acts as another safety net or backup plan, with lower values being preferred since there should be another airport in the vicinity if something happens to the chosen option.

The described pipeline enables the creation of a wide variety of scenarios. The way wind and weather effects are incorporated and modeled represents an improvement over the initial dataset and brings the simulation closer to real-world conditions. The central element of the simulation, is without doubt, the fuel calculation, since the weight of the aircraft is such an important factor for many other aspects of the options. This required a detailed stepwise simulation together with wind modeling and adjustments in order to realistically capture the interaction between fuel use, aircraft performance, and environmental conditions.

6.3.6 Error Modeling and Uncertainty

The scenario generation also includes an error modeling component that introduces uncertainty into the weather information, corresponding to expected

deviations for certain weather conditions. This is intended to reflect the fact that in real-world operations, decisions are never made with perfect knowledge of the environment. The errors are defined by structured ranges that correspond to the severity of the weather type. Four levels are used: no phenomena with very small deviations, light, medium, and heavy, with each level specifying progressively larger ranges of possible error. During scenario generation, the assigned weather type determines which range is active, so that more severe weather conditions are associated with higher levels of uncertainty. This reflects real-world conditions since, during storms or extreme weather phenomena, it becomes increasingly difficult to obtain accurate estimates.

The errors are applied in the calculation step to the following:

- Wind speed.
- Wind direction.
- Visibility.
- Cloud ceiling.

For each parameter, a random error percentage is drawn from the defined range and then used to modify the original value either upwards or downwards. Wind direction is approached in degrees so that the change directly corresponds to angular deviation. As an example, a wind speed of 20 knots under a medium error setting might be altered to anywhere between 16 and 24 knots. Since wind speed and direction are directly linked to other calculations, the wind field itself is also modified by these changes. This, in turn, affects the subsequent fuel calculations, as well as updates to crosswind and tailwind components and the estimates of runway length requirements.

6.4 Summary

This chapter described the full data generation pipeline used to create realistic decision scenarios for the DAAS problem and provide training data for the ExACT-MCDM methodology. The scope and assumptions were made explicit. A unified expert role was adopted by the author for practicality while still incorporating feedback from colleagues and pilots and aligning with technical manuals. The focus was limited to the Airbus A321 to ensure the setup remained realistic and manageable. Decisions were modeled as snapshots, with outputs assumed valid for a short operational window, and the option set per scenario was restricted to the five closest airports to reflect pilot-expressed preferences.

The factors were defined and grouped into context- and option-specific categories, with transformation functions and soft and hard constraints specified to encode expert intent. Scenario generation combined real flight plans from EUROCONTROL, airport information parsed from the X-Plane database within a European bounding box, and wind fields taken from the MERRA-2 reanalysis. Wind at altitude was approximated with a power law profile and spatial interpolation. A runway scoring function balanced runway margin, crosswind and tailwind components, and ILS availability. Fuel requirements were estimated by combining BADA-based consumption up to the decision point with a custom stepwise simulation that integrated wind effects through the wind triangle and accounted for descent fuel, penalties, and possible fuel leaks. Technical failures were modeled across several classes and encoded with a hierarchical scheme so that severity and combinations could be used directly by learning algorithms.

Uncertainty was introduced through structured error ranges applied to weather-related variables. These errors propagated through the wind field and into downstream calculations, such as fuel burn and runway requirements, yield-

ing multiple datasets that reflect different levels of operational noise. Taken together, the pipeline delivers a diverse and realistic set of labeled scenarios.

7 Experimental design

This chapter outlines the experimental setup used to evaluate the model selection and training performance part of the ExACT-MCDM methodology and provide insight into one of its key steps. The chapter begins with an overview of the experimental design, outlining the idea behind the experiments and factors of interest. Following this, the models included are presented together with the reasoning for their inclusion and the adaptations required to convert some of them into a form suitable for a supervised ranking task. The chapter then specifies the dataset generation process and the data used to train the models, after which the relevant metrics considered in the experiments are presented. This includes performance metrics and measures concerning feature importance and explainability, as well as robustness metrics. The chapter concludes with details on the experimental execution, covering training and hyperparameter tuning.

7.1 Overview of Experimental Design

7.1.1 Experimental Aim

Given the proposed methodological approach designed to tackle many of the identified problems regarding emergency time-critical MCDM, the goal now is to test it using real-world use case from the field of aviation and observe the outcome.

To accomplish this, various models will be applied to the same datasets and compared in order to gain greater insight into how certain models perform. For the purposes of this thesis, performance is defined by three relevant aspects:

1. Raw model performance
   This refers to the model's performance on standardized performance metrics, such as accuracy. It is of paramount importance to compare how well the models learn and to gain an understanding of a reference point, since this problem is new and has not been addressed until now.

2. Interpretability and Explainability
   This extends the goal of a transparent process in which model interpretability and explainability are vital and more understandable and easy-to-comprehend models are greatly preferred. This includes an additional dimension to be compared alongside performance, to determine the level of performance that would be lost, if any, when applying more suitable models, and whether such a trade-off is considered acceptable. This element also related to feature importance estimation, which is a natural inclusion given the connection to MCDM whereby weighting factors are

central to the decision-making process.

3. Robustness
   Robustness is the final aspect and, given the use case and the variability of the real world, it is an important consideration. To this end, four datasets were created with varying levels of error and variability to help understand how different models would deal with uncertainty and how it would affect their performance.

7.1.2 Interpretability and Explainability

Having defined what performance entails, a clearer distinction is needed regarding interpretability and explainability, as their definitions are often used interchangeably and are not aptly standardized within the literature itself. Moving forward, it is important to draw a sharper line between the two in order to avoid confusion and enable a more granular discussion about specific model characteristics.

Interpretability refers to the degree to which a model can be directly understood by a human without the need for additional tools or approximations, as in the case of a weighted sum or a small number of shallow decision trees. It can be surmised that these models can be reasonably understood and comprehended by a human.

Explainability refers to the degree to which a model's reasoning can be uncovered or approximated through the use of external techniques, such as SHAP, even when the model itself is too complex to be directly grasped. Put simply, while some understanding of the model's internal workings can be obtained, it is unlikely that a human could reasonably derive this understanding without the use of specific techniques separate from the model itself.

This distinction will guide the discussion and evaluation of models throughout the work.

7.1.3 Machine learning approaches

Since the dataset generated has scores calculated based on the evaluation function, the training of the models can be specified as a five-class classification task, where the option with the best class is the correct choice for each scenario. This changes the problem into a straightforward classification task that is quite common in the field.

However, during preliminary testing and while reflecting on the problem, certain issues with the classification approach became clear. More specifically, the classification setup is marred by the model input shape and by scalability issues in terms of the number of options.

Increased complexity

The input shape problem arises from the way typical classification models expect inputs, namely as a single vector. This means that the tabular representation of the problem had to be unrolled into vector form, with the context features at the beginning of the vector. This was not entirely unexpected and was one of the reasons the LCS was considered, especially given its performance on the Multiplexer problem, which has a similar structure and was therefore used to create the benchmarking problem in Chapter 4. Still, this introduces extra complexity as the model must learn many more intricate relationships, resulting in a disadvantage because the input form is not naturally suited to the task.

Increased scalability

This issue also relates directly to scalability. With the unrolled structure, each additional option does not increase the input size by just one, but rather by the full number of features describing that option. This creates a severe scaling problem that drastically increases complexity. A further complication emerges in feature importance extraction, since the same features are spread across multiple positions in the vector and must later be averaged or reintegrated to

recover the importance of a specific feature. The loss in interpretability is especially visible for the LCS because the rules are induced over the long flattened input and end up mirroring the unrolled vector rather than expressing clean conditions on individual factors. As a result, the expected rule pattern, where each classifier carries clear bounds for each feature, becomes diluted across many positions, which makes the population harder to read and undermines one of the main reasons to use LCS in the first place.

For these reasons, although classification may first appear to be a perfect fit, in practice, it is less than ideal.

Ranking formulation

Contemplation regarding convolutional neural networks and how their architecture is tailored to image processing, where shared filters move across the image to capture local patterns, inspired thoughts about how to tailor the input for the models so that it remains in its appropriate tabular form. It was this line of thinking that led to the consideration of ranking as a suitable option for this problem. Ranking allows for the options to be considered as a group, which, in a way, maintains their truer form. Of the three ranking formulations—pointwise, pairwise, and listwise—the first two were discarded after preliminary tests. The pointwise approach did not consider direct comparisons between options and produced poor results, while the pairwise approach did not address the issues with the LCS, which would require rules that mix features from two options with features representing their differences, increasing complexity and reducing interpretability. Therefore, the clear course of action was to model it as a listwise ranking problem. By doing so, the input problems would be solved, models would have less complexity to address, scaling could be improved, and ranking metrics and approaches were more sensible in cases where the top options can be in very close vicinity to each other and are more viable for MCDM problem configurations.

7.2 Scenario Data

7.2.1 Dataset Description and Perturbation Design

Regarding the inputs of our model, four distinct datasets were created with 250,000 scenarios each. Since this is an initial investigation of the methodology, we made a slight departure regarding the label generation process. Instead of generating multiple perturbations per scenario and using the associated stochastic dominance aggregation function, only one perturbation for each generated scenario was performed and used in tandem with the evaluation function to produce labels for the options. This strategy was performed for two main reasons:

1. Reduction in computational time and cost.
   The custom Python simulation used for scenario generation required substantial computation, particularly for large-scale experiments. This reinforces the value of preparing variations and simulations in advance to ensure timely decision support. It also reflects an inherent trade-off in such systems: exploring a sufficiently rich set of scenarios while maintaining precise option estimates, analogous to the exploration–exploitation balance in reinforcement learning.

2. Greater control over error injection.
   The scenario generation process involves simulating complex and interdependent factors such as wind, fuel usage and its influence on weight, and other weather phenomena. Using a single perturbed scenario for each base case allows for a more controlled and direct incorporation of error levels.

Single-Perturbation Sampling as Monte Carlo Approximation

Moreover, this single perturbation approach is not without statistical founda-

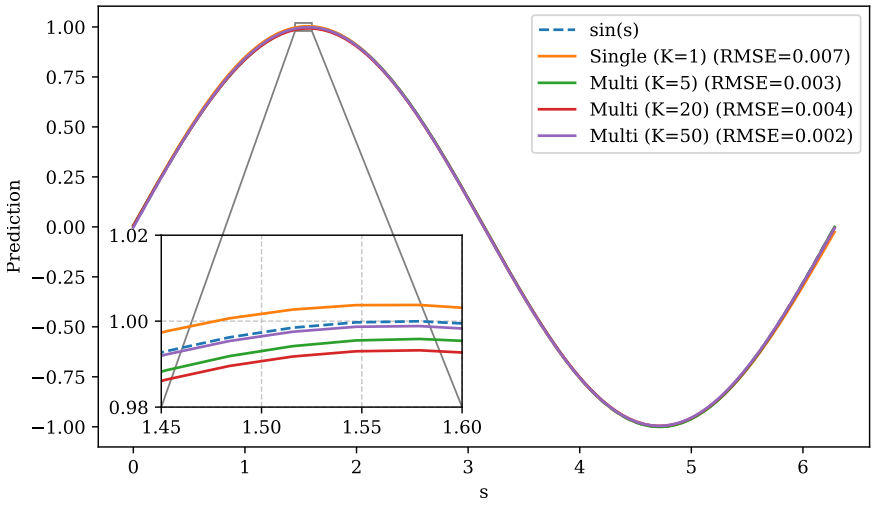

Figure 7.1: Comparison of learning outcomes under single- and multi-perturbation sampling using Liner Regression with Ridge Regularization

tion. In this context, each scenario s represents a complete operational context (e.g., environmental conditions, aircraft state, traffic situation), while each perturbation ξ captures a bounded source of uncertainty affecting that scenario (in this case, weather variation). Using one structured perturbation per scenario therefore generates paired samples (s, ξ) drawn from the joint distribution $p(s, \xi)$. From a Monte Carlo methods perspective [145, 238], empirical estimates formed from many such independent draws should offer a similar learning signal as those provided by many repeated variations if the number of samples from the joint distribution is great enough.

To validate this, a small controlled experiment was conducted, the results of which can be seen in Figure 7.1. In this experiment, a linear regression model with ridge regularization was trained under a fixed simulation budget of $T = 10{,}000$. The model uses a simple nonlinear feature map consisting of a bias term 1, a linear term s, a quadratic term s^2, and the trigonometric terms $\sin(s)$ and $\cos(s)$. Using these features, the predictor takes the form

$$\hat{f}(s) = w_0 + w_1 s + w_2 s^2 + w_3 \sin(s) + w_4 \cos(s). \tag{7.1}$$

The goal was then to compare the difference when the models is is trained using many scenarios with one perturbation each ($K = 1$), and fewer scenarios with multiple perturbations averaged per scenario ($K > 1$). All approaches recovered the same expected function $E[f \mid s]$, with the multi-perturbation variants showing only a modest reduction in variance. The qualitative shape of the learned function and the quantitative RMSE values were highly similar across all values of K, demonstrating that the broader coverage of the joint scenario–uncertainty space (s, ξ) obtained through $K = 1$ sampling can provide a sufficiently strong learning signal. Thus, it could be argued that by increasing K, the primary reduction is in terms of noise and not complete alteration of the underlying objective.

Based on the experiment, one can assert that this inherently increases the difficulty of the problems the model faces, as a single perturbation could represent a rather large departure from the initially presented case. However, due to the manner in which the simulation operates and the method by which noise levels are introduced, the effects should not be tantamount to random noise injection; rather, they remain, to a greater extent, bounded by the scenario types and the inherent instability represented by weather components in the scenario. This is also reminiscent of the origins of the approach as single-step reinforcement learning problem in which the model obtains approximation of the factors to make its decision and then receives a reward based on the true outcome.

However, the use of a single perturbation was limited to these initial testing stages to examine the methodology and to assess how the models handle

controlled error levels in a systematic way. For future practical use, multiple perturbations per scenario and their aggregation with an aggregation function inspired by stochastic dominance must be undertaken.

7.2.2 Dataset Characteristics and Error-Level Behavior

For each dataset, each scenario based on its scenario type was subject to four categories encompassing different error ranges:

- No error.
- Light.
- Medium.
- Hard.

The various error ranges for the four different datasets created are presented in Table 7.1.

Weather Category	No Error	Light	Medium	Hard
No Phenomena	(0.0, 0.0)	(0.01, 0.05)	(0.02, 0.10)	(0.10, 0.15)
Light	(0.0, 0.0)	(0.05, 0.10)	(0.10, 0.15)	(0.15, 0.25)
Medium	(0.0, 0.0)	(0.10, 0.15)	(0.15, 0.20)	(0.25, 0.35)
Hard	(0.0, 0.0)	(0.15, 0.20)	(0.20, 0.30)	(0.35, 0.50)

Table 7.1: Error ranges across different dataset types, expressed as (min, max) values.

Since the final evaluation is based on the perturbed scenario, which is meant to represent an amalgamation of many possible simulations, the following tables show the class distribution and changes that occur compared to evaluating the originally generated scenarios.

No Error Dataset

Option	No Errors % (count)	With Errors % (count)	Change
0	20.1% (52,242)	20.1% (52,242)	+0.0%
1	20.0% (52,022)	20.0% (52,022)	+0.0%
2	20.0% (51,885)	20.0% (51,885)	+0.0%
3	19.8% (51,477)	19.8% (51,477)	+0.0%
4	20.0% (51,973)	20.0% (51,973)	+0.0%

Table 7.2: Best option distribution percentage for the No error dataset

Light Dataset

Option	No Errors % (count)	With Errors % (count)	Change
0	19.9% (51,326)	20.0% (51,420)	+0.0%
1	20.0% (51,408)	20.0% (51,374)	-0.0%
2	20.0% (51,549)	20.0% (51,460)	-0.0%
3	20.0% (51,464)	20.0% (51,371)	-0.0%
4	20.0% (51,581)	20.1% (51,703)	+0.0%

Table 7.3: Best option distribution percentage for the light dataset

From / To	Rank 0	Rank 1	Rank 2	Rank 3	Rank 4
From 0	85.7%	10.3%	2.6%	1.0%	0.4%
From 1	10.3%	72.5%	12.3%	3.6%	1.3%
From 2	2.6%	12.4%	68.2%	13.0%	3.8%
From 3	1.0%	3.5%	13.2%	69.4%	13.0%
From 4	0.4%	1.3%	3.8%	13.0%	81.5%

Table 7.4: Rank transition matrix expressed in percentages for the light dataset

Medium Dataset

Option	No Errors % (count)	With Errors % (count)	Change
0	20.1% (50,743)	20.0% (50,410)	-0.1%
1	19.9% (50,114)	20.0% (50,309)	+0.1%
2	20.0% (50,487)	20.1% (50,757)	+0.1%
3	19.9% (50,281)	19.9% (50,151)	-0.1%
4	20.0% (50,531)	20.0% (50,529)	-0.0%

Table 7.5: Best option distribution percentage for the medium dataset

From / To	Rank 0	Rank 1	Rank 2	Rank 3	Rank 4
From 0	59.2%	25.4%	10.3%	3.9%	1.2%
From 1	21.5%	40.4%	23.6%	10.7%	3.8%
From 2	11.2%	19.9%	37.9%	22.1%	8.9%
From 3	5.8%	10.1%	19.9%	42.9%	21.3%
From 4	2.3%	4.2%	8.4%	20.3%	64.8%

Table 7.6: Rank transition matrix expressed in percentages for the medium dataset

Hard Dataset

Option	No Errors % (count)	With Errors % (count)	Change
0	20.0% (56,207)	20.1% (56,329)	+0.0%
1	20.1% (56,295)	20.0% (56,137)	-0.1%
2	19.9% (55,861)	20.0% (55,996)	+0.0%
3	20.0% (56,165)	19.9% (55,919)	-0.1%
4	19.9% (55,922)	20.0% (56,069)	+0.1%

Table 7.7: Best option distribution percentage for the hard dataset

From / To	Rank 0	Rank 1	Rank 2	Rank 3	Rank 4
From 0	54.5%	26.4%	12.0%	5.1%	1.9%
From 1	22.0%	36.1%	24.0%	12.5%	5.3%
From 2	12.7%	20.0%	33.4%	22.8%	11.1%
From 3	7.3%	11.7%	20.0%	38.0%	23.0%
From 4	3.5%	5.9%	10.5%	21.6%	58.6%

Table 7.8: Rank transition matrix expressed in percentages for the hard dataset

Based on the changes, we can calculate a percentage change for the label compared to the originally generated scenarios, shown in Figure 7.2. Additionally, the change in ordering can be estimated by borrowing the NDCG evaluation metric from learning-to-rank methods, which is expressed as:

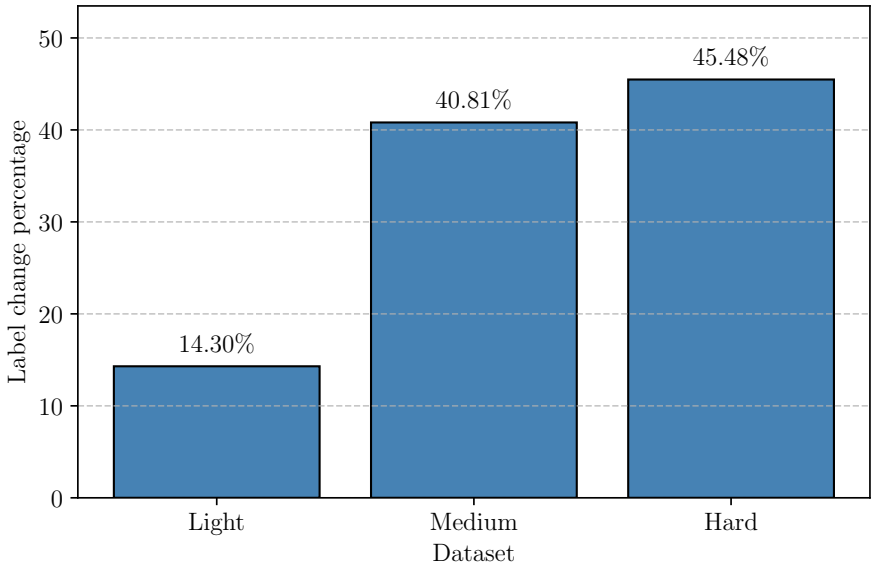

Figure 7.2: Label change

$$\text{DCG@k} = \sum_{i=1}^{k} \frac{rel_i}{\log_2(i+1)} \tag{7.2}$$

$$\text{NDCG@k} = \frac{\text{DCG@k}}{\text{IDCG@k}} \tag{7.3}$$

where rel_i denotes the linearly scaled relevance score of the item at rank position i in the predicted ranking. The denominator $\log_2(i+1)$ introduces a logarithmic discount that reduces the contribution of lower-ranked items. IDCG@k represents the ideal DCG, i.e., the DCG value obtained from the perfect ranking, and serves as a normalization factor. In this configuration, the relevance scores are assumed to have uniform (linear) spacing, consistent with ordinal labels $\{0, 1, 2, 3, 4\}$.

In this way, the NDCG ranges between 0 and 1, where a value of 1 indicates that the predicted ranking is identical to the ideal order, while lower values reflect progressively larger deviations. For the purpose of the evaluation presented here, it allows us to quantify the extent of the change in the ordering of the possible options when scenario modifications—such as altered wind fields or weather conditions—are introduced. The NDCG metric is computed for the entire ranking consisting of all five options, i.e., NDCG@5. The three NDCG values obtained for the dataset are shown in Figure 7.3.

Based on the figures, it is noteworthy that while the error ranges increase significantly between the medium and hard datasets, this does not lead to a proportional increase in the number of changed labels. Instead, the relationship between error severity and label changes seems to follow a sigmoid-like behavior, where initial increases in errors produce stronger changes but further increases lead to a slower growth and eventually approach a plateau.

From these four datasets—the three shown in the figures plus the no-error dataset—the medium dataset is deemed to be closest to the real-world representation of the problem in terms of its error rate and will therefore be given greater focus, with the others serving as a way to understand the effects of differing error rates on the model's performance.

Increased number of options datasets

Taking the medium dataset as representative of the real world, two additional datasets were generated based on it, again consisting of 250,000 scenarios each. These datasets possessed the same level of perturbation but included different numbers of options. The new datasets contain scenarios with ten and fifteen

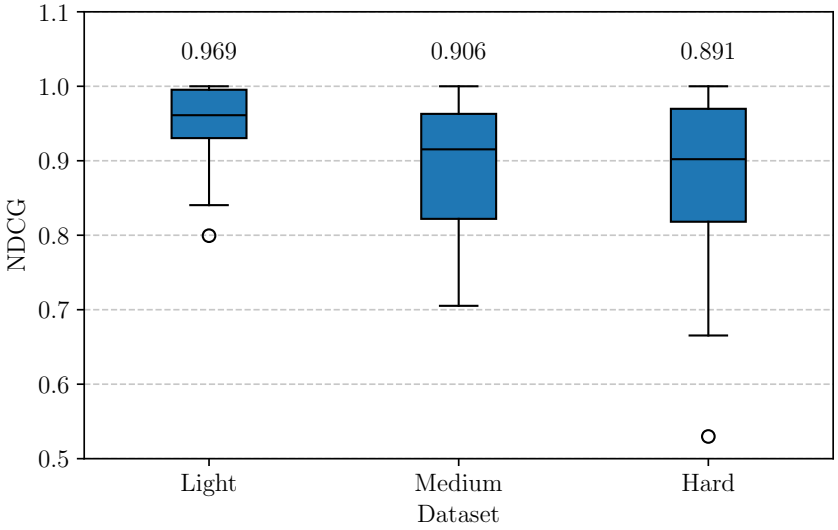

Figure 7.3: NDCG change per dataset

options instead of the original five. These additional datasets allow exploration of how the ranking approach handles an increased number of options.

7.3 Models considered

In terms of the models that would be employed, the following are considered:

1. Weighted sum, as the most explainable and interpretable model that also stems from the field of MCDM.
2. Learning Classifier Systems, due to their explainable and interpretable rule-based structure.
3. XGBoost, due to its top performance with tabular data and ability to handle categorical and real-valued inputs extremely well.
4. Neural Networks, as universal approximators that can be scaled and adjusted as needed.

The models considered were divided into two categories:

1. Simple and interpretable models.

    - The weighted sum model is in line with more typical MCDM approaches. The weighted sum model is considered here is essentially a weight vector whereby, for all scenarios in the datasets, it produces a score for each option and the option, with the best score is the recommended one. This model is inherently the most simple and interpretable by humans as the process of selection is direct and clear.
    - The simple XGBoost model is the second model in this group. This model was limited to 20 trees with a shallow depth of 2. It is slightly more complex than the weighted sum but the restrictions allow it to remain a rather a simple and interpretable model. Here, the interpretability is due to the limited number of trees, which are arguably still understandable by humans.

- The simple LCS model is the final model in this category. It is based on the XCSF approach and uses a normalized least mean squares linear predictor in tandem with a hyperrectangle condition type. The interpretability is also maintained in a similar fashion to the XGBoost algorithm by limiting the population size to 20 classifiers.

2. Performance-oriented models.

- The XGBoost model has been expanded to 5000 trees with a depth of 8. This configuration increases the model's capabilities but decreases its interpretability—with more than 20 trees, it becomes difficult for humans to fully comprehend how the model works in detail. However, it remains explainable to a degree because decision trees can be analyzed with feature importance and SHAP values, even if the overall structure is too complex for a human to follow directly.
- The LCS linear model is the same as the simple LCS but with a much larger population of around 650 classifiers. This reduces model interpretability but given that the predictors are linear it retains a certain degree of explainability when compared to XGBoost because the weight vectors can still be inspected, even if the population size makes the model as a whole harder to grasp.
- The neural network is the final model and the least interpretable of all the models examined. For the purposes of the experiments, the neural network was deliberately limited to two hidden layers of 32 nodes each, placing it at the edge of what could perhaps still be argued as interpretable in some sense. The reasoning behind this design choice was to ensure the network remained shallow and relatively small so that its internal representations might still be studied (for instance, through layer activations or simplified saliency techniques) while also maintaining competitive performance. Yet, even in this reduced form, the model cannot be called interpretable in the same way as the weighted sum or small XGBoost models, and it instead belongs in the category of explainable models whereby external tools such as SHAP are needed to provide insights into the reasoning process.

7.3.1 Model adaptation to ranking

Based on preliminary results, the ranking approach was defined as listwise with linear rankings, in which each option in a scenario received a number from 0 to n indicating its position. This choice was made because the raw scores were difficult to use, especially when the top two options were very similar. Experimentation was carried out by adding a temperature factor to adjust the option values; however, the LCS model in particular struggled to learn under this scheme. Therefore, the more effective solution was to convert the scores into numeric rankings, which provided a more stable and interpretable training signal across all models.

Weighted sum and neural network ranking ranking adaptation

Building on this representation, the weighted sum and neural network models were adapted to a ListNet-style ranking formulation. ListNet optimizes ranking performance through a probability-based loss that converts both the predicted scores and the true rankings into probability distributions via a softmax function. Higher-ranked options receive higher probabilities, and the model minimizes the cross-entropy between these distributions so that it learns to assign larger scores to options that should be ranked higher. While ListNet does not directly optimize NDCG, its probabilistic formulation naturally emphasizes the top positions, aligning well with the goal of achieving high ranking quality. To obtain the predicted probabilities from the linear rankings, a softmax transformation was applied, which proved to yield more stable and better-performing models. In contrast, XGBoost can employ a specialized tree-based objective

that directly optimizes NDCG within its gradient-boosting process. Since the neural and weighted sum models require a differentiable loss and NDCG itself is non-differentiable due to sorting, ListNet provided a suitable and effective alternative.

LCS ranking adaptation

The LCS ranking implementation attempted to employ a newly developed approach to approximate NDCG optimization through reward shaping. Instead of attempting to optimize NDCG directly, which is non-differentiable due to the sorting operation, the idea was for the system to uses finite differences to estimate each option's contribution to the overall ranking quality. Thus, for each scenario, the LCS would first generate scores for all available options using a context-to-option concatenated input format. Afterwards, a baseline NDCG score would be calculated and systematically perturbs each option's score by a small ϵ to measure its marginal contribution to the ranking quality:

$$\frac{NDCG_{new} - NDCG_{current}}{\epsilon}. \tag{7.4}$$

These marginal contributions are normalized using z-score standardization and mapped to the $[-1, 1]$ range using a tanh function to ensure stable and well-distributed rewards across scenarios. This normalized reward signal would then guide the LCS's learning process, where individual classifiers receive rewards proportional to their contribution to the ranking quality, and their prediction weights are adjusted using normalized least mean squares (NLMS). However, this implementation was not successful, primarily due to slowness stemming from the many prediction calls required for each scenario and the subsequent calculations and updates for each individual call. This led to training times of approximately 14 days for just one fold of the 5-fold cross-validation. Additionally, during preliminary testing it was not evident that this new approach produced favorable outcomes, as the models struggled to show consistent improvement.

Therefore, to adapt the LCS for a ranking approach, a modified pointwise method was employed. The task then became one of regression in which the model was called upon to predict a score for each option. For training of the model, all the options were assigned scores based on their rankings within their scenarios and then normalized to $[0, 1]$, thus resulting in normalized targets or scores $[1, 0.75, 0.5, 0.25, 0.0]$. Here, importantly, the options are all fed as individual data points and not grouped as part of a scenario. However, since the scores were derived from scenario-level rankings, a degree of ranking information remained implicitly encoded despite the pointwise formulation.

7.4 Metrics and Execution

7.4.1 Metrics

Performance metrics

The performance metric across the dataset represents the primary way to gain insight into how well the model is able to learn the task at hand.

Given that the initial formulation of the problem explored is that of a supervised learning five-class classification problem, the primary metric used to gauge model performance will be the accuracy metric, which was showcased in the scientific fundamentals chapter in section 2.3. This represents a typical and straightforward measure of model performance for a classification problem. However, the focus on accuracy, with the omission of precision, recall, and even $F1$ score outside of hyperparameter tuning, is due to the balanced class nature of the datasets, where they have almost a perfect 20% split across classes. This was additionally confirmed during preliminary testing where the $F1$ score matched the accuracy. Therefore accuracy will be used as a convenient measure for the five-class classification formulation of the problem.

Regarding the ranking formulation of the problem, the NDCG measure was

selected, which is presented both in the scientific fundamentals chapter and also in the previous section to better describe the changes introduced by variability in the datasets. The NDCG metric was chosen due to its flexibility regarding its application as NDCG@n to showcase performance at different positions in the rankings. For this thesis, NDCG@1 and NDCG@5 will be used. This allows insight into both the top-suggested ranked option and the ranking of the five options that would be presented to the pilots. It is important to specify that for datasets with only five options, we essentially evaluate the entire ranking, while for those with ten and fifteen options, only a subset of all possible options is evaluated.

Furthermore, given the more lenient nature of NDCG as it rewards partial ordering and its limit at 1.0, the inclusion of additional metrics to quantify its improvement compared to a random agent was also employed. Specifically, we introduce a normalized improvement metric that measures how much of the possible improvement over random performance each model captures.

The normalized improvement I^n is calculated as:

$$NI = \frac{NDCG^n_{model} - NDCG^n_{random}}{1.0 - NDCG^n_{random}} \times 100\% \tag{7.5}$$

where $NDCG^n_{random}$ represents the theoretical baseline performance, 1.0 represents the maximum possible NDCG score, and $1.0 - NDCG^n_{random}$ represents the maximum possible improvement.

This metric normalizes performances across different numbers of options and shows how much of the "learnable signal" each model captures.

Feature importance and explainability

The feature importance analysis of the models serves three key aspects:

1. Providing insight into the model's internal processes regarding the current dataset and the formulation of the problem being considered.
2. Relating the new ML approach back to the roots of MCDM to understand which factors are considered important and how they are weighted.
3. Gaining insight into how different explainability approaches, both model-specific and model-agnostic, can be used in line with explainable and interpretable time-critical MCDM.

For the LCS method, particularly regarding its highlighted problems in terms of a classification problem formulation, a population-based feature importance calculation was used. This metric combines two specificity and accuracy-weighted contributions per feature. Specifically, for each classifier c in the population of size N, and each feature f, the importance score I_f is calculated as follows:

$$I_f = 0.5 \cdot \left(\frac{1}{N} \sum_{c=1}^{N} S_{f,c} \right) + 0.5 \cdot \left(\frac{\sum_{c=1}^{N} A_{f,c}}{\sum_{c=1}^{N} \text{fitness}_c} \right). \tag{7.6}$$

where the the specificity $S_{f,c}$ of feature f in classifier c is defined as

$$S_{f,c} = 1 - \frac{\text{spread}_{f,c}}{\text{max\_spread}}, \tag{7.7}$$

with $\text{spread}_{f,c}$ being the spread value of the hyperrectangle condition for feature f in classifier c, and max_spread is normalized to 1.0.

The accuracy-weighted contribution $A_{f,c}$ incorporates the classifier's fitness fitness_c as is defined as follows:

$$A_{f,c} = S_{f,c} \cdot \text{fitness}_c. \tag{7.8}$$

The final importance score for feature f is thus the mean of two components. For option features that appear multiple times (once per option), the scores are averaged across all instances:

$$I_{f,\text{opt}} = \frac{1}{n_{\text{options}}} \sum_{o=1}^{n_{\text{options}}} I_{f,o}. \tag{7.9}$$

Finally, all importance scores are normalized to ensure they equate to one:

$$I_f^{\text{norm}} = \frac{I_f}{\sum_k I_k}. \tag{7.10}$$

This combines both the specificity of the feature's matching conditions (how precisely it matches inputs) and its contribution to accurate predictions (weighted by classifier fitness), providing a balanced measure of feature relevance in the LCS population.

For XGBoost, gain-based feature importance was used, where gain captures the average improvement in accuracy understood as the reduction in training loss whenever a feature is chosen for a split across the trees in the ensemble; since this measure is widely used in the literature, it is not expanded on further here. Unlike simple split counts that only tally how often a feature appears, the gain measure reflects the extent to which a feature actually reduces error and therefore provides a more meaningful indication of its influence on the model's predictions. Given the widespread usage of this feature importance method in the literature, it will not be expanded upon further.

Having addressed the model-specific methods to be used, and with the weighted sum included by default, the two main model-agnostic methods considered are SHAP and Integrated Gradients, with the addition of Permutation Feature Importance. All of these were introduced in Section 2.7 of the scientific fundamentals chapter.

Robustness

Many methods exist to measure the robustness of the models. In preliminary testing, the performance drop rate and relative robustness were also examined. However, for the final analysis, only average robustness was used. This allows easier comparison as one needs to examine only a single metric; additionally, it reduces similar insight that could have been derived from multiple different robustness methods. Average robustness can provide insight into how well a model maintains its performance across various error conditions relative to its performance under clean conditions. It is defined as:

$$\text{AR} \;=\; \frac{1}{m} \sum_{i=1}^{m} \frac{P_{\text{err},i}}{P_{\text{clean}}} \tag{7.11}$$

where $P_{\text{err},i}$ is the model performance on the i-th error dataset such as light, medium, or hard, m is the number of error datasets, and P_{clean} is the performance on the no error dataset. A higher value of AR (closer to 1) indicates that the model maintains consistent performance across the error datasets and therefore shows greater robustness.

7.4.2 Execution and Hyperparameter tuning

Execution

The experiment and data generation process was conducted on the Kratos High-Performance Data Analytics (HPDA) cluster at the German Aerospace Center (DLR). All simulations were executed in a controlled environment using Python 3.11.9.

For the experiments, several machine learning frameworks and libraries were employed. For the tuning of the weighted sum, the ADAM optimizer was chosen [144]. Gradient boosting models were implemented using the XGBoost library in Python [57], while neural networks were implemented using PyTorch

[209]. For Learning Classifier Systems, the open-source Python implementation of XCSF available from the `xcsf` GitHub repository was employed [282].

Hyperparameter tuning

Hyperparameter optimization across all models was conducted using the `Optuna` framework [15], with 100 trials per model. Optuna is a Bayesian optimizer that proposes promising settings based on past results and terminates weak trials early, increasing both the efficiency and thoroughness of the search within the trial budget. In practice, we used its Tree-structured Parzen Estimator sampler and an early stopping pruner, and the objective trained each candidate with five-fold cross-validation and returned the mean score to be maximized. For the classification models, the target metric was the $F1$, score and for each trial, the average $F1$ over the five-fold validation was used as the optimization target.

Regarding the ranking models the `Optuna` framework was used again, with 100 trials per model. This optimization also used five-fold cross-validation and maximized the mean NDCG@5 across the validation folds. The hyperparameter tuning and training process was conducted on 200,000 scenarios, with the remaining 50,000 scenarios used as a holdout set for final evaluation.

In the vein of transparency, it must be noted that the LCS model received extra attention during tuning, including hand tuning and trial and error before running the `Optuna` search. Although all models had the same budget of automated trials, the LCS model received additional human time and focus, which should be kept in mind when interpreting its results.

The exact hyperparameter settings obtained for each model are provided in the appendix C.

Model agnostic feature importance implementation

For XGBoost, the SHAP TreeExplainer from the SHAP library [174] was used. It provides native support for tree ensembles and did not require background sampling. Multiclass outputs were reduced by averaging the absolute SHAP values across classes, and attributions were computed on the full holdout set. For the neural network, the KernelExplainer was used with a background set of two hundred random samples and an analysis set of one hundred samples. For LCS, the values were computed using the KernelExplainer with a background set of two hundred samples and an analysis set of one hundred samples. Across all methods, the final importance scores were normalized to sum to one, option level features that appear multiple times were grouped by averaging their occurrences across options, and multiclass outputs were handled by averaging absolute attributions across classes.

Of particular note was the need for a custom implementation of permutation feature importance for the ranking algorithms. This was due to the scenario-based nature of the dataset and the fact that, during ranking, the models employ individual options as inputs within those scenarios. Additionally, within the dataset, each input fed to the model consists of n context features (which are identical for all options within a scenario) concatenated with the option-specific features. To accommodate this structure, the custom implementation ensures that when a context feature is permuted, it is reflected consistently across all options in the scenario, and when an option feature is permuted, it is modified at all positions where it appears across the five options in that scenario.

7.5 Summary

In this chapter, the experimental setup for evaluating the ExACT-MCDM methodology was laid out in a way that mirrors the practical demands of time-critical decision support, beginning with a clear statement of the aims of the experiments and continuing with the selection of models to be tested. The focus was to incorporate and compare simple and interpretable models,

namely a weighted sum, a shallow XGBoost, and a small population LCS, against performance-oriented variants that included a deeper XGBoost, an LCS with a larger population, and a compact neural network. These models were then adapted to a learning-to-rank formulation using listwise training, with a ListNet-style loss for the weighted sum and the neural network, and an adapted pointwise approach for the LCS.

Building on this, the evaluation metrics were selected to reflect both task fit and operational relevance. Accuracy was used for the initial classification framing (NDCG at one and five for ranking quality) and a normalized improvement measure to express gains over random ordering across different option counts, while robustness was assessed through average robustness to quantify degradation from clean to more difficult conditions. Interpretability and explainability were taken into account, with feature importance derived from model-specific methods such as gain for XGBoost and a population-based measure for LCS, and from model-agnostic methods such as SHAP. The chapter concludes by detailing what exact frameworks and implementations were used and how the models were trained and tuned.

8 Results analysis

The following chapter presents the main experimental results of the thesis, based on the models applied to the newly created datasets and an analysis of their outcomes. The chapter begins by examining the results obtained when treating the problem as a classification problem. Second, the results of the ranking formulation are shown and were carried out to address issues identified with the classification-based approach. Third, an analysis is presented where multiple versions of the medium dataset with more options are compared to gain a clearer view of scaling. Finally, an investigation into model selection is included, where selection is treated as an MCDM problem, making use of an a priori sensitivity study along with an examination of the trade-off between performance and interpretability.

8.1 Classification analysis

8.1.1 Performance analysis

Simple models

Given the division of the models into two separate groups, it seems like a natural step to begin by examining the performance of the simple and interpretable models first, as this provides a reasonable baseline. The performance of these models in terms of accuracy is shown in Figure 8.1. From the figure, it is clear that the simple XGBoost model performs the best across all of the datasets, with the weighted sum following in second place. The XGBoost performance is rather strong in the no-error dataset where it achieves an accuracy of 76%, but this performance degrades as the level of noise in the datasets increases. The LCS linear model, on the other hand, proved completely incapable of learning anything meaningful, most likely due to its small population size, and in practice, its performance was close to that of a random agent offering no real benefit. What is more surprising is the performance of the weighted sum, which rivals that of the simple XGBoost model, with the two being almost equal and showing only a 1% difference in the crucial medium dataset. This is most likely due to the fact that the weighted sum is naturally able to conform more directly to the scenario-based format of the problem since it operates as a weight vector of exactly 19 features, with 4 context features and 15 option features, and does not require its input to be rolled out into a table of all options with the context features placed at the beginning, making it a more suitable representation for this specific problem.

Performance-oriented models

Turning now to the performance-oriented models, Figure 8.2 shows the results for the larger LCS linear model, the full XGBoost model, and the neural network. From the figure, it is clear that the full XGBoost model achieves the best performance overall, reaching an impressive accuracy of 91% in the no-error dataset and still maintaining a relatively strong 73% accuracy in the

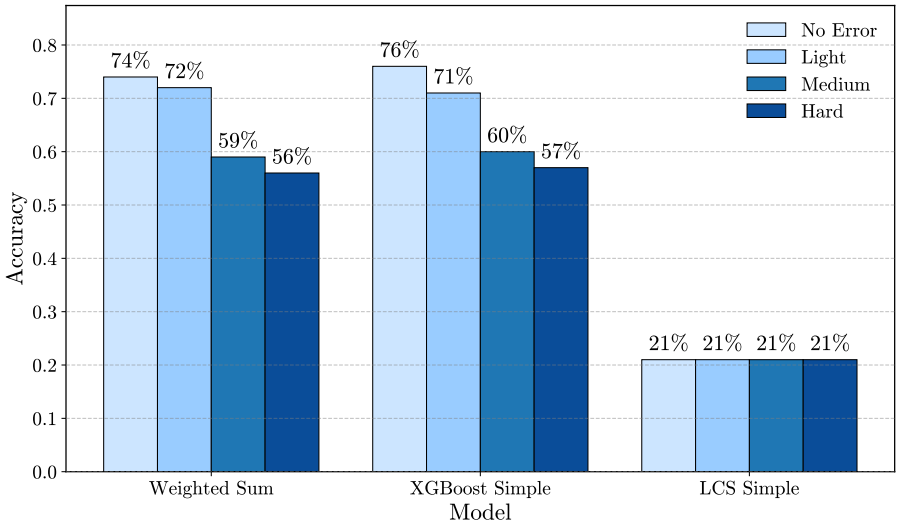

Figure 8.1: Simple model performance

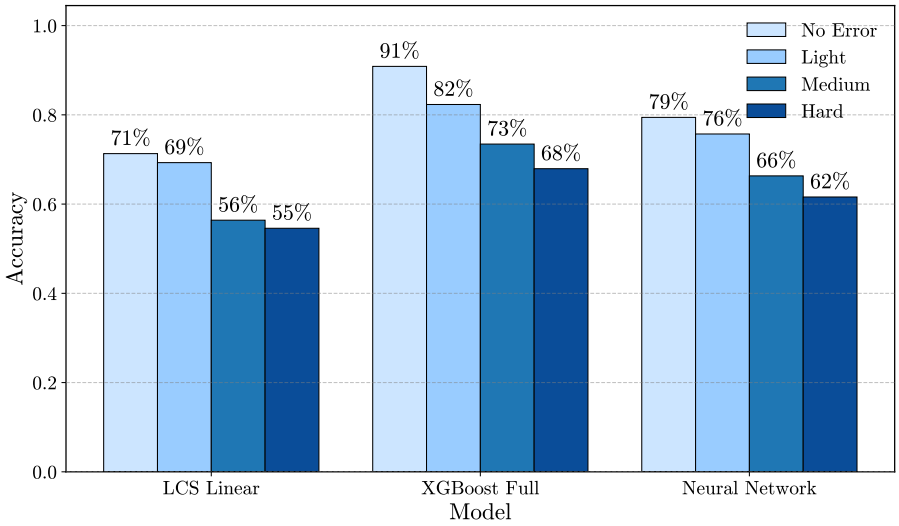

Figure 8.2: Performance-oriented models

medium dataset. This makes it the strongest of all tested models in terms of raw predictive capability, although its interpretability is effectively lost at this scale with thousands of trees involved.

The neural network follows closely behind, with performance that is competitive and, in some cases, surprisingly resilient, showing 79% accuracy in the no-error dataset and 76% in the light-error dataset, before dropping more sharply to 66% and 62% in the medium and hard datasets, respectively. This indicates that the neural network is able to learn useful representations and deliver some solid performance, but its sensitivity to increasing error levels highlights fragility that would need to be addressed before deployment in safety-critical contexts.

The LCS linear model, even when expanded to a larger population size, still struggled to match the performance of the other models. It achieved 71% in the no-error dataset and then quickly dropped to 56% and 55% in the medium and hard datasets, showing that simply increasing population size is not sufficient to close the gap. The linear structure of the predictors may preserve some degree of explainability, but the overall performance leaves much to be desired compared to XGBoost and the neural network.

These results highlight that the performance-oriented models do indeed provide stronger raw accuracy, with XGBoost in particular standing out as the most capable model across varying error levels. However, this comes at the cost of

interpretability, given that the internal logic of the model cannot realistically be followed by human stakeholders once the complexity increases to thousands of trees or dozens of neural network nodes. The lesson here is that while performance-oriented models push the boundary in terms of predictive power, their integration into safety-critical decision-support systems would require the use of explainability methods such as SHAP in order to make their reasoning accessible; even then, there is an open question as to whether the trade-off between performance and interpretability is acceptable in contexts where trust and oversight are essential.

All models

Finally, Figure 8.3 presents the performance of all models across all datasets, allowing for a direct comparison between the simple interpretable approaches and the more complex performance-oriented ones. From the figure, it is clear that the full XGBoost model remains the best performer overall, reaching 91% accuracy in the no-error dataset and showing consistent strength across the others, with only gradual declines as noise levels increase. The neural network follows as the second-best model in terms of raw accuracy, although it shows a sharper drop when error is introduced, while the weighted sum proves to be a surprisingly strong contender, nearly matching the performance of the simple XGBoost model in the medium dataset and outperforming both LCS variants.

The comparison also highlights the weakness of the LCS simple model, which failed to learn anything meaningful and remained at the level of a random baseline across all datasets, and the limited but slightly stronger contribution of the LCS linear model with a larger population, which still could not compete with XGBoost or the neural network. This emphasizes that while LCS models may provide some interpretability benefits, they did not perform competitively in this classification setting, especially when noise was present.

Upon examining all models together, a clear trade-off between interpretability and performance emerges. The weighted sum and simple XGBoost models demonstrate that interpretable models can perform surprisingly well, particularly when the problem structure is more adapted to them, as in the case of the weighted sum.

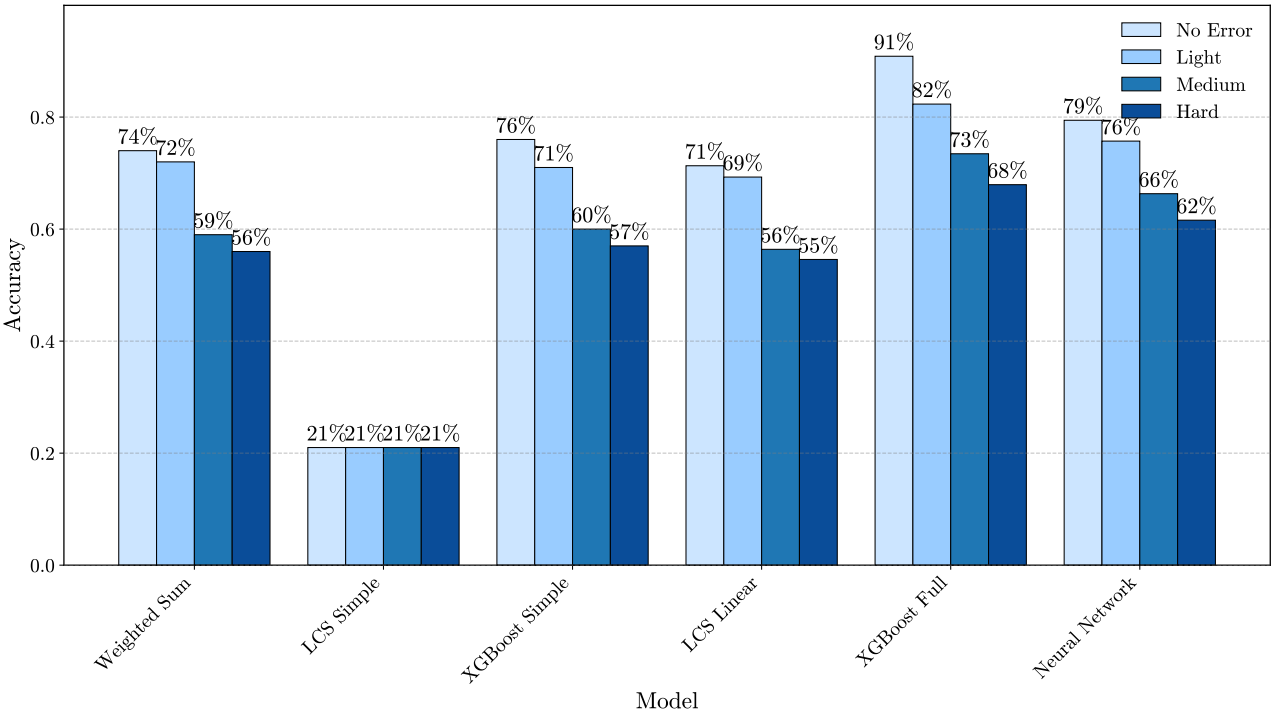

Figure 8.3: All model performance across datasets

8.1.2 Feature importance analysis

The feature importance analysis was a particularly tricky undertaking given the large number and variety of models examined. In an effort to provide an additional comprehensive analysis while still maintaining a suitable level of compactness, the feature analysis was conducted using two heatmaps in which the features are shown on the y-axis, while the methods and specific explainability techniques are shown on the x-axis, which allows direct comparisons methods and models. The idea was to have a model-specific method (or a metric tailored to that method) and also include a model-agnostic method (in this case, SHAP). As mentioned in the previous chapter, the feature importance values from all methods were normalized to a scale of 0–1 so that meaningful comparisons could be made. The analysis presents three separate heatmaps, one in which the values are min–max-scaled, and another in which feature importance is converted to ranks.

Across all the heatmaps, several distinct patterns can be observed. First, the average distance to closest airports, visibility, tailwind, and crosswind are almost universally of highest importance, with the strong result observed for the average distance to closest airports being a particularly surprising inclusion. While this initially appears unusual given that the direct relevance of such a measure is not as obvious as more physical factors like visibility or crosswind, it seems that this feature effectively acted as a measure of geographical density. By capturing how closely clustered the surrounding airports are, it provided information that extended far beyond simple distance. In areas such as Western Europe where airports are dense, it highlighted the availability of nearby alternatives, the redundancy of the network, and even the likelihood of correlated weather conditions across multiple sites. In less dense areas, it reflected the isolation of an option and the risks of limited alternatives. In this way, the feature was not only a distance measure but a composite indicator of spatial redundancy and environmental correlation.

The prominence of this feature across all models shows how a seemingly simple metric can become a hidden combination of influences that provides a great deal of predictive power. At the same time, it raises interpretability questions, since decision makers may not easily understand what such a feature really represents. This highlights both the strength and risk of composite features. They can provide a powerful signal that improves accuracy but can also make the reasoning of the model less transparent. For this reason, the balance between performance and interpretability must be considered directly when designing features and generating scenarios.

The heatmaps seem to indicate that for many models and methods, the most influential factors are those tied to weather and fuel, with the contingency metric—the average distance to the closest airports—making a surprisingly strong showing as well. This largely fits expectations since these are exactly the variables with high variability and clear constraints, yet it also provides interesting insight into the methods themselves because certain approaches appear to concentrate on only a handful of factors and almost dismiss the rest. In the case of the simple XGBoost this was relatively expected, as the narrower trees zero in on the most important signals and effectively eliminate others, which explains the many zeros in that column. In contrast, the weighted sum and LCS population metrics display a more nuanced spread of importance and clarify that the remaining factors, especially those pertaining to context, do carry weight. It should also be noted that most methods had to contend with the feature split across the entire input vector and the subsequent reassembling and averaging, whereas the weighted sum offers a more direct and transparent mapping.

This naturally raises the question of which method should be preferred for explanation. LCS is a good example since different techniques produced different importance profiles. A caveat is that SHAP is a rigorous and well-founded

method, while the population-based metric is the author's own construction and therefore not as established, even if it rests on reasonable logic. This, in turn, raises a broader question about whether explainability in practice requires multiple methods to be applied in parallel, and when those methods offer conflicting insights, which interpretation should ultimately be trusted. It also highlights certain limitations, particularly with SHAP: although SHAP is very effective for explaining individual predictions by showing how different factors influence a specific outcome, it may not always provide a fully satisfactory or reliable global picture of how the model behaves overall.

Overall, the feature analysis does provide an understanding of the models' reasoning; however, it raises many new questions, especially concerning how to handle complex features, such as average distance to the closest airport, which were not initially envisioned to be this important. It also raises the question of explainability methods in general and whether they can be employed sufficiently in a global manner or must be relegated to a more granular view.

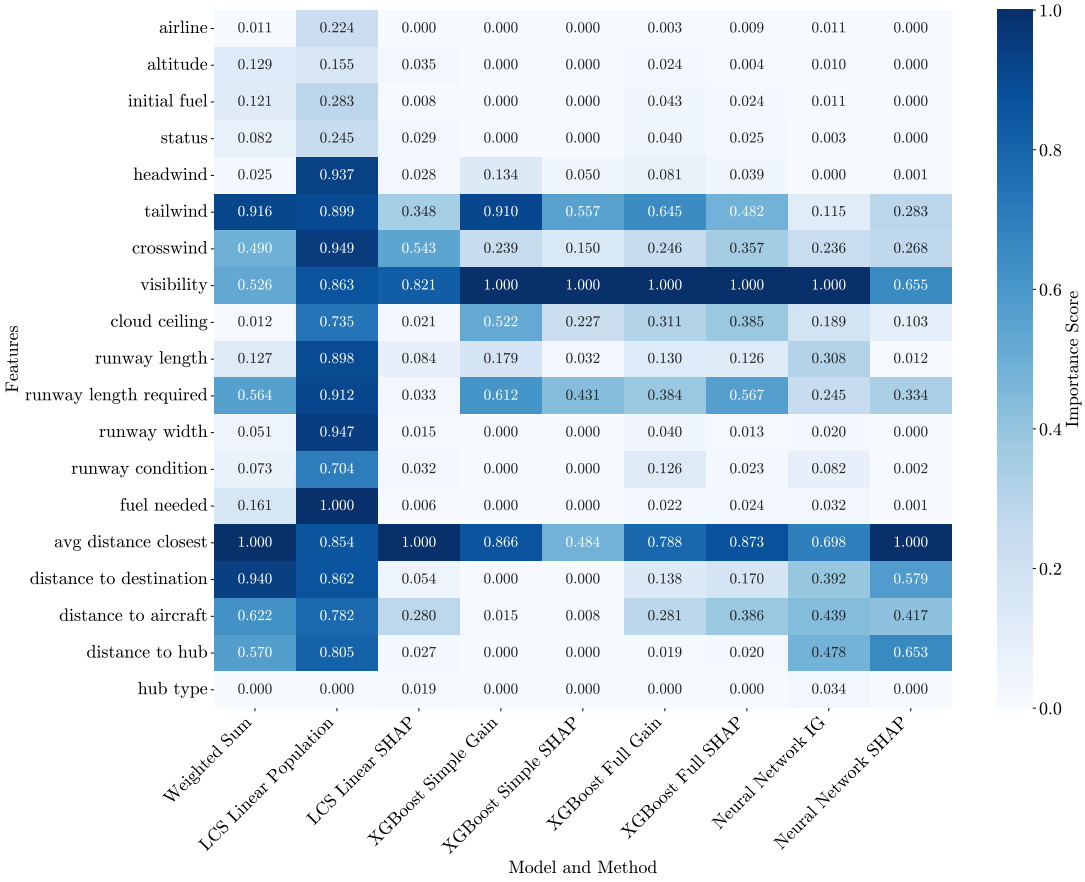

Figure 8.4: Minmax Feature importance, all models on medium dataset

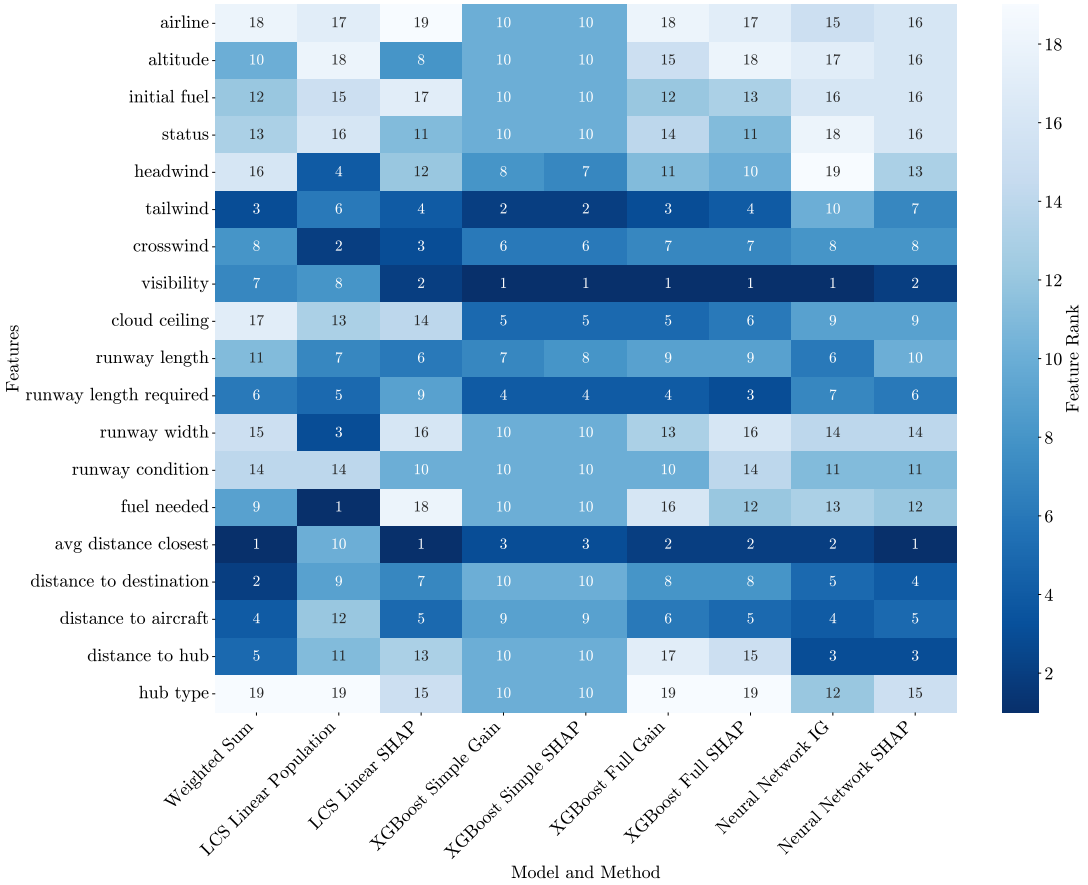

Figure 8.5: Ranking Feature importance, all models on medium dataset

8.1.3 Robustness analysis

The robustness analysis, shown in Figures 8.6, provides insight into how well the models maintain their performance under varying levels of error in the datasets. This aspect is particularly important for time-critical and high-stakes decision-making problems, since models deployed in such environments must be able to handle noisy or uncertain inputs without exhibiting severe degradation in output quality.

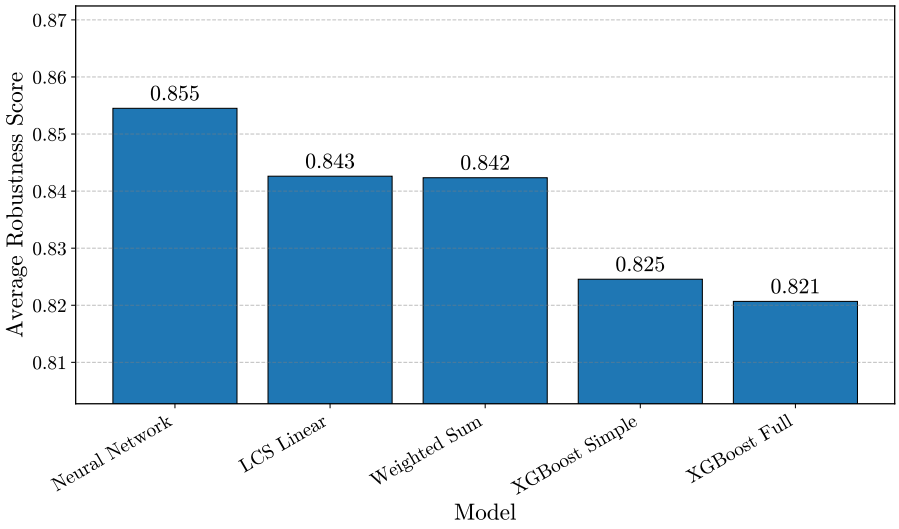

Figure 8.6: Relative robustness scores across models, higher = more robust

Based on the figure, the neural network achieves the highest average robustness score at 0.855. LCS Linear follows at 0.843, and the weighted sum model is similar at 0.842. Both XGBoost configurations are lower, with 0.825 for the

simple version and 0.821 for the full version. These results suggest that models with either inherent flexibility (such as neural networks) or strong structural constraints (such as LCS Linear) adapt better to varying error levels than tree-based approaches.

Based on the analysis, the neural network, despite the relatively constrained architecture used here, proved to be the most robust and stable across different noise conditions. At the same time, the LCS Linear model demonstrates that robustness does not necessarily require complexity, since its relatively simple linear predictors still achieved strong results. The weighted sum model shows reasonable robustness but with clear limitations as noise levels increase, while the XGBoost models, although powerful in terms of raw performance, are more vulnerable to degradation under error conditions.

However, although certain models appear more robust, one need only look at the y-axis of the plots to see that the differences between models seem small. Or do they? As with performance, difficulties arise when attempting to draw firm conclusions beyond simple model comparisons. The fact of the matter is that for problems involving safety, even small percentage differences can carry significant weight once deployed in the real world, and it is especially challenging to fully grasp how these changes might influence real-world operations.

8.1.4 Key takeaways for classification analysis

The analysis showcased both interesting and unexpected insights. In terms of performance, the full XGBoost model outperforms all other models regarding raw performance but becomes less stable as noise increases. The compact neural network is similar shows the strongest stability under perturbations. The weighted sum shows excellent performance given its simplicity, nearly matching the simple XGBoost on the medium dataset while keeping the logic visible. Despite boasting the most interpretable and explainable structure, the LCS variants showed unimpressive results compared to the other models. In the feature importance analysis, across methods, the same drivers continue to appear: visibility, tailwind and crosswind, and distance-based terms, with the average distance to the closest airports acting as a proxy for spatial redundancy and even shared weather patterns, while airline and hub type remain of little importance. Regarding robustness, all models seem to show only small differences, but in a practical safety-critical setting like DAAS, even small differences can carry significant weight. Despite being a less than ideal formulation of the problem, the classification analysis revealed valuable information.

8.2 Ranking analysis

8.2.1 Performance analysis

Simple models

As with the classification analysis, an initial investigation is shown in Figures 8.7 and 8.8. In each figure, the bar height and number depicted above it denotes the mean performance, while the error bars indicate ± one standard deviation. Unlike the classification analysis in which only accuracy was considered, here, we use NDCG@1 and NDCG@5 to show the quality of the most selected option and the entire ranking. As in the previous performance comparison, the XGBoost model performs best, achieving 0.78 on NDCG@1 and 0.92 on NDCG@5 on the medium dataset. It also seems that reformulating the problem as a ranking task suits the models well and increases performance. The simple LCS model is especially impressive, now surpassing the weighted sum and appearing as an excellent contender to the XGBoost model.

Performance-oriented models

The results of the performance-oriented models are shown in Figures 8.9 and 8.10. Here, we see an interesting turn of events in which the simpler neural net-

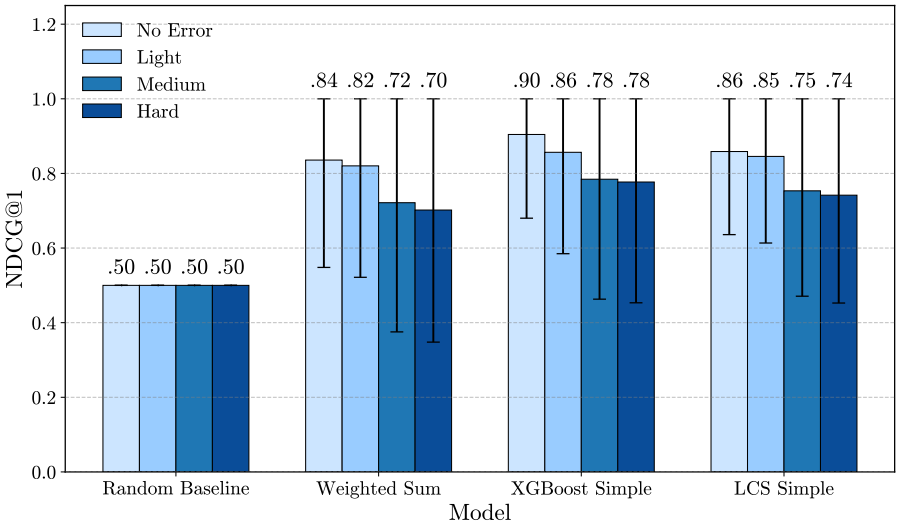

Figure 8.7: Simple models' NCDG@1 performances across datasets

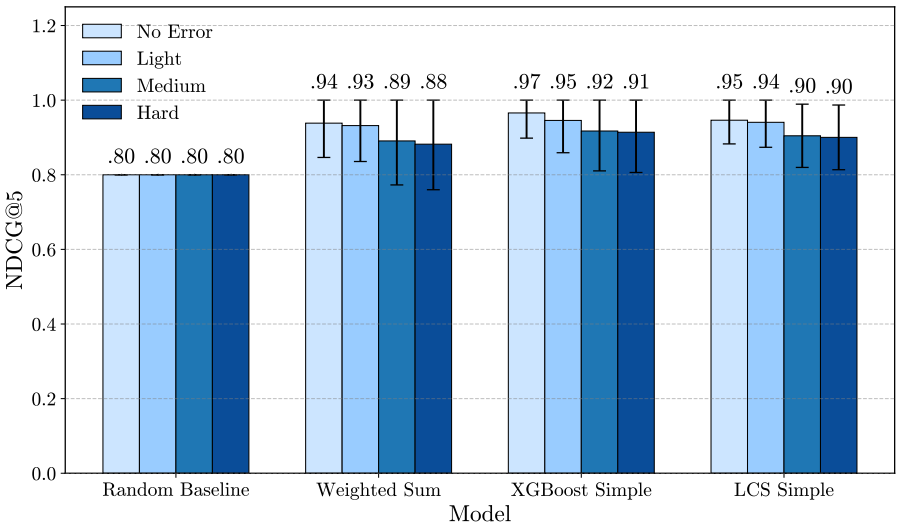

Figure 8.8: Simple models' NCDG@5 performances across datasets

work achieves better results than the XGBoost model. This was an interesting finding since the nature of the dataset, which contains categorical inputs, suggested that XGBoost might remain strongest. In terms of NDCG@1, the neural network reaches an impressive 0.87, which increases to 0.95 for NDCG@5. This shows strong performance, especially when it comes to providing high-quality rankings across all options.

All models' performances

The performances of all models, both simple and performance oriented, are presented in Figures 8.11 and 8.13. The outcome of most interest is the impressive improvement of the simple LCS model, which transformed from a model that learned almost nothing in the classification formulation into being reasonably sufficient in the ranking setup. This result fits well with the relatively strong outcome of the weighted sum, which has certain advantages given the linear constraints used in the evaluation function. In other words, an LCS that uses multiple linear approximators and can divide the space even with a small population gains several benefits.

Additionally, due to the less obvious nature of the linear NDCG metric, the improvement of the models' performances in comparison to random is presented in Figures 8.12 and 8.14 for both NDCG@1 and NDCG@5, respectively. In essence, these plots reflect the performances in terms of the model evaluation.

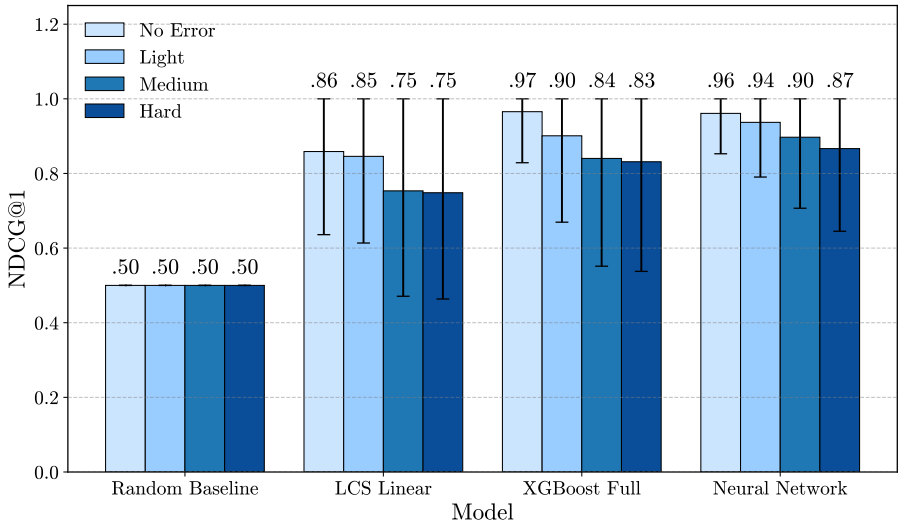

Figure 8.9: Performance-oriented models NCDG@1 performances across datasets

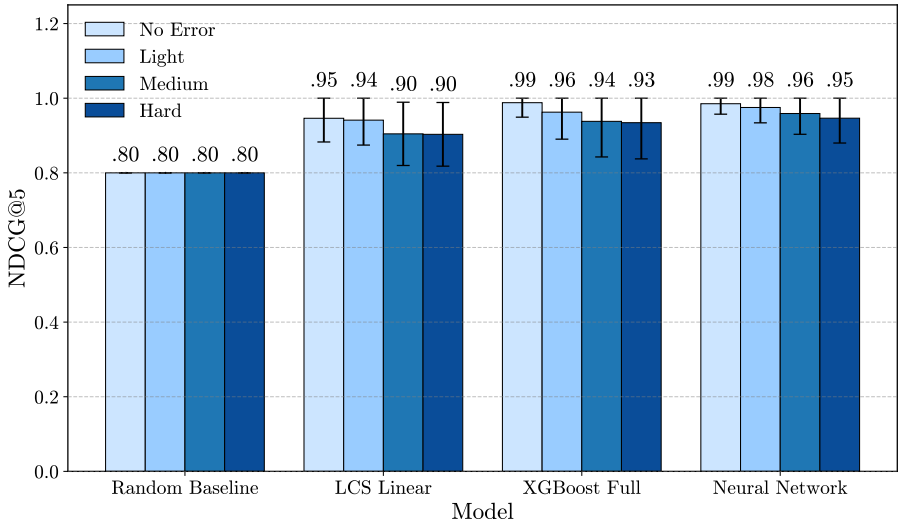

Figure 8.10: Performance-oriented models NCDG@5 performance across datasets

However, they also offer a clearer view of the possible improvement over a random agent. Based on the figures, the dominance of the neural network becomes more apparent than in the pure performance plots. The plots also more clearly evidence the wider gap between the LCS models and the weighted sum vs. the XGBoost models and the full neural network.

Despite not exhibiting the raw performance edge of the neural network or XGBoost, the interpretable and explainable nature of LCS elevates it to the position of a model worth considering. That consideration is further strengthened by the reformulation of the problem as a ranking and the realization that accuracy or a focus on the single top choice are not the most suitable metrics, especially when the best options can be very close to each other or reflect different trade-offs. Therefore, reformulating the problem as a ranking task is not only more suitable in terms of performance gains but also aligns better with a decision-support setting in which a human decision maker remains responsible for the final choice.

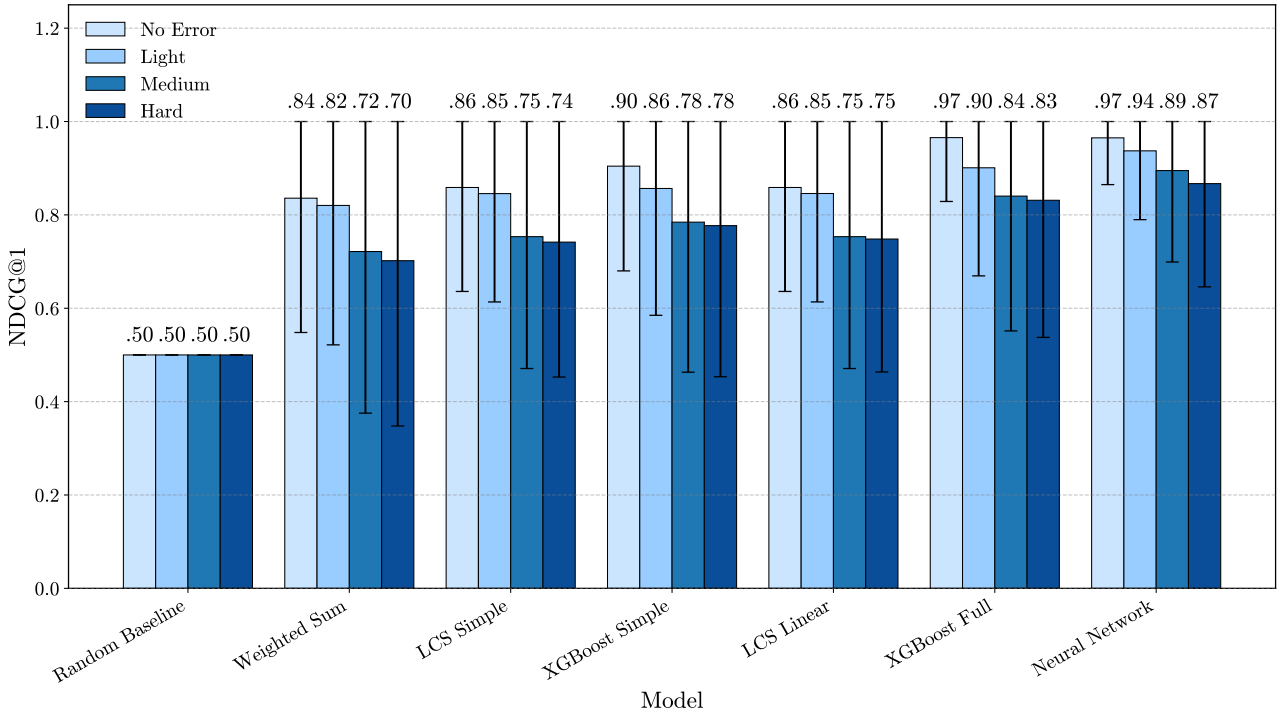

Figure 8.11: All models' NCDG@1 performances across datasets

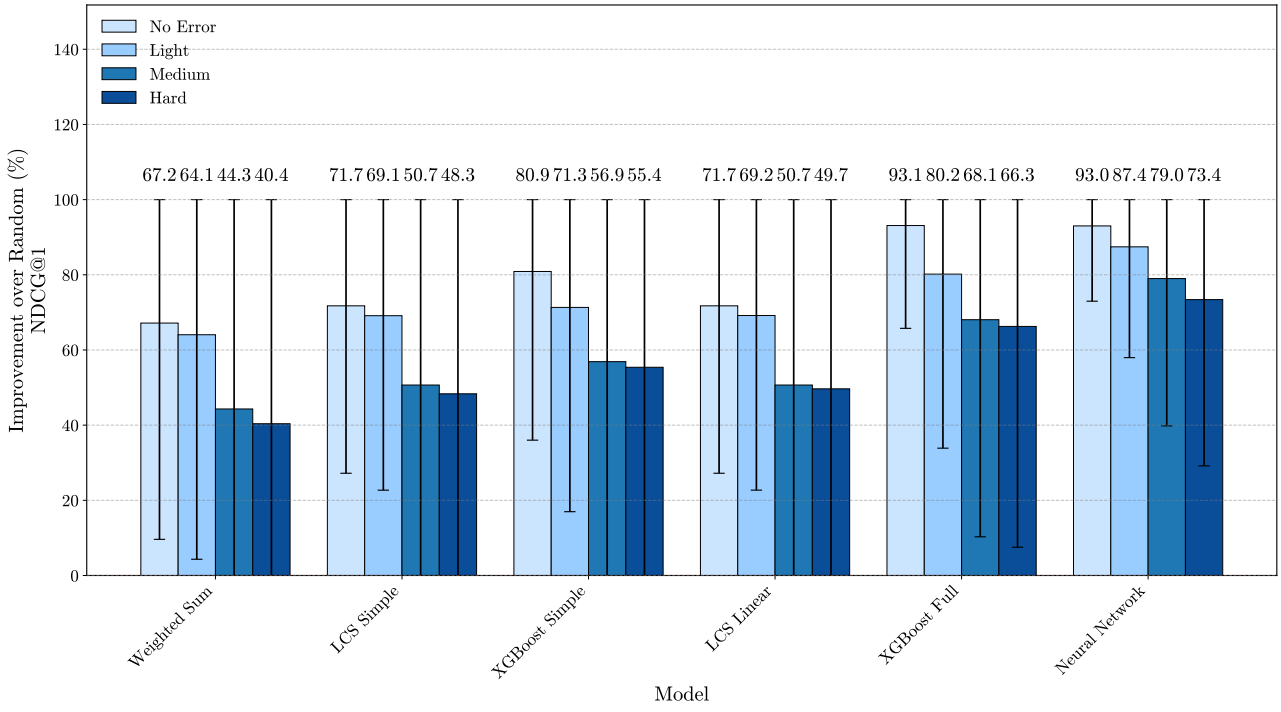

Figure 8.12: All models' NCDG@1 improvement over random across datasets

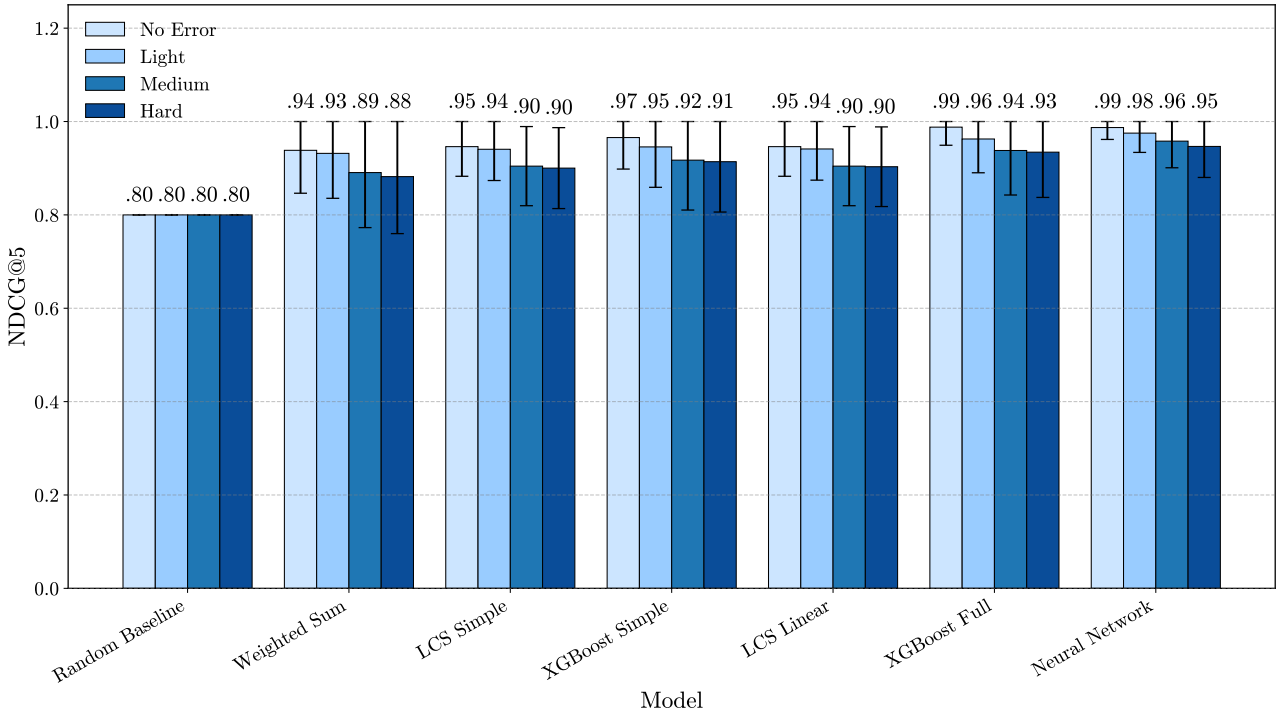

Figure 8.13: All models' NCDG@5 performances across datasets

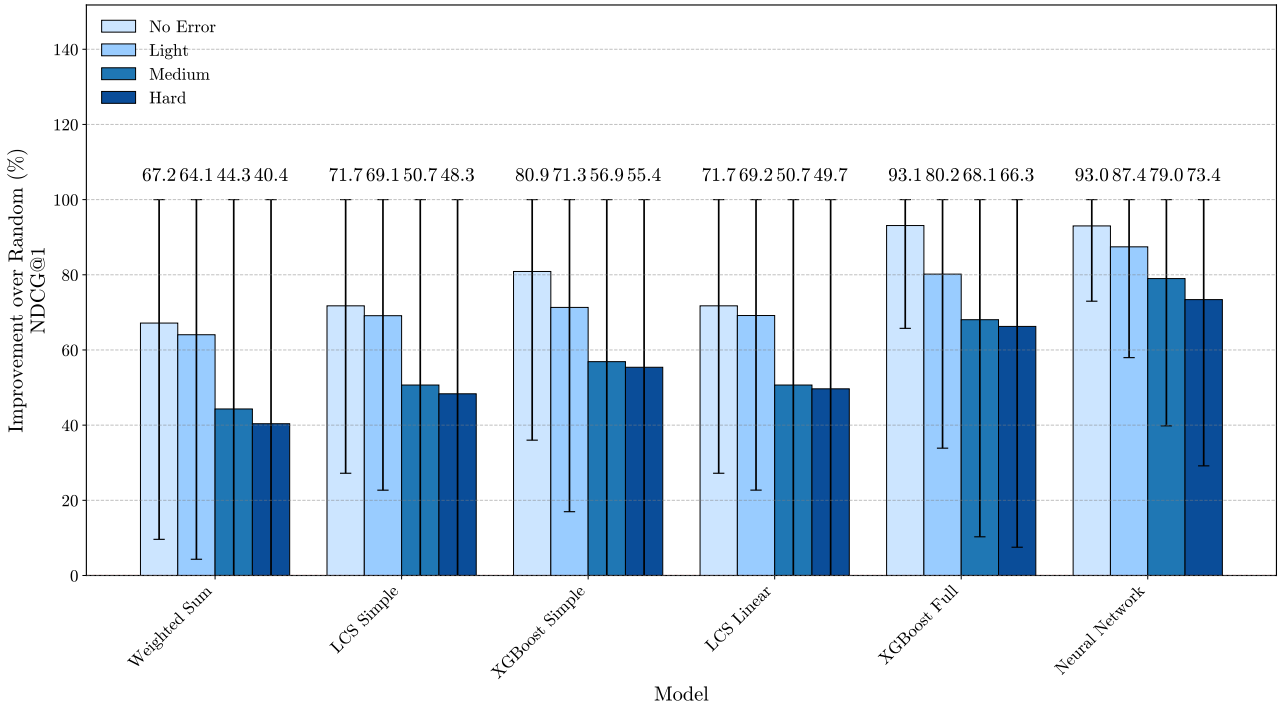

Figure 8.14: All models' NCDG@5 improvement over random across datasets

8.2.2 Feature importance analysis

As with the classification analysis, the same two types of heatmaps are shown here. A similar pattern of feature importance can be observed, but with subtle differences in others. Unlike in classification, feature importance does not seem to focus as strongly on only the top critical features. At the same time, the usual suspects retain their dominant positions, including average distance to closest airports, visibility, crosswind, and tailwind. Interestingly, methods relying on linear prediction, which include LCS and the weighted sum, tend to assign much more influence to context features compared to the neural network and XGBoost models. Of particular note is the difference between the full LCS population-based feature importance and its own SHAP-based feature importance. Again, this highlights that although methods can produce similar, or even the same, high-level signals, they can also diverge in important ways, and it is necessary to consider what these differences mean.

Following the observation that multiple explainability methods are needed, one may note that the neural network feature importance analysis also includes a permutation method rather than relying solely on SHAP and Integrated Gradients. This relates to how the ranking models processed their inputs. While classification models received one long vector with context features at the beginning, the ranking models received context features repeated before each option's features. As a result, SHAP and Integrated Gradients consistently reported the context features as overwhelmingly important, with almost no contribution from the option-specific features. This finding conflicts with the model's strong overall performance. The permutation approach avoided this bias and provided a more balanced picture. This type of problem with gradient-based methods, especially for neural networks, has been documented in prior work such as [242] and [9]. However, observing it in this context was unexpected. This further emphasizes the need for multiple explainability methods and sanity checks, demonstrating that even theoretically sound and rigorous methods such as SHAP may be misleading. A more honest assessment would acknowledge that the scenario-based nature of the dataset, and specifically how context features were structured, simply did not align well with these gradient-based methods. This raises an important question: should the neural network be excluded from consideration given the difficulties in reliably explaining it, even at the cost of its superior performance? While this decision depends on the deployment context, the permutation-based analysis seems to provide reliable explanations, suggesting that with an appropriate methodology, even complex models can be made interpretable for high-stakes applications.

In addition to the feature importance heatmaps, a heatmap showing the the simple LCS models population is shown in Figure 8.17. With a population of twenty, the weight vectors of the first and top classifiers are clearly evident, which are ordered by performance on the x-axis. The pattern is somewhat unexpected given that the usually high-ranking features are not assigned the greatest importance. Instead, runway condition and runway length dominate, with hub type also playing a surprisingly strong role. What can also be seen is the general uniformity across the population, with classifier 7 being a more notable departure. It would seem that while the global importance of features was judged to be similar, different classifiers, or small groups of them, specialized in different regions of the space. However, it must also be stated that the differences are rather subtle, which may be why it performes slightly better than the weighted sum. Finally, this figure also showcases the advantage of using simple models and a suitable and favorable problem formulation, i.e., a complex problem has been transformed into one that can be solved reasonably well using a more interpretable and human-understandable model.

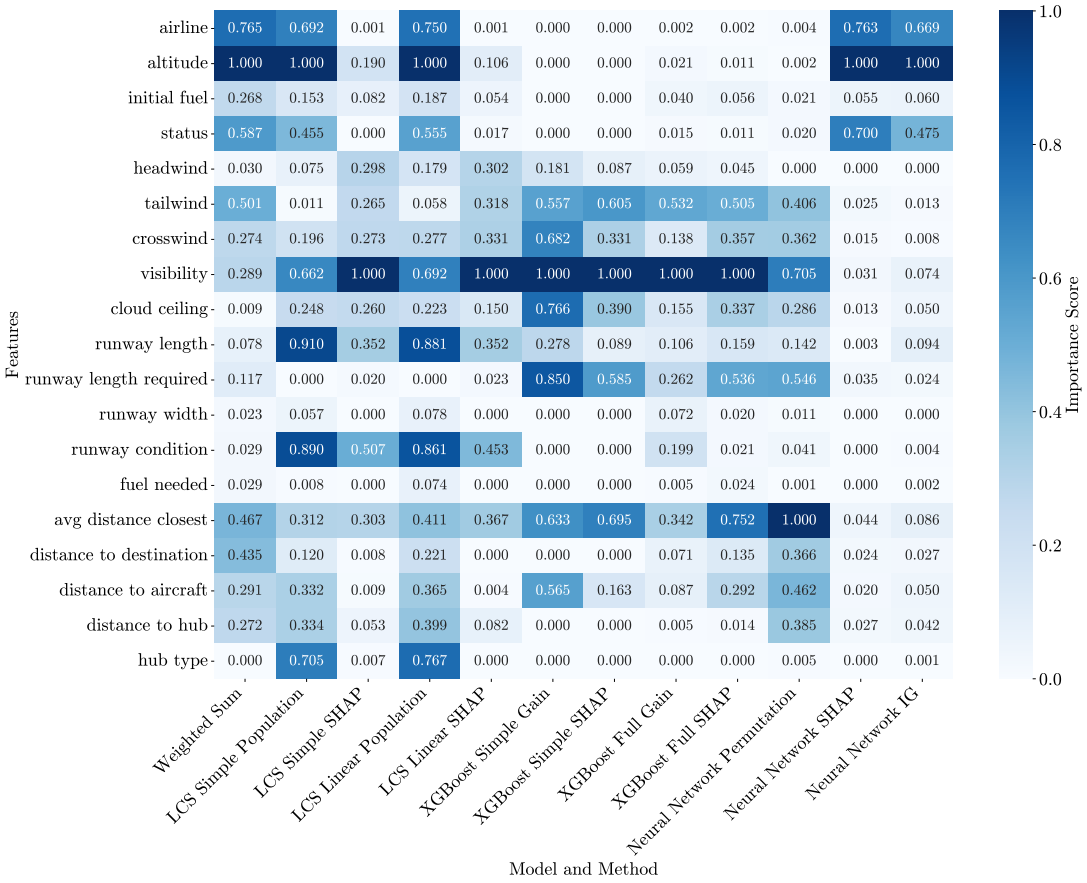

Figure 8.15: Minmax Feature importance, all models on medium dataset

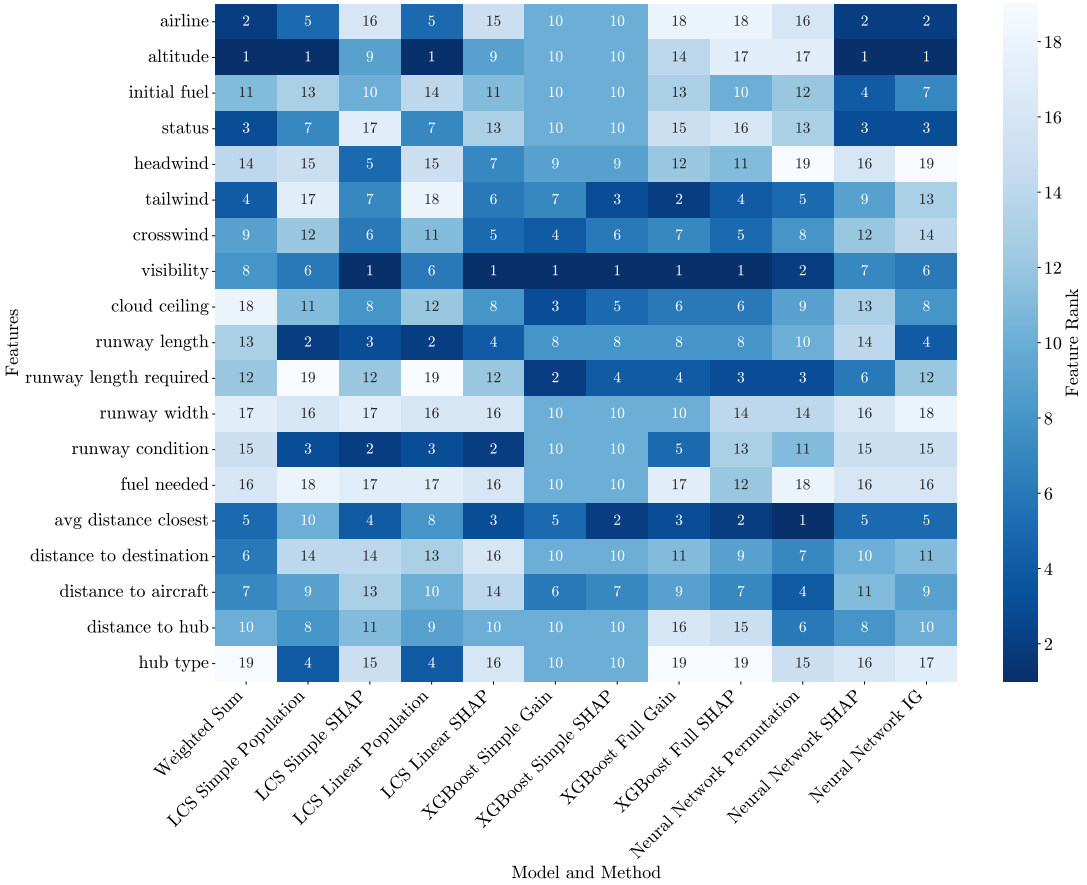

Figure 8.16: Minmax Feature importance, all models on medium dataset

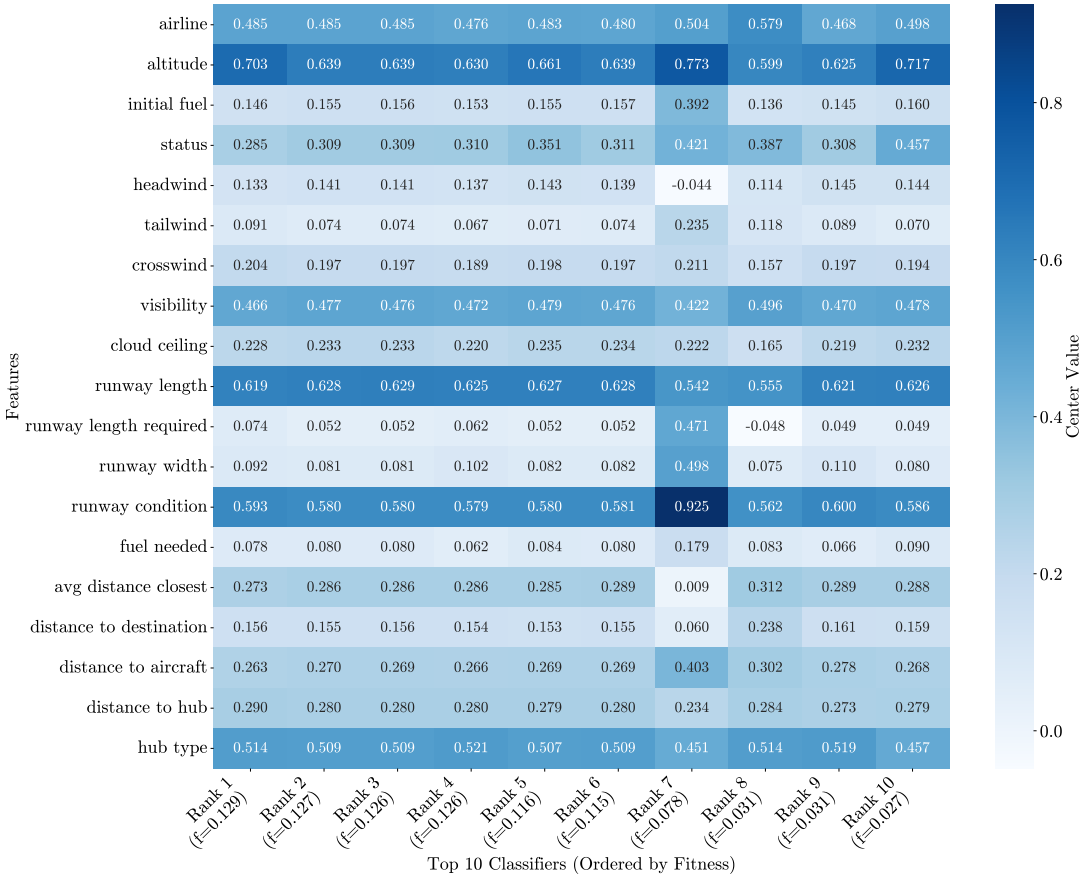

Figure 8.17: LCS simple top 10 classifier weights

8.2.3 Robustness analysis

Finally, as before, the robustness analysis examines the models' abilities to cope with variability. In NDCG@1 (Figure 8.18), the neural network achieves the highest average robustness at 0.932. LCS Linear and LCS Simple follow at 0.911 and 0.909. The weighted sum reaches 0.895, with XGBoost Simple at 0.891 and XGBoost Full at 0.888. This supports the idea that the neural network and the linear LCS variants perform best, while the tree-based approaches and the weighted sum are more sensitive. The strong result of the simple LCS again suggests that a very simple model can represent a practical contender.

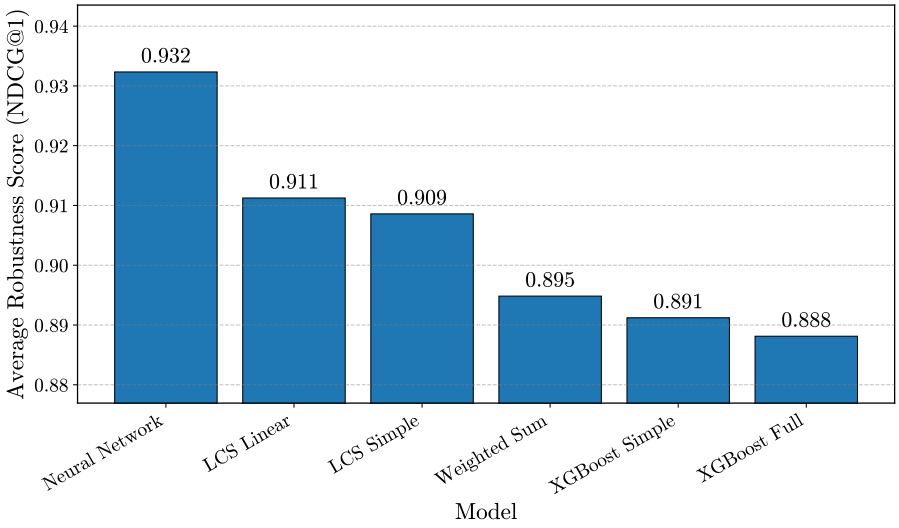

Figure 8.18: Relative robustness scores for NCDG@1 across models, higher = more robust

When examining NDCG@5 (Figure 8.19), the gaps between models narrow.

The neural network scores 0.972, LCS Linear scores 0.968, LCS Simple scores 0.967, the weighted sum scores 0.961, XGBoost Simple scores 0.959, and XGBoost Full scores 0.956. Incestigating the top five introduces a form of built-in redundancy, so even if the very top choice is affected by noise, other strong candidates still appear in the set. This mirrors how a DSS would present options to a crew and results in the ranking view being deemed a useful way to judge stability. The overall pattern remains the same, with the neural network leading and the LCS models close behind.

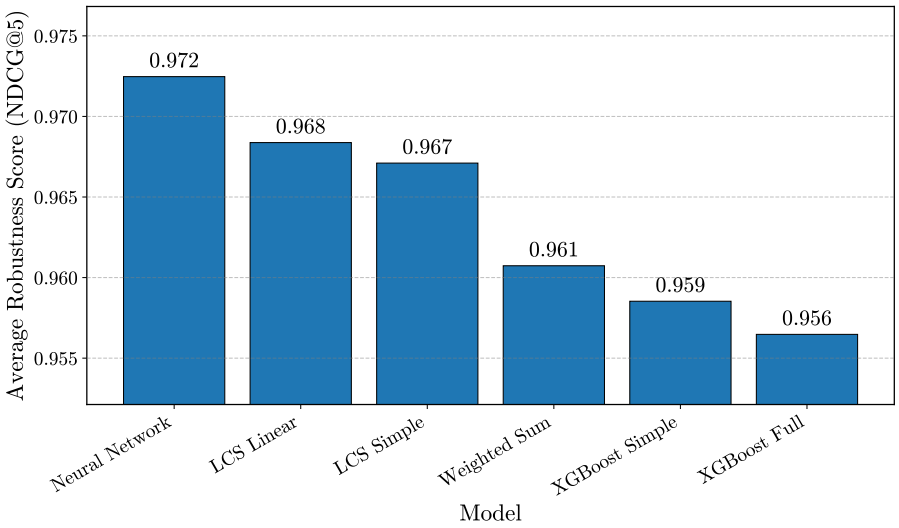

Figure 8.19: Relative robustness scores for NCDG@5 across models, higher = more robust

8.2.4 Key takeaways for ranking analysis

The ranking analysis shows that the models achieved good performance, with the NN overtaking XGBoost as the top model. Aside from simply exhibiting good performance, the reformulation of the problem allowed models like the simple LCS to progress from being unable to learn anything, showing performance similar to that of a random agent, to becoming models worthy of consideration, especially given their interpretable nature.

Particularly interesting were the results of the feature analysis, which were not drastically different from those obtained in the classification problem formulation, with the obvious exception of the NN. The NN showcased the limitations of relying on methods such as SHAP and Integrated Gradients, which required alternative approaches to produce logically consistent feature-importance insights. This was of particular interest, as it demonstrated that the model with both the highest performance and robustness still had issues with interpretability and explainability, even when using specialized methods. These are problems that have been highlighted in the literature and observed here firsthand..

8.3 Increased options analysis

The aim of this analysis is to examine how model performance changes when the number of available options increases. While the number of options ultimately presented to decision-makers was set to five, this does not preclude the system from considering a larger pool of possibilities. In fact, the ability to evaluate many options while focusing only on the top-ranked subset is a central characteristic of the ranking problem formulation and one of its key advantages. For the investigation, NDCG@1 and NDCG@5 remain the primary evaluation metrics, while the set of considered options consists of the top ten and fifteen closest airports.

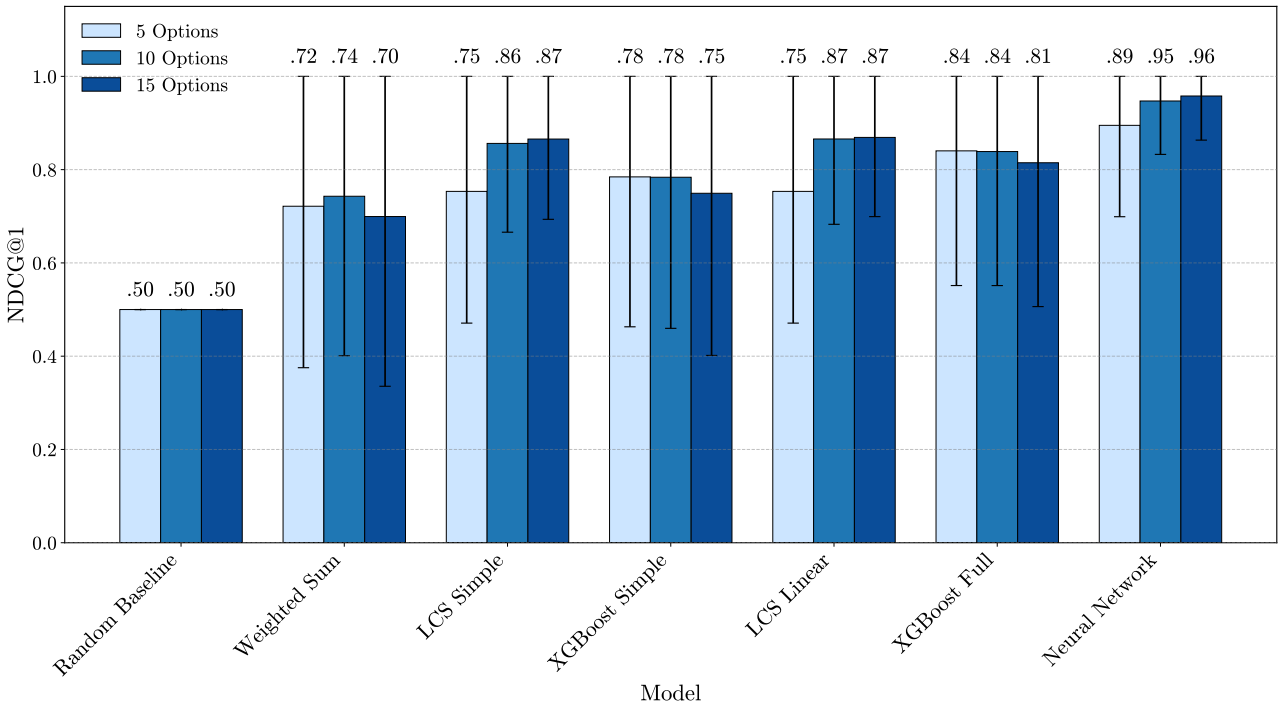

Figure 8.20: All models' NDCG@1 performances across datasets with different numbers of options

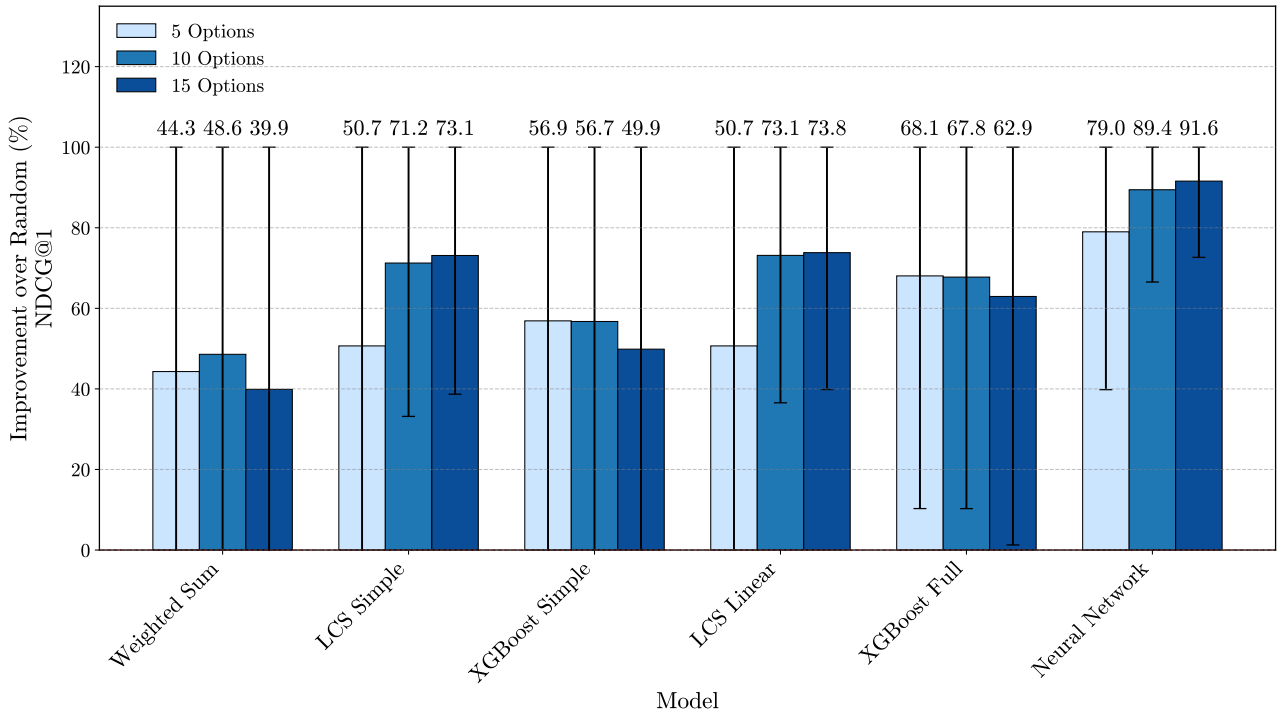

Figure 8.21: All models' improvements in NDCG@1 performances across datasets with different numbers of options

Examining Figures 8.20 and 8.21 evidences the variability in how different models are affected by an increased number of options in terms of NDCG@1 performance. Notably, both the LCS and NN models exhibit an increase in performance, whereas the remaining models show a reduction. This behavior is not unprecedented in the literature, where it has been shown that ranking models respond differently depending on the choice of NDCG@k, the list depth, and the characteristics of the candidate set [251, 54]. A plausible explanation is that for some models, access to a larger pool of options improves score separation and makes the top choice easier to identify, while others are less able

to leverage the additional comparisons and consequently experience reduced performance.

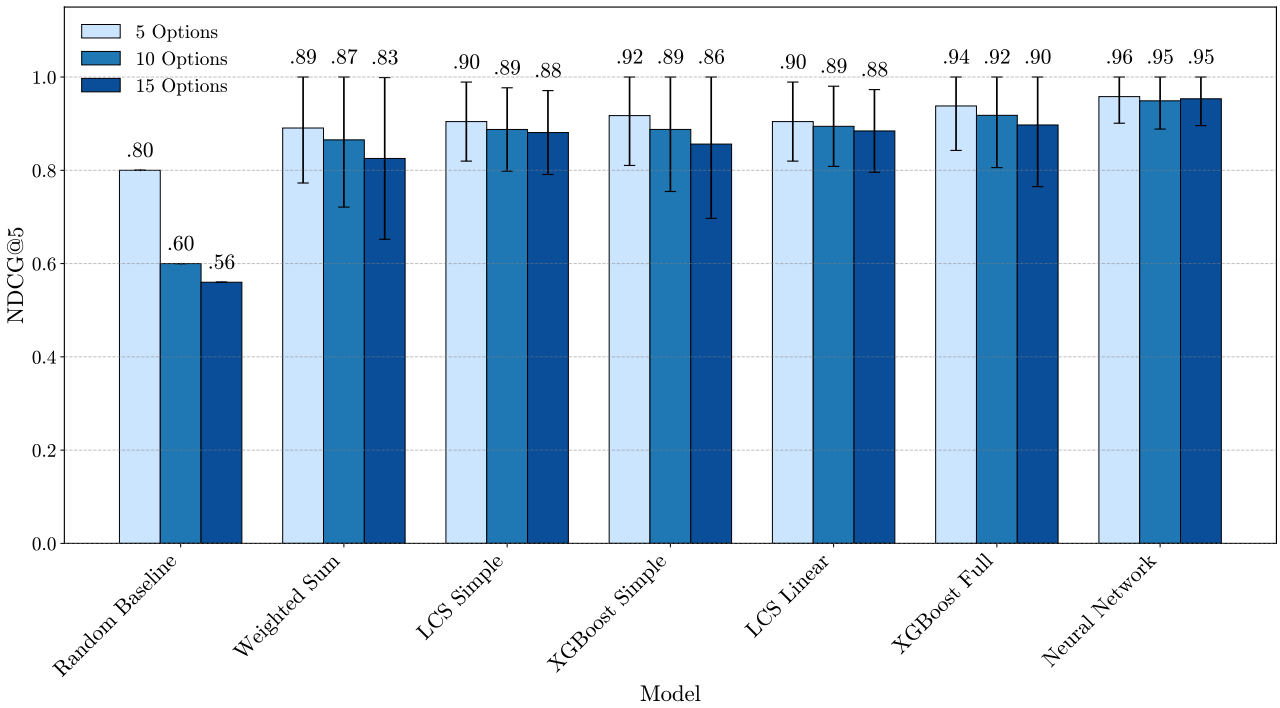

Figure 8.22: All models' NDCG@5 performances across datasets with different numbers of options

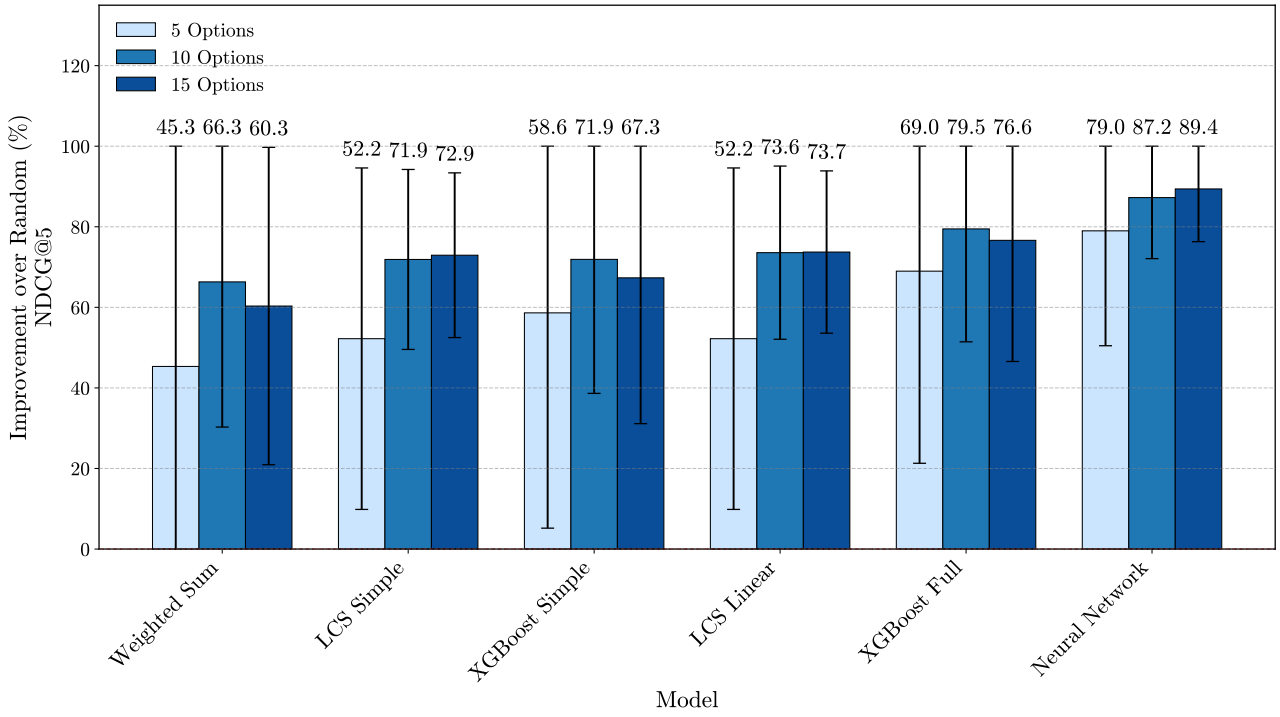

Figure 8.23: All models' improvements in NDCG@5 performances across datasets with different numbers of options

Proceeding with the analysis, Figure 8.22, which presents the NDCG@5 performance, shows a more expected pattern. As the number of options increases, the NDCG@5 score decreases for all models, with the smallest decline occurring for the NN, which almost maintains its original performance. Additionally, Figure 8.23 shows that, although NDCG@5 decreases, all models exhibit increasing improvements over the random baseline as the number of options grows. This

is a particularly interesting outcome and highlights the more malleable nature of the NDCG metric when the size of the candidate set changes.

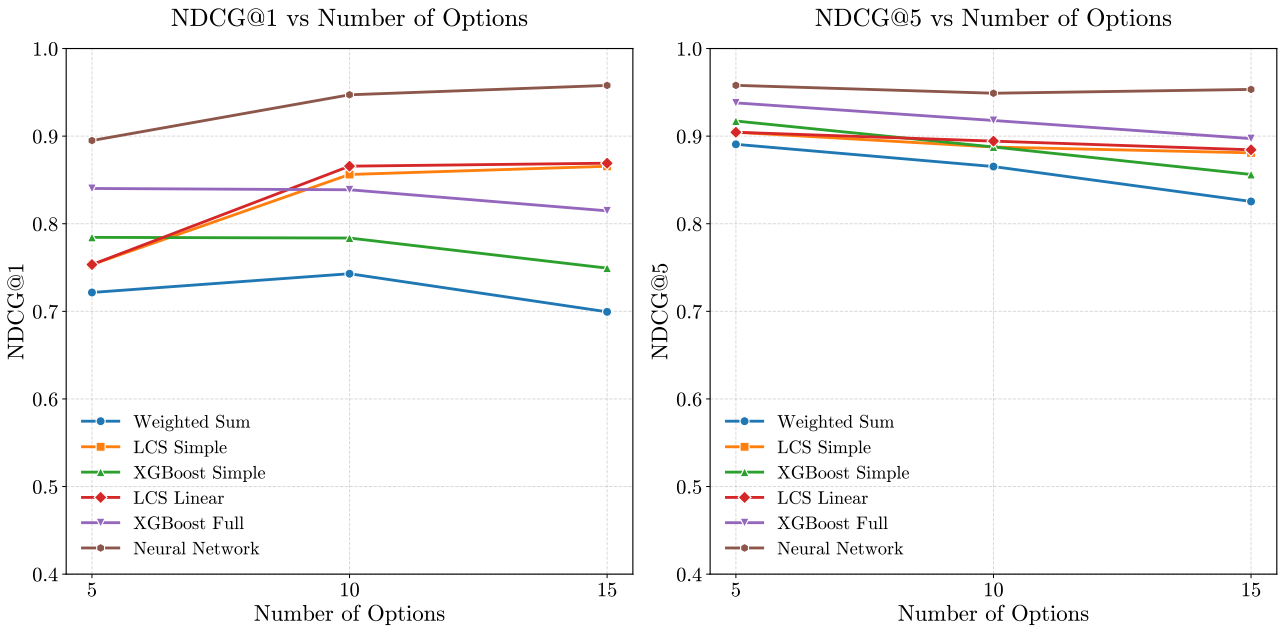

Figure 8.24: Model performance change vs. different numbers of options

Finally, Figure 8.24 shows the side-by-side changes in model performance for both NDCG@1 and NDCG@5, allowing for a more direct comparison. From this figure, the dominance of the NN becomes immediately apparent, with improvements in both NDCG@1 and NDCG@5 demonstrating that it is the most capable model when scaling to larger option sets. Particularly interesting is the improvement shown by the LCS models, especially the simple variant, which reaches a performance level comparable to that of the full model. In contrast, the XGBoost models only exhibit reductions in performance, although this behavior aligns with initial expectations.

In summary, this investigation revealed that different models behave differently when ranking options drawn from sets of varying sizes. This has two key implications. First, although a fixed number of options n may be intended for presentation to decision makers, the system can still evaluate a larger pool $m > n$, thereby increasing the number of options taken into account. Second, for some models, certain values of m may even lead to improved performance. This indicates that the choice of candidate set size is, in itself, a parameter worth further investigation.

8.4 On Model Selection Assistance

Idea

The investigation of multiple models and approaches has highlighted the real difficulty of selecting a suitable final model for deployment. The model selection process, due to the many factors that must be considered, represents an MCDM problem in its own right. However, here, this problem more closely resembles its classical form, with no time pressure and no stress.

Selecting the final model can be approached from many angles and with a wide variety of metrics. The aim of this section is to showcase an initial step that could assist decision makers confronted with this problem. The idea is to perform a type of sensitivity analysis at the start of the decision-making process. This involves three steps:

1. Analysis of areas of importance using predefined vectors of interest. This can involve the use of a vector of weights that focuses on some criteria or

vectors with different balances between the features.

2. Random sampling of many vectors and examining the distribution of these samples across the possible options—essentially, how many times each option performs the best given a large sample of random vectors.

3. Identifying which vectors from those randomly sampled produce the highest score for each of the options—basically, which combination of weights gives each option the best chance to be selected.

The goal is to obtain this information at the start so that decision makers can begin the selection process with more information and more intuition about the behavior of each of the options. The MCDM selection process can then continue by employing other methods, such as AHP, with more insight into the options.

Investigation Showcase

What follows is an exploration of how this could be applied to the current set of models. For this investigation, the performance of the models in ranking the medium dataset is considered, along with specific criteria for each model. The criteria considered are:

1. Performance ($NDCG@1$).

2. Performance ($NDCG@5$).

3. Robustness.

4. Scaling, calculated as the area under the NDCG@5 performance curve across option counts (5, 10, 15).

5. Interpretability, which can take any value from $[0, 1]$, where higher is better and reflects the model's level of interpretability.

6. Practical considerations—this criterion corresponds to practical use of the model and encompasses ease of implementation, available frameworks, ease of tuning, and breadth of support. It is limited to any value in the interval $[0, 1]$, whereby higher is better.

The formulation of the model selection process as a new MCDM problem is presented in Table 8.1. From the table, one can oberve that the initial four factors are values derived from the performances on the medium dataset. However, the interpretability and practicality aspects are subjectively formulated by the author. Regarding interpretability, models were scored highest if they were inherently interpretable, lower if they were somewhat interpretable and needed explainability techniques, and lowest if they were black boxes that required full explainability techniques. For this reason, weighted sum was assogned the highest possible score, while the neural network was allocated the lowest one, with the other models occupying the mid-range while still leaning toward either side, such as in the full XGBoost model, which is more similar to a black box given the high number of trees, yet slightly more interpretable than a neural network due to the nature of decision trees.

The practicality scores were crafted in a similar fashion. Here, the options were ranked by their ease of use, which also includes factors such as practical support in the form of easily available and maintained codebases, as well as ease of tuning and usage. In the specific case of LCS, due to its lower popularity and much higher number of hyperparameters, it received a lower score.

Finally, the table highlights that none of the considered options Pareto dominate each other, thus they are all on equal footing. Additionally, the table depicts only the raw values, which will be normalized using min–max normalization for use in the following steps.

Model	NDCG@1	NDCG@5	Rob.	Scaling	Interpretability	Practicality
Neural Network	0.89	0.96	0.95	0.99	0.0	0.2
Weighted Sum	0.72	0.89	0.93	0.97	1.0	1.0
LCS Simple	0.75	0.90	0.94	0.98	0.85	0.2
LCS Linear	0.75	0.90	0.94	0.98	0.7	0.2
XGBoost Full	0.84	0.94	0.92	0.97	0.2	0.7
XGBoost Simple	0.78	0.92	0.93	0.96	0.55	0.8

Table 8.1: Model criteria for medium datasets

Weight Focus	NDCG@1	NDCG@5	Rob.	Scale	Interp.	Pract.
Ultra Performance	0.40	0.40	0.05	0.05	0.05	0.05
NDCG@1 focused	0.70	0.10	0.05	0.05	0.05	0.05
NDCG@5 focused	0.10	0.70	0.05	0.05	0.05	0.05
Ultra Robustness	0.05	0.05	0.70	0.10	0.05	0.05
Ultra Scaling	0.05	0.05	0.10	0.70	0.05	0.05
Robustness-Scaling balanced	0.10	0.10	0.40	0.20	0.10	0.10
Interpretability	0.05	0.05	0.10	0.10	0.50	0.20
Practical	0.05	0.05	0.10	0.10	0.20	0.50
Balanced Practical/Interpretable	0.05	0.05	0.10	0.10	0.35	0.35
Perfectly Balanced	0.17	0.17	0.17	0.17	0.17	0.17
Performance-leaning	0.25	0.25	0.20	0.10	0.10	0.10
Robustness-Scaling-leaning	0.10	0.10	0.30	0.30	0.10	0.10
Practical-leaning	0.10	0.10	0.10	0.10	0.30	0.30
Pure NDCG@1	0.95	0.01	0.01	0.01	0.01	0.01
Pure NDCG@5	0.01	0.95	0.01	0.01	0.01	0.01
Pure Robustness	0.01	0.01	0.95	0.01	0.01	0.01
Pure Scaling	0.01	0.01	0.01	0.95	0.01	0.01
Pure Interpretability	0.01	0.01	0.01	0.01	0.95	0.01
Pure Practical	0.01	0.01	0.01	0.01	0.01	0.95

Table 8.2: Weight vectors used for different evaluation scenarios

A suitable starting point is an analysis of the general distribution of the models across all the sampled weights, the results of which are shown in Table 8.3. It can be observed that the neural network is deemed best in 9,000 out of 10,000 cases, accounting for 90.0% of all samples. This is a rather large chunk of the possible samples. Weighted sum follows in second place with 7.24%, Simple LCS in third place with 2.39%, and finally, XGBoost Full with 0.57%. This indicates the strong baseline dominance of the neural network, with the decrease in samples between the best and second-best options being sizable.

Model	Frequency	Percentage
Neural Network	9000	90.0%
XGBoost Full	57	0.57%
Weighted Sum	724	7.24%
LCS Simple	219	2.19%
XGBoost Simple	0	0.0
LCS Linear	0	0.00%

Table 8.3: Model preference distribution with weighted sum scoring (10,000 samples)

Shifting to a more specific view, the models selected by the predefined weights are shown in Table 8.4. A clear duopoly emerges regarding the neural network and weighted sum models. The neural network appears to dominate the performance and scaling categories, while the weighted sum model performs best

in practicality and interoperability. Of these two, the neural network is more prominent, achieving superiority in thirteen of the nineteen categories. The remaining models are mostly relegated to the runner-up column. Their behavior is consistent with expectations. The full XGBoost model performs well in performance-oriented categories, while simple XGBoost fares better in practical categories and those that balance practicality and interpretability. Finally, the LCS excels in robustness and pure interpretability.

Weight Focus	Best Model	Score	Runner-up
Ultra Performance focused	Neural Network	0.91	XGBoost Full (0.636)
NDCG@1 focused	Neural Network	0.91	XGBoost Full (0.633)
NDCG@5 focused	Neural Network	0.91	XGBoost Full (0.638)
Ultra Robustness focused	Neural Network	0.91	LCS Linear (0.556)
Ultra Scaling focused	Neural Network	0.91	LCS Simple (0.729)
Robustness-Scaling balanced	Neural Network	0.82	LCS Linear (0.529)
Interpretability focused	Weighted Sum	0.738	LCS Simple (0.621)
Practical focused	Weighted Sum	0.738	XGBoost Simple (0.559)
Balanced Practical/Interpretable	Weighted Sum	0.738	LCS Simple (0.523)
Perfectly Balanced	Neural Network	0.70	XGBoost Full (0.463)
Performance-leaning	Neural Network	0.82	XGBoost Full (0.491)
Robustness-Scaling-leaning	Neural Network	0.82	LCS Linear (0.557)
Practical-leaning	Weighted Sum	0.638	XGBoost Simple (0.493)
Pure NDCG@1	Neural Network	0.982	XGBoost Full (0.691)
Pure NDCG@5	Neural Network	0.982	XGBoost Full (0.699)
Pure Robustness	Neural Network	0.982	LCS Linear (0.575)
Pure Scaling	Neural Network	0.982	LCS Simple (0.843)
Pure Interpretability	Weighted Sum	0.964	LCS Simple (0.827)
Pure Practical	Weighted Sum	0.964	XGBoost Simple (0.774)

Table 8.4: Model performance under different weight combinations with weighted sum scoring

Concluding the analysis is an examination of the best weights found for each model, i.e., which of the sampled weights provided each of the possible models with the highest scores. Examining the results in Table 8.5 shows that the neural network is the only model to reach the highest normalized score of 1.00, with most of its gain coming from NDCG@1, NDCG@5, and robustness. The weighted sum model ranks second, with a highest observed score of 0.91, drawing most of its contribution from interpretability and practicality. Upon further and examining the other models, LCS Simple reaches a score of 0.77 with strong contributions from interpretability and scaling. LCS Linear follows with a score of 0.72, driven primarily by scaling, with noteworthy robustness. The XGBoost Full model attains a score of 0.69, where NDCG@1 and practicality account for most of its contribution. Finally, XGBoost Simple ranks last with a score of 0.65, primarily supported by practicality and NDCG@1.

Model	Score	NDCG@1	NDCG@5	Rob.	Scale	Interp.	Pract.
Neural Network	1.00	0.30	0.25	0.24	0.21	0.00	0.00
Weighted Sum	0.91	0.04	0.04	0.01	0.01	0.47	0.44
LCS Simple	0.77	0.06	0.05	0.02	0.38	0.48	0.02
LCS Linear	0.72	0.02	0.01	0.38	0.56	0.00	0.03
XGBoost Full	0.69	0.35	0.21	0.00	0.05	0.01	0.37
XGBoost Simple	0.65	0.11	0.01	0.12	0.00	0.06	0.70

Table 8.5: Best weight combinations and scores for each model using weighted sum

In summary, the analysis showcased the dominance of both the neural network and the weighted sum model. This was specifically due to their high values in a number of categories. The investigation also shows that, in the current

formulation of the problem, the more extreme models emerged as perhaps those most worthy of consideration. However, an argument can be made that this is problematic as these models had clear disadvantages in other categories, performing worst in some of them. Therefore, another suitable approach is warranted that takes this into account.

In order to address some of the difficulties identified with the typical weighted sum, a Weighted Geometric Mean approach was also employed. The weighted geometric mean is expressed as

$$S_{\mathrm{WGM}}(m) = \left(\prod_{i=1}^{n} x_i^{w_i} \right)^{\frac{1}{\sum_{i=1}^{n} w_i}}, \tag{8.1}$$

where $S_{\mathrm{WGM}}(m)$ is the weighted geometric mean score for model m, w_i represents the weights with $\sum_{i=1}^{n} w_i = 1$, x_i denotes the normalized criterion scores, and n is the number of criteria (in this case, six).

A Weighted Geometric mean is particularly useful in MCDM because it is less compensatory than the weighted sum, where poor scores can be offset by strong scores on other criteria. As a result, it promotes more balanced solutions and is more sensitive to weak performance in any single criterion.

As previously, the investigation begins with an analysis of the distribution of the best-performing choices across the randomly sampled weights, as shown in Table 8.6. It can be seen that the LCS Simple model is selected most often, with 6,978 wins out of 10,000 (69.78%). LCS Linear follows at 13.86%, the neural network at 9.69%, XGBoost Full at 4.36%, and XGBoost Simple at 2.30%. The weighted sum is ranked last, with only one win under the geometric-mean approach. This pattern reflects how the geometric mean rewards balanced profiles and penalizes very low scores on any single criterion, which favors simpler models with steady behavior across the board. Thus, we can observe the best-performing models from previous analyses being completely dethroned by the LCS variants.

Model	Frequency	Percentage
LCS Simple	6,978	69.78%
LCS Linear	1,386	13.86%
Neural Network	969	9.69%
XGBoost Full	436	4.36%
XGBoost Simple	230	2.30%
Weighted Sum	1	0.01%

Table 8.6: Model preference distribution with geometric mean scoring (10,000 samples)

The predefined weight scenarios, as shown in Table 8.7, further confirm the geometric-mean assumptions, with the best-model column now showing far greater diversity. LCS Simple is best in many practical settings and in balanced practical or interpretable configurations, and it also dominates the perfectly balanced case. LCS Linear leads whenever robustness or scaling is the primary concern, including the robustness–scaling-balanced scenario. The neural network outperforms in cases where a single performance dimension is isolated, making it the top choice for ultra performance, NDCG@1, NDCG@5, and pure robustness. For pure scaling and pure interpretability, LCS Simple emerges as the best model once again. XGBoost Simple outperforms only in the pure practicality setting. A notable absence of the weighted sum in the best-model category is evident, with its only appearance in the pure practical runner-up column.

Regarding the best weight combinations shown in Table 8.8, the neural network attains the highest score at 0.97, with most of its score stemming from NDCG@1, NDCG@5, robustness, and scaling. LCS Simple reaches its maximum at 0.70, with strong contributions from interpretability, scaling, and ro-

Weight Focus	Best Model	Score	Runner-up
Ultra Performance focused	Neural Network	0.29	XGBoost Full (0.210)
NDCG@1 focused	Neural Network	0.29	LCS Simple (0.215)
NDCG@5 focused	Neural Network	0.29	XGBoost Full (0.210)
Ultra Robustness focused	LCS Linear	0.51	LCS Simple (0.483)
Ultra Scaling focused	LCS Simple	0.65	LCS Linear (0.647)
Robustness-Scaling balanced	LCS Linear	0.45	LCS Simple (0.438)
Interpretability focused	LCS Simple	0.51	LCS Linear (0.471)
Practical focused	LCS Simple	0.33	LCS Linear (0.324)
Balanced Practical/Interpretable	LCS Simple	0.41	LCS Linear (0.390)
Perfectly Balanced	LCS Simple	0.35	LCS Linear (0.348)
Performance-leaning	LCS Simple	0.29	LCS Linear (0.290)
Robustness-Scaling-leaning	LCS Linear	0.46	LCS Simple (0.460)
Practical-leaning	LCS Simple	0.38	LCS Linear (0.358)
Pure NDCG@1	Neural Network	0.78	XGBoost Full (0.553)
Pure NDCG@5	Neural Network	0.78	XGBoost Full (0.559)
Pure Robustness	Neural Network	0.78	LCS Linear (0.566)
Pure Scaling	LCS Simple	0.82	LCS Linear (0.821)
Pure Interpretability	LCS Simple	0.81	LCS Linear (0.671)
Pure Practical	XGBoost Simple	0.61	Weighted Sum (0.610)

Table 8.7: Model performance under different weight combinations with geometric mean scoring

bustness. LCS Linear peaks at 0.69 and places most of its emphasis on scaling and robustness. XGBoost Full achieves its best score of 0.62, drawing heavily from NDCG@1 and practicality. The weighted sum follows with a highest score of 0.53, driven primarily by interpretability and practicality. XGBoost Simple attains its maximum at 0.52, largely due to practicality and some NDCG@1, although this remains well below the other models. In comparison to the rest, the weighted sum and XGBoost Simple fall noticeably short, further illustrating how the geometric mean penalizes uneven profiles with very low values on certain criteria.

Model	Score	NDCG@1	NDCG@5	Rob.	Scale	Interp.	Pract.
Neural Network	0.97	0.30	0.25	0.24	0.21	0.00	0.00
LCS Simple	0.70	0.04	0.01	0.25	0.35	0.35	0.00
LCS Linear	0.69	0.02	0.01	0.38	0.56	0.00	0.03
XGBoost Full	0.62	0.35	0.21	0.00	0.05	0.01	0.37
Weighted Sum	0.53	0.01	0.00	0.15	0.10	0.39	0.35
XGBoost Simple	0.52	0.11	0.01	0.12	0.00	0.06	0.70

Table 8.8: Best weight combinations and scores for each model using geometric mean

Overall, using the geometric mean shifts the emphasis toward models that show more balanced performances across weights. The LCS variants show greater outcomes precisely because they maintain steadier scores across criteria, while the neural network still offers the highest best case when performance dimensions dominate. This provides a complementary view to the weighted sum approach and seems to be more in line with a balanced view of the model selection problem.

8.4.1 Performance vs. Interpretability

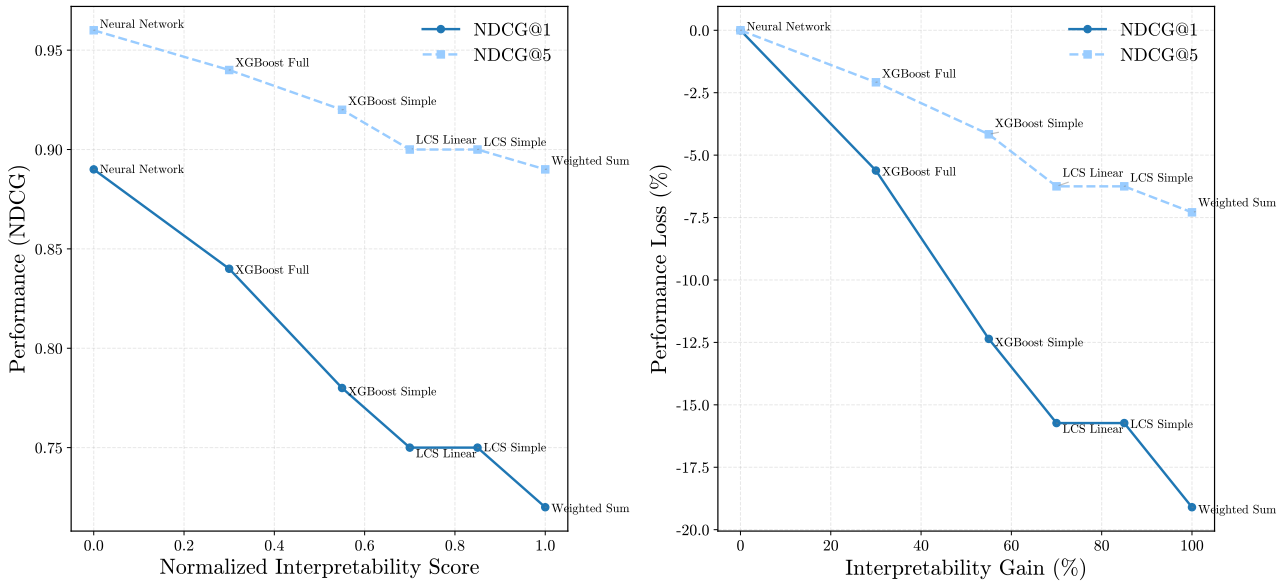

Figure 8.25: Model performance change vs. interpretability tradeoff

Given the results thus far, it seems reasonable to examine the trade-off between performance and interpretability, as the more complex models tend to achieve higher performances. Using both the NDCG@1 and NDCG@5 performance metrics, together with the interpretability scores for each model, Figure 8.25 presents two panels that illustrate this trade-off. The left panel shows how performance (both NDCG@1 and NDCG@5) varies with the normalized interpretability score, while the right panel clarifies the trade-off by depicting the percent change in performance vs. the absolute interpretability gain on a scale of 0–100 %.

Within the analysis, the neural network defines the upper bound of performance and the lower bound of interpretability, with the weighted sum representing the opposite bounds. As shown in the figure, the reduction in performance is more pronounced in the NDCG@1 metric but less sharp in the NDCG@5 metric. The decline appears less steep between the neural network and the XGBoost full model, yet when examining the raw scores, there remains a clear and consistent downward trend. The only deviation from this is seen with the two LCS models, which achieve the same performance with no loss of interpretability. The right-side panel provides a clearer view of the actual percentage of ideal possible performance each model sacrifices in exchange for interpretability.

Based on the figure, it would seem that the price for performance, especially for the more interpretable models, could be considered quite steep. However, when viewed through the intended DSS perspective, where the decision maker considers the full ranking, the NDCG@5 curve becomes more relevant. From this standpoint, the performance loss in relation to interpretability gain appears less costly. This again highlights the challenge of achieving an appropriate trade-off in such a safety-critical and essential decision-making context.

8.4.2 Key Takeaways on model selection

The final model selection for a time-critical decision-support system represents a significantly challenging task, especially in a safety-critical decision-support setting. It is an MCDM problem in which priorities matter. This investigation aimed to provide another tool in the toolbox of potential systems designers. From the current examination, it can be seen that specific arguments could be made for nearly all of the models; however, should one need to be selected, the authors' recommendation is that the simple LCS model be given strong

consideration due to its relatively good performance, excellent interoperability, and reasonably good trade-off between performance and interpretability.

8.5 Summary of Results analysis

This chapter showcased the results form the initial testing of the ExACT-MCDM methodology on the, as yet, most advanced and difficult version of the DAAS Problem. Specifically, this chapter examined how a specific selection of models behaves when applied to the problem under a straightforward classification framing, a learning-to-rank reformulation, and an increased-options setting. The models considered were: a weighted sum approach, due to its simplicity, interpretability, and widespread use in MCDM problem-solving; LCS, for its interpretable nature in tandem with its ability to capture more complex decision boundaries; XGBoost, as one of the prominent contemporary models; and, finally, neural networks, due to their performance and adaptability to approximate essentially any mapping. Additionally, the XGBoost and LCS models were represented by both a full model, which included a large number of trees and a large population size, respectively, ultimately making these models closer to the neural network, and by more interpretable versions with fewer, shallower trees and a simple LCS variant with 20 classifiers.

In classification, the full XGBoost delivered the strongest raw accuracy, followed closely the neural network, despite being more sensitive as noise increased; the weighted sum was a surprisingly strong baseline given its simplicity, while the LCS showed mixed performances, with the simple LCS model with a population of 20 classifiers failing to learn anything. The feature analysis highlighted visibility, tailwind, and crosswind, as some of the most important, with the average distance to closest airports emerging as a surprising feature of importance and seemingly a proxy for spatial redundancy and even shared weather patterns. Robustness differences appeared small at first glance but remain operationally meaningful in safety-critical contexts.

Recasting the task as a supervised ranking problem seemed to fit the decision context better. Based on the NDCG@1 and NDCG@5 performance metrics, the neural network and full XGBoost emerged, once again, as the best models, while the simple LCS benefited markedly from the reformulation and became a credible, interpretable option. The explainability study again highlighted weather and average distance to the closest as important factors, but additionally showcased the contrast between different explainability methods, with the SHAP and Integrated Gradients methods seemingly failing to provide meaningful and accurate feature-importance insight, thus necessitating the use of a permutation-based approach for the neural network feature-importance analysis. This, again, highlights the need for sanity checks and careful consideration when using these methods.

Increasing the number of candidate options from five to ten and fifteen showed a somewhat expected reduction in performance, which was more evident in NDCG@5, while the top choice even improved when evaluating on ten options. This was likely because increasing the number of options enabled the model to make more comparisons, which helped the model better differentiate the top candidates. However, this also led to an overall decrease in the ranking of the top five options, as it became more difficult to obtain the correct full. These promising results indicate the potential for more options to be taken into account than simply the number displayed to the final decision-makers.

The examination concluded with a section presenting an a priori sensitivity-analysis approach and showcase aimed at understanding which of the models in the examination might be chosen and providing greater insight into the performance–interpretability trade-off between the possible options. The neural network exhibited the highest performance but lowest accuracy, and we recommend that the simple LCS consisting of 20 classifiers be given due consideration

due to its reasonable performance and high interpretability.

9 Conclusion and Future Work

This chapter summarizes the main contributions and lessons learned from the work presented in this thesis. It begins by revisiting the goals that were set out at the start of the research and clarifying the contributions made in response to them. The chapter then discusses the research questions posed in the introduction and draws together the insights gained across the different investigations. The chapter concludes by outlining directions for future work, regarding both practical integration into the developing IPAS system and the extension of the methodological ideas developed within this work.

9.1 Goals and contributions

G1: Develop a methodology for solving atypical and time-critical multi-criteria decision-making problems using artificial intelligence to complement and extend traditional MCDM approaches

This goal was achieved through the creation of the ExACT-MCDM methodology. The methodology was designed to address the scarcity of data relating to emergency and atypical situations and provide models that can support decision-making in these situations, where classical approaches are too slow or too demanding for the decision-makers involved. The main approach is to reformulate emergency MCDM as a supervised learning task. Within it, each scenario would be treated as its own decision problem with options and factors. A structured evaluation function encodes expert knowledge, enabling the incorporation of constraints for specific and/or all factors, which ultimately provides scores for each option in the scenario. This, in combination with a synthetic data generation process, allows for large amounts of labeled data to be created without requiring experts to evaluate each case one by one.

The methodology also incorporates robustness by varying scenario factors to simulate realistic uncertainty. The robustness emphasizes proper variability of the factors, rather than random perturbations, while simulating reasonable variability and incorporating subsequent changes in other factors in order to preserve their interdependency. This, in tandem with ideas adapted from stochastic dominance, would then be used to provide robust scores for each option in the scenario. The resulting dataset would reflect the type of inputs a deployed system would encounter in practice. Models trained on this dataset would then act as surrogates, replacing many simulations and variations while being able to provide a recommendation almost instantaneously when needed. This aspect is particularly impor-

tant in real-world emergencies in which the available decision window is often extremely narrow. A striking example is the "Miracle on the Hudson" in 2009, where the pilots had less than one minute to decide on a course of action after both engines failed. In such cases, it is not sufficient for a system to eventually provide the right answer: the recommendation must be available immediately so that decision makers can register the situation, weigh their options, and execute the chosen action in time. By acting as a surrogate to heavy simulations and delivering near-instant suggestions, the ExACT-MCDM methodology directly addresses this requirement and ensures that the system can be a meaningful aid when time is the most critical resource.

These models preserve the MCDM structure of options and attributes while learning from generated and augmented data. In this way, ExACT-MCDM complements classical MCDM by maintaining its structure but extending it with AI models that generalize, adapt, and provide expedient assistance in emergency time-critical decision-making situations.

G2: Demonstrate and evaluate this approach through the dynamic alternate-airport selection problem contributing practical insights to the further development of the IPAS system

The methodology was applied and tested on the dynamic alternate-airport selection problem. This case was chosen because it represents a realistic and demanding emergency decision in aviation where multiple criteria must be balanced under changing conditions. Following the methodology, a scenario generation process produced a wide range of cases, including events that closely resemble those pilots might encounter, as well as more extreme situations that push the boundaries of operational feasibility.

It is important to highlight that the methodology was applied in a limited capacity, as the author represented a unified committee of experts tasked with the development of the models and carried out only a single cycle in what would otherwise be a more cyclical and multi-stage approach.

The resulting trained models demonstrated strong performance, particularly when the problem was reformulated as a supervised ranking task rather than a classification task. The ranking approach enabled the use of simpler models because it allowed the dataset to be represented in a tabular form rather than being rolled out into a vector input as required for classification. This also facilitated better scaling and made it possible to apply ranking metrics, such as NDCG, which provide more meaningful insights into the recommended options and their relationships with one another.

Importantly, the ranking perspective also aligns better with the actual decision-making context of alternate-airport selection. In practice, a pilot is not asked to accept or reject a single option but must compare and prioritize multiple feasible alternatives. Providing an ordered list of recommendations therefore reflects the way such decisions are made in reality and increases the practical relevance of the models.

Finally, this alignment also supports future integration into systems such as the IPAS framework under development at DLR. Since IPAS is designed to present multiple airport options to the crew, a ranking-based model output can be incorporated more naturally into the interface, ensuring that the recommendations fit seamlessly into the workflow of a decision-support system and provide pilots with actionable, prioritized alternatives in critical situations.

9.1.1 Contributions

The biggest contribution of this thesis is the development of the ExACT-MCDM methodology, which demonstrates how a time-critical MCDM problem

can be transformed into a supervised ranking task. This allows the training of models that can provide meaningful insights precisely when decision-makers need them most. Although the methodology was inspired by and tested on a practical aviation problem, it has the potential to be applied to a wide range of other domains that share similar requirements and constraints.

The second contribution lies in advancing the understanding of the DAAS problem itself. At the outset, this thesis defined DAAS as a multi-criteria decision-making problem and articulated its additional requirements and constraints. Building on this foundation, the thesis then applied the newly developed methodology to the most advanced and challenging version of the problem, yielding meaningful results and deeper insights into its structure and solution space.

Of note is the creation of the Multi-objective Multiplexer Decision-Making Benchmark Problem, which represents a scalable benchmark inspired by DAAS and does not require exact aviation data to construct. Instead, it is grounded in well-known problems from the field of multi-objective optimization, which allows researchers interested in these kinds of questions to experiment with different approaches without being tied to the aviation domain.

Finally, this thesis contributes to the development of a concrete component of an actual decision-support system currently under development. By demonstrating how trained models can be integrated into such a system, the work strengthens the link between theoretical research and operational practice, with the possibility that elements of this approach may one day find their way into the cockpit of future aircraft.

9.2 Conclusions

RQ1: In what ways can artificial intelligence techniques be integrated with MCDM concepts to improve decision support in emergency, time-critical situations where the main decision-makers and responsibility holders are embedded in the emergency scenario itself?

Artificial intelligence, and more specifically machine learning models embedded within MCDM frameworks, can be integrated to improve decision support in several important ways, and the work of this thesis shows how this can be achieved in practice. The literature indicates that such integration is often motivated by the need to reduce the time and cognitive effort required for complex multi-criteria evaluations since MCDM methods are resource intensive, not only in terms of computational effort but also in terms of the mental workload placed on decision-makers. By learning from previous decisions and patterns, AI can automate parts of the evaluation process and make MCDM more efficient and usable in settings where time and clarity are limited.

In emergency and time-critical decision-making, this integration becomes even more valuable since, under stress and strict time constraints machine learning models are able to provide rapid recommendations while still operating within the structure of MCDM. The central challenge is therefore not whether AI can be used at all but how it can be integrated in such a way that it meaningfully supports human decision-makers remain responsible for the outcomes and who must retain oversight of the process. The literature has explored different paths to integration, including the use of machine learning to estimate factor weights or treating the task as a regression problem where each option is scored. Building on these ideas, this thesis introduced the ExACT-MCDM methodology, which reformulates emergency decision-making as a ranking problem with elements drawn from contextual bandits. In this design, machine learning is trained in a supervised fashion to produce rankings of the available options while preserving the MCDM structure of options and attributes. The advantage

of this integration is that the model is able to adapt its rankings to the specific context of a new emergency scenario rather than simply replicating choices from the past, and this adaptability is especially important when even small contextual changes can alter what the best option should be.

Both classification and ranking approaches were tested in the experiments. The classification setup required the model to identify the single best option, while the ranking formulation asked the model to order all available options by suitability. The results showed that while classification could provide useful insights, it struggled in cases where multiple options were close in quality, leading to less stability in the predictions. The ranking approach, by contrast, proved to be more natural for MCDM since it preserved the relative comparisons between options and allowed the final choice to remain with the human decision-maker. This resulted in not only a better technical fit but also a better conceptual match for emergency scenarios where flexibility and oversight are crucial.

The results showed that many of the tested models were indeed able to perform well, particularly in the ranking tasks where NDCG@5 scores reached values as high as 96% for neural networks with XGBoost following closely behind. These findings established a clear point of reference for the problem by testing a variety of models in a systematic way, which had not been achieved before. However, the question of whether such results are fully sufficient for deployment remains open since this thesis does not propose a fixed threshold for what should be considered acceptable performance. Instead, this decision is left to the experts who would design and implement such a system, relying on their judgment and the specific requirements of the operational environment.

These observations suggest that while the models can provide strong performance and valuable insights, it does not seem advisable at this stage to treat them as fully independent decision-making agents. Their role is better understood as decision-support tools that can process large amounts of information, highlight the most promising options, and present them in a structured form, while leaving the responsibility of the final choice with the human experts who have oversight of the situation. In this way, the integration of AI with MCDM remains aligned with its intended purpose of supporting, rather than replacing, the decision-makers embedded within emergency scenarios.

RQ2: Due to the diverse nature of emergencies, how can the AI model be trained in such a way that it may be able to provide decision support adapted to the situation at hand?

Emergencies often fall outside of normal operations and represent unusual events,representing diverse scenarios not only when compared to standard situations but also when compared to one another. In decision-making, this carries significant weight since choice that is not possible in one set of circumstances may be the right choice in another, which means that the training of the model cannot be limited to predicting outcomes alone but must also take into account the broader context in which the decision is made.

In this thesis, this challenge was addressed by borrowing ideas from reinforcement learning (more specifically, contextual bandits) and the ExACT-MCDM methodology integrated contextual features directly into the decision scenarios. This made it possible for the model to not only learn the link between options and outcomes but also to capture how this link changes when the situation changes. The process was straightforward in its design. First, the relevant context features were identified, including factors such as airline, technical status, initial fuel, and altitude, and these were then incorporated in two different ways—either kept constant across all options in the ranking tasks or used as constant starting features in the

rolled-out vector in the classification check.

The feature importance results showed that the models made active use of these context features, and while they were used in different ways across the experiments, greater focus was consistently placed on those features that had more weight in the labeling process. For instance, visibility and tailwind had much more impact on the outcome than airline, which corresponds with what would be expected in an actual emergency and demonstrates that the models did not simply copy patterns but rather adapted their rankings depending on the specific circumstances.

There are also two broader design choices when it comes to how context can be integrated. One approach, which was followed here, treats the context as separate and constant across the entire scenario, ensuring that it frames the decision space without being entangled in the option features. The other possible approach is to embed the context directly within the options themselves so that it becomes part of the option factors, which may provide greater flexibility but can also blur the distinction between context and choice. Both approaches seem viable, and the decision on which to use should be guided by the experts involved and by what is most suitable for the specific problem being addressed.

RQ3: What data requirements, limitations, and design considerations need to be addressed to develop and train an AI model that can reliably support time-critical, high-stakes MCDM scenarios?

The data requirements for combining ML and MCDM constitute a considerable challenge, and this is especially true in emergencies that are, by nature, outliers. To address this problem, this thesis explored a generated scenario-based approach, a method often used when data acquisition is limited and difficult, and in this way, it became possible to create a suitable dataset even for rare and extreme situations. However, the most significant factor is not the quantity of generated data but whether it is a realistic representation of the type of inputs the deployed model would face, given that even the largest dataset is of little use if the trained model fails when confronted with the real conditions of deployment.

This approach—generating scenarios and simulating variations—also introduces its own limitations and design considerations. Each option within a scenario must be labeled, which requires significant computational effort and time. In effect, the method exchanges the cost of acquiring data with the cost of generating it, and this cost can be substantial if the underlying simulations are too detailed. At the same time, the advantage is that the labeling burden is shifted away from human experts and placed onto computational resources, which can be scaled and parallelized with more ease. In this way, the methodology prioritizes expert input to define and guide the scenarios, while the heavy calculation and labeling is handled by computers.

A further consideration lies in feature engineering. In typical ML workflows, feature reduction or transformation can improve model performance, but in this context, such methods must be treated with caution. Because interpretability and explainability are essential, creating new aggregated features through dimensionality reduction techniques such as PCA is not advisable unless decision makers can clearly understand what these features represent. The design process must therefore ensure that feature choices do not undermine clarity or trust, even if this comes at some cost in predictive accuracy.

The feature analysis carried out in this thesis also produced important insight. The average distance to airports, used as a contingency factor, was heavily relied upon by all models and resulted in containing more information than first expected. It not only reflected the physical position of the aircraft relative to nearby airports but also indirectly captured geography

and even weather conditions, since the dataset generation process linked these variables together. In practice, the feature acted as a hidden combination of several influences rather than a single straightforward measure, which greatly increased its predictive strength but also raised questions about interpretability.

This leads to an important design consideration. The goal of supporting time-critical and high-stakes decisions requires not only high performance but also trust and clarity. A composite feature may improve accuracy but can also obscure the reasoning process, making it harder for decision makers to understand why a particular option is recommended. If a model places heavy weight on such a feature, it may be unclear whether the choice was driven by a single dominant element, an implicit combination of factors, or variables entirely hidden within the aggregated data.

Composite features therefore present both opportunities and risks. They can provide a strong signal that improves performance, but they can also reduce transparency and make it difficult to explain the decision to a human in a high-stakes situation. The balance between accuracy and interpretability must therefore be considered directly when selecting features and generating scenarios, and this design choice is as important as the choice of model itself.

In conclusion, the data requirements, limitations, and design considerations are not limited to the size or scope of the dataset but extend to how the data is generated, how it is labeled, and how features are represented. The experts involved in designing and creating the scenarios must therefore consider not only which factors to include but also how these factors influence both interpretability and explainability, given that reliability in high-stakes and time-critical MCDM requires more than raw model performance.

RQ4: How can interpretability and explainability be integrated so that stakeholders can maintain clear oversight of the decision-making process?

Interpretability and explainability can be integrated from multiple angles, and the work in this thesis shows several ways in which this can be achieved. The methodological design already placed strong emphasis on transparency, with the expert labeling function serving as a central mechanism for ensuring domain knowledge is explicit and traceable. By requiring experts to encode their reasoning through structured guidelines rather than only through simplified inputs, such as pairwise rankings or direct weights, the process made it possible to later examine and understand how knowledge was embedded, which provides clearer oversight for all stakeholders.

This integration was also reflected in the choice and comparison of models. While high-performing models, such as neural networks and XGBoost, proved effective in ranking and classification, the reformulation of the problem into a ranking task allowed simpler models, such as a small LCS population to achieve competitive performance. This demonstrated that careful problem formulation introduces the possibility of using models that are not only accurate but also transparent and easier to validate, an important requirement when the system must eventually be deployed in a safety-critical environment and certified for use.

A further layer of integration was achieved through the use of model-agnostic explainability tools, particularly SHAP. SHAP offers both global insights into feature importance and more detailed local explanations that show how individual factors influenced a specific decision. This dual perspective makes it valuable for developers who need to understand the inner workings of the system and for end users who require clarity in high-stakes contexts. At the same time, it also illustrates the trade-offs that must be

considered since SHAP requires substantial sampling and heavy computation, and while optimized implementations exist for certain models, such as tree-based methods, others like LCS do not yet benefit from such tailored support.

The point made in this thesis is that interpretability and explainability are best achieved not by relying on a single measure, but by weaving them into the process from the start—beginning with transparent labeling and scenario generation, continuing through careful model choice where simpler options can occasionally be more suitable, and reinforcing them via model-agnostic tools when these can be applied. In this way, stakeholders are provided insight into how the system functions and why particular decisions are suggested, which is essential if such a system is to be trusted and relied upon in time-critical and high-stakes environments. The lesson from this work is that interpretability and explainability must be treated as a guiding principles in system design, influencing decisions from data preparation to model selection and evaluation, if such systems are to gain acceptance and real practical value.

9.3 Future Work

Although the work in this thesis represents a suitable exploration of the topic, in which many meaningful answers were derived and the main goals achieved, the sheer breadth of the subject and the possible investigations are greater than the scope of the thesis. It is the author's belief that this also signifies the importance of the topic investigated, where new questions and considerations naturally emerge as efforts are made to clarify the initial questions and address the initial goals. This is especially true given the development of the ExACT-MCDM methodology, the focus on scenario generation, and the reformulation of the problem as ranking, which, together, provided answers to certain questions but also opened the door to many others.

First, it must be acknowledged that the methodology testing carried out in this work was only conducted using a single cycle, with the author acting as the unified panel of experts. This means that although the methodology shows promise, it still needs to be tested further with a proper assembly of experts and studied in a more appropriate environment. Future work should not only focus on involving a larger number of experts but also exploring how to manage disagreement, uncertainty, and bias between them. This would allow for the possibility to test how the system functions when real-world diversity of opinion is introduced, and how such diversity should be represented in the model. In addition, in the case presented here, many of the constraints were represented as piecewise linear transformation functions, which, although suitable for the current problem, do not provide the full picture of the methodology's capabilities. Therefore, more experimentation and a more detailed investigation should be carried out to examine how the integration of more complex and non-linear transformation functions would change the outputs and influence model performance—for instance, in cases where constraints may have thresholds, cascading effects, or non-linear safety margins.

Second, the methods need to be expanded to include additional components, two of which have already been identified as important. The first is the development of mechanisms to help interested parties, especially the system's creators, determine what level of model performance should be considered suitable or acceptable and provide a structured manner of selecting the model that will be used. This would replace the current reliance on simple expert consensus with a more transparent and rigorous process. Such mechanisms also connect directly to the aviation field's requirements of certification and thorough testing, and outside aviation, they would represent a prudent step for any safety-critical system. The second component concerns visualization and analysis tools. Within

this thesis, much effort was paid to incorporating simpler and more explainable models, as well as exploring model-agnostic methods from the field of XAI. However,the field is missing suitable and well-developed techniques that would allow effective visual analysis so that decision-makers can spend less time struggling to interpret outputs and more time considering their suitability and refining how domain knowledge is encoded. The goal is to incorporate streamlined visualization and analytical tools that reduce friction in the overall process and allow system developers to focus on what truly matters. This is not only a matter of convenience but also of trust and oversight, since decision-makers in safety-critical systems must be able to understand and justify the recommendations they are given. Current visualization methods, such as heatmaps and bar plots, are more appropriate for experimental evaluation, and the helper tools used in their present form are insufficient to serve as part of the methodology itself.

Third, given that this thesis has a practical component—addressing an actual problem from the field of aviation—and is concerned with the development of models for DSS, it stands to reason that the next logical step would be to test how such a model can be integrated into a DSS system and evaluated with human decision-makers. Therefore, future plans involve the trained models being integrated into the IPAS system currently being developed at DLR and used to conduct tests with pilots. The idea is to combine the AI core module, which contains the trained model, with an appropriately designed AI–crew interaction system (shown in Figure 9.2), and to use the DLR's iSIM simulator (shown in Figure 9.1) as a test bed for the newly developed system. In this way, the work presented here moves beyond theoretical and exploratory investigations and is extended into real-world application, with the goal of eventually being implemented in the cockpit of an operational aircraft.

Figure 9.1: DLR in-house custom cockpit simulator iSIM

Going one step further, there are ideas regarding how this aspect of the decision-support system could be augmented to not only consider the possible airports that could be reached but also generate a dynamic list of all potential surfaces that might serve as landing platforms, including highways, clearings, and other suitable areas. Such an approach would require the system to dynamically generate a set of feasible landing options with variable lengths and account for additional factors beyond those relevant to airports. At the same time, it would provide pilots with valuable alternatives, particularly in situations in which fuel is critically low or when all airports are out of reach.

Beyond aviation, it should also be recognized that the methodology itself is

Figure 9.2: AICIS interface integrated into the iSIM showing alternate-airport list, taken from authors publication [81]

not limited to one field alone. The combination of scenario generation, ranking reformulation, and contextual modeling could be applied to other emergency domains, such as disaster response, medical triage, or energy grid management. Each of these domains raises its own challenges in terms of data, context, and expert knowledge, but they share the same need for rapid, interpretable, and trustworthy decision support. Exploring these directions would not only test the generality of the methodology but also help establish whether the balance between accuracy, interpretability, and adaptability achieved here can form a broader standard for how AI is integrated into high-stakes decision-making.

In summary, the thesis has shown that emergency MCDM problems can be reformulated as supervised ranking tasks, enabling AI models to provide interpretable, context-sensitive decision support under time pressure. The hope is that this work not only contributes to the development of IPAS but also lays the groundwork for the broader integration of AI into high-stakes decision-making domains.

Bibliography

[1] User manual for the base of aircraft data (bada) revision 4.2. Tech. Rep. EEC Technical/Scientific Report, EUROCONTROL Experimental Centre, Brétigny-sur-Orge, France (2021), https://www.eurocontrol.int/publication/bada-aircraft-performance-model, provides standardized aircraft performance and fuel consumption models based on flight manuals and operational data

[2] Nasa merra-2 reanalysis dataset (modern-era retrospective analysis for research and applications, version 2). NASA GMAO website (2025), https://gmao.gsfc.nasa.gov/reanalysis/MERRA-2/, global atmospheric reanalysis providing wind, temperature, pressure, and humidity fields from 1980 to present

[3] Eurocontrol demand data repository (ddr). EUROCONTROL website (nd), https://www.eurocontrol.int/ddr, service providing access to past filed flight plan trajectories and actual trajectories across European airspace since August 2012. Accessed: 2025-08-27

[4] Eurocontrol flight plan filing and management (ifps/cfmu). EUROCONTROL website (nd), https://www.eurocontrol.int/service/flight-plan-filing-and-management, service describing the centralized collection, validation, and distribution of filed flight plans via IFPS—Integrated Initial Flight Plan Processing System. Accessed: 2025-08-27

[5] Abdel-Basset, M., Mohamed, R., Elhoseny, M., Chang, V.: Evaluation framework for smart disaster response systems in uncertainty environment. Mechanical Systems and Signal Processing **145**, 106941 (2020). https://doi.org/10.1016/j.ymssp.2020.106941

[6] Abdulkareem, K.H., Al-Mhiqani, M.N., Dinar, A.M., Mohammed, M.A., Al-Imari, M.J., Al-Waisy, A.S., Alghawli, A.S., Al-Qaness, M.A.A.: Mef: Multidimensional examination framework for prioritization of covid-19 severe patients and promote precision medicine based on hybrid multi-criteria decision-making approaches. Bioengineering **9**(9), 457 (2022). https://doi.org/10.3390/bioengineering9090457

[7] Abdulla, A., Baryannis, G.: A hybrid multi-criteria decision-making and machine learning approach for explainable supplier selection. Supply Chain Analytics **7**, 100074 (2024)

[8] Adebayo, J., Gilmer, J., Muelly, M., Goodfellow, I., Hardt, M., Kim, B.: Sanity checks for saliency maps. Advances in neural information processing systems **31** (2018)

[9] Adebayo, J., Gilmer, J., Muelly, M., Goodfellow, I., Hardt, M., Kim, B.: Sanity checks for saliency maps. Advances in neural information processing systems **31** (2018)

[10] Administration, F.A.: Pilot's handbook of aeronautical knowledge. Skyhorse Publishing Inc. (2009)

[11] Agrawal, S., Goyal, N.: Thompson sampling for contextual bandits with linear payoffs. In: International conference on machine learning. pp. 127–135 (2013)

[12] Ahmad, S., Bibi, B., Jun, W., Ali, R.: A novel multi-criterion decision-making strategy using topsis under non-linear diophantine fuzzy numbers for the covid-19 problem. Soft Computing pp. 1–20 (2023). https://doi.org/10.1007/s00500-023-09273-8

[13] Airbus: A318/A319/A320/A321 Flight Crew Operating Manual (2013), issue Date: 11 Dec 2013, Reference: LFO A318/A319/A320/A321 Fleet FCOM

[14] Akiba, T., Sano, S., Yanase, T., Ohta, T., Koyama, M.: Optuna: A next-generation hyperparameter optimization framework. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (2019)

[15] Akiba, T., Sano, S., Yanase, T., et al.: Optuna: A next-generation hyperparameter optimization framework. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 2623–2631 (2019). https://doi.org/10.1145/3292500.3330701

[16] Ali, A., Rehman, N., Ali, M., Hila, K.: A novel approach to three-way decision model under fuzzy soft dominance degree relations and emergency situation. Expert Systems with Applications **239**, 122369 (2024). https://doi.org/10.1016/j.eswa.2023.122369

[17] de Almeida, A.: Multicriteria decision model for outsourcing contracts selection based on utility function and electre method. Computers and Operations Research **34**(12), 3569–3574 (2007)

[18] Alsalem, M.A., Albahri, O.S., Zaidan, A.A., Al-Obaidi, J.R., Alnoor, A., Alamoodi, A.H., Albahri, A.S., Zaidan, B.B., Jumaah, F.M.: Rescuing emergency cases of covid-19 patients: An intelligent real-time msc transfusion framework based on multicriteria decision-making methods. Applied Intelligence **52**, 9676–9700 (2022). https://doi.org/10.1007/s10489-021-02813-5

[19] Aminjarahi, M., Abdoli, M., Fadaee, Y., Kohan, F., Shokouhyar, S.: The prioritization of lean techniques in emergency departments using vikor and saw approaches. Ethiopian Journal of Health Sciences **31**(2), 283–292 (2021). https://doi.org/10.4314/ejhs.v31i2.11

[20] Anselin, A., Meire, P., Anselin, L.: Multicriteria techniques in ecological evaluation: an example using the analytical hierarchy process. Biological Conservation **49**(3), 215–229 (1989)

[21] Arabameri, A., Yamani, M., Pradhan, B., Melesse, A., Shirani, K., Bui, D.T.: Novel ensembles of copras multi-criteria decision-making with logistic regression, boosted regression tree, and random forest for spatial prediction of gully erosion susceptibility. Science of the total environment **688**, 903–916 (2019)

[22] Arrieta, A.B., Díaz-Rodríguez, N., Del Ser, J., Bennetot, A., Tabik, S., Barbado, A., García, S., Gil-López, S., Molina, D., Benjamins, R., et al.: Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai. Information fusion **58**, 82–115 (2020)

[23] Auer, P., Cesa-Bianchi, N., Fischer, P.: Finite-time analysis of the multiarmed bandit problem. Machine learning **47**, 235–256 (2002)

[24] Authority, C.A.: Flight-crew human factors handbook. CAP **737**, 55–70 (2014)

[25] Bäck, T.: Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms. Oxford University Press (1996)

[26] Baeza-Yates, R., Ribeiro-Neto, B.: Modern Information Retrieval. Addison-Wesley Longman Publishing Co., Inc. (1999)

[27] Bait, S., Lauria, S.M., Schiraldi, M.M.: Multi-criteria decision-making model for supporting manufacturing settlements location in africa after covid-19. International Journal of Engineering Business Management **13**, 18479790211023348 (2021). https://doi.org/10.1177/18479790211023348

[28] Bankes, S.C.: Exploratory modeling for policy analysis. Operations Research **41**(3), 435–449 (1993)

[29] Barbarosoglu, G., Yazgac, T.: An application of the analytic hierarchy process to the supplier selection problem. Production and Inventory Management Journal **38**(1), 14 (1997)

[30] Bartholomew, E., Kwakkel, J.H.: On considering robustness in the search phase of robust decision making: A comparison of many-objective robust decision making, multi-scenario many-objective robust decision making, and many objective robust optimization. Environmental Modelling & Software **127**, 104699 (2020)

[31] Behzadian, M., Otaghsara, S.K., Yazdani, M., Ignatius, J.: A state-of-the-art survey of topsis applications. Expert Systems with Applications **39**(13), 13051–13069 (2012)

[32] Belton, V., Stewart, T.: Multiple Criteria Decision Analysis: An Integrated Approach. Springer (2002)

[33] Benayoun, R., Roy, B., Sussman, B.: Electre: Une méthode pour guider le choix en présence de points de vue multiples. Tech. Rep. 49, SEMA (Metra International) (1966)

[34] Bentéjac, C., Csörgő, A., Martínez-Muñoz, G.: A comparative analysis of gradient boosting algorithms. arXiv preprint arXiv:1911.01914 (2019)

[35] Biran, O., Cotton, C.: Explanation and justification in machine learning: A survey. In: IJCAI-17 workshop on explainable AI (XAI). vol. 8, pp. 8–13 (2017)

[36] Birlutiu, A., Groot, P., Heskes, T.: Efficiently learning the preferences of people. Machine Learning **90**, 1–28 (2013)

[37] Bittermann, V., Deker, G., Sassus, P., Mielnik, J.C., Jud, J.M.: Finder, a system providing complex decision support for commercial transport replanning operations. IEEE Aerospace and Electronic Systems Magazine **9**(3), 12–19 (2002)

[38] Blank, J., Deb, K.: pymoo: Multi-objective optimization in python. IEEE Access **8**, 89497–89509 (2020)

[39] Boodhun, N., Jayabalan, M.: Risk prediction in life insurance industry using supervised learning algorithms. Complex & Intelligent Systems **4**(2), 145–154 (2018)

[40] Bozóki, S., Fülöp, J., Rónyai, L.: On optimal completion of incomplete pairwise comparison matrices. Mathematical and computer modelling **52**(1-2), 318–333 (2010)

[41] Bozorg-Haddad, O., Zolghadr-Asli, B., Loáiciga, H.A.: A handbook on multi-attribute decision-making methods. John Wiley & Sons (2021)

[42] Bozorg-Haddad, O., Zolghadr-Asli, B., Loáiciga, H.A.: A Handbook on Multi-Attribute Decision-Making Methods. John Wiley & Sons, Hoboken, NJ (2021). https://doi.org/10.1002/9781119563501

[43] Breiman, L.: Random forests. Machine Learning **45**(1), 5–32 (2001)

[44] Breiman, L., Friedman, J., Stone, C.J., Olshen, R.A.: Classification and Regression Trees. Wadsworth International Group (1984)

[45] Brent, A., Rogers, D., Ramabitsa-Siimane, T., Rohwer, M.: Application of the analytical hierarchy process to establish health care waste management systems that minimise infection risks in developing countries. European Journal of Operational Research **181**(1), 403–424 (2007)

[46] Bureau of Transportation Statistics: U.s. bureau of transportation statistics: June 2024 data. Tech. rep., U.S. Department of Transportation (2024), https://www.transtats.bts.gov/

[47] Burges, C.J.: From ranknet to lambdarank to lambdamart: An overview. In: Microsoft Research Technical Report MSR-TR-2010-82 (2010)

[48] Burges, C.J., Shaked, T., Renshaw, E., Lazier, A., Deeds, M., Hamilton, N., Hullender, G.: Learning to rank using gradient descent. In: Proceedings of the 22nd international conference on Machine learning (ICML). pp. 89–96 (2005)

[49] Butz, M.V., Lanzi, P.L., Wilson, S.W.: Hyper-ellipsoidal conditions in xcs: Rotation, linear approximation, and solution structure. In: Proceedings of the 8th annual conference on Genetic and evolutionary computation. pp. 1457–1464 (2006)

[50] Cai, S., Zheng, K., Chen, G., Jagadish, H., Ooi, B.C., Zhang, M.: Armnet: Adaptive relation modeling network for structured data. In: Proceedings of the 2021 International Conference on Management of Data. pp. 207–220 (2021)

[51] Cao, Z., Qin, T., Liu, T.Y., Tsai, J.H., Li, H.: Learning to rank: from pairwise approach to listwise approach. In: Proceedings of the 24th international conference on Machine learning (ICML). pp. 129–136 (2007)

[52] Cassimon, T., Eyckerman, R., Mercelis, S., Latré, S., Hellinckx, P.: A survey on discrete multi-objective reinforcement learning benchmarks. In: Proceedings of the Adaptive and Learning Agents Workshop (ALA 2022) (2022)

[53] Caylor, J.P., II, R.J.H.: Utilization of multi-criteria decision-making for emergency management. Computación y Sistemas **25**(4), 863–872 (2021). https://doi.org/10.13053/CyS-25-4-4102

[54] Chapelle, O., Keerthi, S.S.: Efficient algorithms for ranking with svms. Information retrieval **13**(3), 201–215 (2010)

[55] Charnes, A., Cooper, W.W.: Management models and industrial applications of linear programming. Management Science **7**(4), 455–469 (1961)

[56] Chen, H., Covert, I.C., Lundberg, S.M., Lee, S.I.: Algorithms to estimate shapley value feature attributions. Nature Machine Intelligence **5**(6), 590–601 (2023)

[57] Chen, T., Guestrin, C.: Xgboost: A scalable tree boosting system. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 785–794. ACM (2016)

[58] Chen, W., Wang, X., Wang, W., Zhu, Y., Cai, Z., Yang, S.: A heterogeneous gra-cbr-based multi-attribute emergency decision-making model considering weight optimization with dual information correlation. Expert Systems with Applications **182**, 115208 (2021). https://doi.org/10.1016/j.eswa.2021.115208

[59] Choquet, G.: Theory of capacities. In: Annales de l'institut Fourier. vol. 5, pp. 131–295 (1954)

[60] Civil Aviation Safety Authority (CASA), Australia: Medical Assessment for Aviation: Human Factors Handbook. Civil Aviation Safety Authority, Canberra, Australia (2024)

[61] Coello, C.A.C.: Evolutionary algorithms for solving multi-objective problems. Springer (2007)

[62] Council, N.R., of Behavioral, D., Sciences, S., on the Human Dimensions of Global Change, C., on Strategies, P., for Climate-Related Decision Support, M.: Informing decisions in a changing climate. National Academies Press (2009)

[63] Cybenko, G.: Approximation by superpositions of a sigmoidal function. Mathematics of Control, Signals and Systems **2**(4), 303–314 (1989)

[64] Darko, A.P., Liang, D.: Modeling customer satisfaction through online reviews: A flowsort group decision model under probabilistic linguistic settings. Expert Systems with Applications **195**, 116649 (2022)

[65] David, L., Duckstein, L.: Multi-criterion ranking of alternative long-range water resources systems. Journal of the American Water Resources Association **12**(4), 731–754 (1976)

[66] Davodabadi, A., Daneshian, B., Saati, S., Razavyan, S.: Prioritization of patients in icu: Composite approach of multiple-criteria decision-making and discrete event simulation. Brazilian Journal of Operations Production Management **18**(1), 1–21 (2021). https://doi.org/10.14488/BJOPM.2021.008

[67] Day, J.: What is an expert? Radiography **8**(2), 63–70 (2002)

[68] Deb, K., Thiele, L., Laumanns, M., Zitzler, E.: Scalable multi-objective optimization test problems. In: Proceedings of the 2002 Congress on Evolutionary Computation. CEC'02 (Cat. No.02TH8600). vol. 1, pp. 825–830 vol.1 (2002). https://doi.org/10.1109/CEC.2002.1007032

[69] Deb, K.: Nonlinear goal programming using multi-objective genetic algorithms. Journal of the Operational Research Society **52**(3), 291–302 (2001)

[70] Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: Nsga-ii. IEEE Transactions on Evolutionary Computation **6**(2), 182–197 (2002)

[71] Deb, K., Thiele, L., Laumanns, M., Zitzler, E.: Scalable multi-objective optimization test problems. Proceedings of the 2005 Congress on Evolutionary Computation pp. 825–830 (2005)

[72] Demircan, B.G., Yetilmezsoy, K.: A hybrid fuzzy ahp-topsis approach for implementation of smart sustainable waste management strategies. Sustainability **15**(8), 6526 (2023)

[73] Deng, H., Yeh, C.H., Willis, R.J.: Inter-company comparison using modified topsis with objective weights. Computers Operations Research **27**(10), 963–973 (2000)

[74] Deparday, V., Gevaert, C.M., Molinario, G., Soden, R., Balog-Way, S.: Machine learning for disaster risk management (2019)

[75] Dewar, J.A.: Assumption-Based Planning: A Tool for Reducing Avoidable Surprises. Cambridge University Press (2002)

[76] Djartov, B., Mostaghim, S.: Multi-objective multiplexer decision making benchmark problem. In: Proceedings of the Companion Conference on Genetic and Evolutionary Computation. pp. 1676–1683 (2023)

[77] Djartov, B., Mostaghim, S., Papenfuß, A., Wies, M.: Description and first evaluation of an approach for a pilot decision support system based on multi-attribute decision making. In: 2022 IEEE Symposium Series on Computational Intelligence (SSCI). pp. 141–147. IEEE (2022)

[78] Djartov, B., Mostaghim, S., Papenfuß, A., Wies, M.: A learning classifier system approach to time-critical decision-making in dynamic alternate airport selection. In: 2024 IEEE Congress on Evolutionary Computation (CEC). pp. 1–8. IEEE (2024)

[79] Djartov, B., Oswald, F.M., Jakobi, J.: Through the psychological lens: Unveiling biases in multi-criteria decision-making. In: International Conference on Human Interaction & Emerging Technologies IHIET (2024)

[80] Djartov, B., Papenfuß, A., Wies, M.: Wings of wisdom: Learning from pilot decision data with interpretable ai models. In: International Conference on Human-Computer Interaction. pp. 241–256. Springer (2024)

[81] Djartov, B., Würfel, J.: Navigating Decisions in the Cockpit: The Intelligent Pilot Advisory System, pp. 305–325. Springer Nature Switzerland, Cham (2025). https://doi.org/10.1007/978-3-031-83512-4_18, https://doi.org/10.1007/978-3-031-83512-4_18

[82] Dong, J., Ota, K., Dong, M.: Uav-based real-time survivor detection system in post-disaster search and rescue operations. IEEE Journal on Miniaturization for Air and Space Systems **2**(4), 209–219 (2021)

[83] Doshi-Velez, F., Kim, B.: Towards a rigorous science of interpretable machine learning. arXiv preprint arXiv:1702.08608 (2017)

[84] Dugger, Z., Halverson, G., McCrory, B., Claudio, D.: Principal component analysis in mcdm: An exercise in pilot selection. Expert Systems with Applications **188**, 115984 (2022)

[85] Eelagh, M.D., Abbaspour, R.A.: A location-allocation optimization model for post-earthquake emergency shelters using network-based multi-criteria decision-making. Decision Analytics Journal **10**, 100430 (2024). https://doi.org/10.1016/j.dajour.2024.100430

[86] Eker, S., Kwakkel, J.H.: Including robustness considerations in the search phase of many-objective robust decision making. Environmental Modelling & Software **105**, 201–216 (2018)

[87] Eshkevari, M., Rezaee, M.J., Saberi, M., Hussain, O.K.: An end-to-end ranking system based on customers reviews: Integrating semantic mining and mcdm techniques. Expert Systems with Applications **209**, 118294 (2022)

[88] Eurocontrol: Aircraft performance details: Airbus a320 (2024), https://contentzone.eurocontrol.int/aircraftperformance/details.aspx?ICAO=A320, accessed: 2024-07-16

[89] EUROCONTROL: Eurocontrol standard inputs for economic analysis. Tech. rep., EUROCONTROL, 96 Rue de la Fusée, B-1130 Brussels (2024), https://ansperformance.eu/economics/cba/standard-inputs/chapters/cost_of_diversion.html, tel: +32 (0)2 729 1152, Fax: +32 (0)2 729 514

[90] Eurocontrol Experimental Center: Revisiting the "swiss cheese" model of accidents. Tech. rep., Eurocontrol (2006), https://www.eurocontrol.int/publication/revisiting-swiss-cheese-model-accidents, accessed: 2024-07-16

[91] European Union Aviation Safety Agency: Artificial intelligence roadmap 2.0: A human-centric approach to ai in aviation (May 2023), https://www.easa.europa.eu/en/document-library/general-publications/easa-artificial-intelligence-roadmap-20, accessed: 2025-06-03

[92] Evans, J.S.B.: Dual-processing accounts of reasoning, judgment, and social cognition. Annual Review of Psychology **59**, 255–278 (2008)

[93] Fakhry, A.E., Ali, A.M., Abdelhafeez, A., Tantawy, A.A.: Evaluation the key success factors in emergency management using neutrosophic dematel method. International Journal of Neutrosophic Science **21**, 195–207 (2023). https://doi.org/10.54216/IJNS.21015

[94] Fari, S., Wang, X., Roy, S., Baldi, S.: Addressing unmodeled path-following dynamics via adaptive vector field: A uav test case. IEEE Transactions on Aerospace and Electronic Systems **56**(2), 1613–1622 (2019)

[95] Feng, Z., Li, G., Wang, W., Zhang, L., Xiang, W., He, X., Zhang, M., Wei, N.: Emergency logistics centers site selection by multi-criteria decision-making and gis. International Journal of Disaster Risk Reduction **96**, 103921 (2023). https://doi.org/10.1016/j.ijdrr.2023.103921

[96] Ferreiro-Cabello, J., Fraile-Garcia, E., de Pison Ascacibar, E.M., Martinez-de Pison, F.: Metamodel-based design optimization of structural one-way slabs based on deep learning neural networks to reduce environmental impact. Engineering Structures **155**, 91–101 (2018)

[97] Figueira, J., Greco, S., Roy, B., Słowiński, R.: An overview of electre methods and their recent extensions. Journal of Multi-Criteria Decision Analysis **20**(1-2), 61–85 (2013)

[98] Fisher, A., Rudin, C., Dominici, F.: All models are wrong, but many are useful: Learning a variable's importance by studying an entire class of prediction models simultaneously. Journal of Machine Learning Research **20**(177), 1–81 (2019)

[99] Fisher, R.A.: Frequency distribution of the values of the correlation coefficient in samples from an indefinitely large population. Biometrika **10**(4), 507–521 (1915)

[100] Flather, G.W.: A study of decision-making behavior of aircraft pilots deviating from a planned flight. Tech. Rep. NASA-CR-166057, NASA Ames Research Center (1981), https://ntrs.nasa.gov/citations/19820062727

[101] Flightradar24: Flightradar24. https://www.flightradar24.com/50.13,8.29/11, accessed: 2025-07-08

[102] Fogel, D.B., Computation, E.: Toward a new philosophy of machine intelligence. IEEE Evolutionary Computation **1080** (1995)

[103] Ford, D.N., Wolf, C.M.: Smart cities with digital twin systems for disaster management. Journal of management in engineering **36**(4), 04020027 (2020)

[104] Friedman, J.H.: Greedy function approximation: A gradient boosting machine. Annals of Statistics **29**(5), 1189–1232 (2001)

[105] Gigerenzer, G., Gaissmaier, W.: Heuristic decision making. Annual Review of Psychology **62**, 451–482 (2011)

[106] Goldberg, D.E.: Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley Longman Publishing Co., Inc. (1989)

[107] Goldberg, D.E., Deb, K.: A comparative analysis of selection schemes used in genetic algorithms. Foundations of Genetic Algorithms **1**, 69–93 (1991)

[108] Goodman, B., Flaxman, S.: European union regulations on algorithmic decision-making and a "right to explanation". AI magazine **38**(3), 50–57 (2017)

[109] Govindan, K., Jepsen, M.B.: Electre: A comprehensive literature review on methodologies and applications. European Journal of Operational Research **250**(1), 1–29 (2016)

[110] Green, K.C., Armstrong, J.S.: Value of expertise for forecasting decisions in conflicts. Monash University Econometrics and Business Statistics Working Paper (27/04) (2004)

[111] Guo, J., Fan, Y., Ai, Q., Croft, W.B.: A deep look into neural ranking models for information retrieval. Information Processing & Management **57**(6), 102067 (2020)

[112] Guo, M., Zhang, Q., Liao, X., Chen, F.Y., Zeng, D.D.: A hybrid machine learning framework for analyzing human decision-making through learning preferences. Omega **101**, 102263 (2021)

[113] Hajilo, M., Talkhab, A., Pennington-Gray, L.: Spatial analysis of earthquake-prone rural areas and residents' preparedness. Natural Hazards **120**(5), 4101–4130 (2024). https://doi.org/10.1007/s11069-023-06364-5

[114] Han, S., Jia, X., Chen, X., Gupta, S., Kumar, A., Lin, Z.: Search well and be wise: A machine learning approach to search for a profitable location. Journal of Business Research **144**, 416–427 (2022)

[115] Harries, C., Harvey, N.: Taking advice, using information and knowing what you are doing. Acta Psychologica **104**(3), 399–416 (2000)

[116] Hastie, T., Tibshirani, R., Friedman, J.H., Friedman, J.H.: The elements of statistical learning: data mining, inference, and prediction, vol. 2. Springer (2009)

[117] Hatami-Marbini, A., Tavana, M.: An extension of the electre i method for group decision-making under a fuzzy environment. Omega **39**(4), 373–386 (2011)

[118] Haykin, S.: Neural Networks and Learning Machines. Pearson, 3 edn. (2009)

[119] Hazelrigg, G.A.: A note on the weighted sum method. Journal of Mechanical Design **141**(10), 100301 (2019). https://doi.org/10.1115/1.4043800

[120] He, Z., Tran, K.P., Thomassey, S., Zeng, X., Xu, J., Yi, C.: A deep reinforcement learning based multi-criteria decision support system for optimizing textile chemical process. Computers in Industry **125**, 103373 (2021)

[121] Hellmann, G.: Uber die bewegung der luft in den untersten schichten der atmosphäre. Meteorologische Zeitschrift **34**, 273–285 (1916), introduced the wind profile power law (Hellmann exponent) for estimating wind speed at different heights

[122] Hill, S., Ready-Campbell, N.: Expert stock picker: the wisdom of (experts in) crowds. International Journal of Electronic Commerce **15**(3), 73–102 (2011)

[123] Hoekstra, J., Ellerbroek, J.: Bluesky atc simulator project: an open data and open source approach. In: Proceedings of the 7th International Conference on Research in Air Transportation (ICRAT) (2016)

[124] Hogarth, R.M., Lejarraga, T., Soyer, E.: The two settings of kind and wicked learning environments. Current Directions in Psychological Science **24**(5), 379–385 (2015)

[125] Holland, J.H.: Escaping brittleness: the possibilities of general-purpose learning algorithms applied to parallel rule-based systems. In: Machine Learning, pp. 593–623. Springer (1986)

[126] Hornik, K., Stinchcombe, M., White, H.: Multilayer feedforward networks are universal approximators. Neural Networks **2**(5), 359–366 (1989)

[127] Hsu, C.C., Chang, H.C., Li, Y.C., Liou, J.J.: Developing an airport resilience assessment model for climate change. Journal of Air Transport Management **119**, 102646 (2024). https://doi.org/10.1016/j.jairtraman.2024.102646

[128] Hwang, C.L., Yoon, K.: Multiple attribute decision making: methods and applications. Springer-Verlag (1981)

[129] IATA: IATA world air transport statistics 2024: Most used aircraft – airbus a321 usage (2025), the Airbus A321 logged 3.4 million flights and 1.1 trillion ASKs in 2024, ranking it among the most-used narrow-body aircraft globally

[130] Ince, M.: Bilstm and dynamic fuzzy ahp-ga method for procedural game level generation. Neural Computing and Applications **33**(15), 9761–9773 (2021)

[131] Järvelin, K., Kekäläinen, J.: Cumulated gain-based evaluation of ir techniques. ACM Transactions on Information Systems (TOIS) **20**(4), 422–446 (2002)

[132] Jiang, G.J., Chen, H.X., Sun, H.H., Yazdi, M., Nedjati, A., Adesina, K.A.: An improved multi-criteria emergency decision-making method in environmental disasters. Soft Computing **25**(15), 10351–10379 (2021). https://doi.org/10.1007/s00500-021-05826-x

[133] Joachims, T.: Optimizing search engines using clickthrough data. In: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 133–142 (2002)

[134] Jomthanachai, S., Wong, W.P., Khaw, K.W.: An application of machine learning regression to feature selection: a study of logistics performance and economic attribute. Neural Computing and Applications **34**(18), 15781–15805 (2022)

[135] Kahneman, D.: Thinking, fast and slow. New York: Farrar, Straus and Giroux (2011)

[136] Kahraman, C., Onar, S.C., Oztaysi, B.: Fuzzy multicriteria decision-making: A literature review. International Journal of Computational Intelligence Systems **8**(4), 637–666 (2015)

[137] Kasprzyk, J.R., Nataraj, S., Reed, P.M., Lempert, R.J.: Many objective robust decision making for complex environmental systems undergoing change. Environmental Modelling & Software **42**, 55–71 (2013)

[138] Kavitha, T., Saraswathi, S.: Smart technologies for emergency response and disaster management: new sensing technologies or/and devices for emergency response and disaster management. In: Smart Technologies for Emergency Response and Disaster Management, pp. 1–40. IGI Global (2018)

[139] Kazemi, S., Azad, N.L., Scott, K.A., Oqab, H.B., Dietrich, G.B.: Satellite collision avoidance maneuver planning in low earth orbit using proximal policy optimization. In: 2024 IEEE Congress on Evolutionary Computation (CEC). pp. 1–9. IEEE (2024)

[140] Keeney, R.L., Raiffa, H.: Decisions with Multiple Objectives: Preferences and Value Trade-Offs. Cambridge University Press (1993)

[141] Keul, M.: Are there any factors that make the pilots' decisions predictable? An analysis about the influence of the conditions on the choice of alternate airports. Master's thesis, Hochschule Fresenius, Frankfurt am Main (2023)

[142] Kim, B., Khanna, R., Koyejo, O.O.: Examples are not enough, learn to criticize! criticism for interpretability. Advances in neural information processing systems **29** (2016)

[143] Kim, S.W., Melby, J.A., Nadal-Caraballo, N.C., Ratcliff, J.: A time-dependent surrogate model for storm surge prediction based on an artificial neural network using high-fidelity synthetic hurricane modeling. Natural Hazards **76**, 565–585 (2015)

[144] Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)

[145] Kleywegt, A.J., Shapiro, A., Homem-de Mello, T.: The sample average approximation method for stochastic discrete optimization. SIAM Journal on optimization **12**(2), 479–502 (2002)

[146] Koza, J.R.: Genetic programming as a means for programming computers by natural selection. Statistics and computing **4**, 87–112 (1994)

[147] Kukreja, V.: Hybrid fuzzy ahp–topsis approach to prioritizing solutions for inverse reinforcement learning. Complex & Intelligent Systems **9**(1), 493–513 (2023)

[148] Kumar, A., Sah, B., Singh, A., et al.: A review of multi criteria decision making (mcdm) towards sustainable renewable energy development. Renewable and Sustainable Energy Reviews **69**, 596–609 (2017)

[149] Kuznetsova, E., Li, Y.F., Ruiz, C., Zio, E., Ault, G., Bell, K.: Reinforcement learning for microgrid energy management. Energy **59**, 133–146 (2013)

[150] Kyrkou, C., Kolios, P., Theocharides, T., Polycarpou, M.: Machine learning for emergency management: A survey and future outlook. Proceedings of the IEEE **111**(1), 19–41 (2022)

[151] Laminar Research: X-plane airport and navigation data. Laminar Research website (2025), https://developer.x-plane.com/data/, airport and navigation data used within the X-Plane flight simulator. Accessed: 2025-08-27

[152] Leising, D., Borgstede, M.: Hypothetical constructs. In: Encyclopedia of personality and individual differences, pp. 1–6. Springer (2019)

[153] Lempert, R.J.: Robust Decision Making (RDM). Springer (2019)

[154] Lempert, R.J., Bryant, B.P., Collins, M.T., Hackbarth, A., LaTourrette, T., Reville, R.T., Popper, S.W., Mijere, C., Groves, D.G., Keller, K., et al.: Making good decisions without predictions: Robust decision making for planning under deep uncertainty (2013)

[155] Lempert, R.J., Groves, D.G., Popper, S.W., Bankes, S.C.: A general, analytic method for generating robust strategies and narrative scenarios. Management Science **52**(4), 514–528 (2006)

[156] Lempert, R.J., Popper, S.W., Bankes, S.C.: Shaping the next one hundred years: new methods for quantitative. Long-Term Policy Analysis **208** (2003)

[157] Levy, H.: Stochastic Dominance: Investment Decision Making under Uncertainty. Kluwer Academic Publishers, Dordrecht, 2nd edn. (1992)

[158] Levy, H.: Stochastic Dominance: Investment Decision Making under Uncertainty. Springer, 3rd edn. (2015)

[159] Li, H.: A short introduction to learning to rank. IEICE Transactions on Information and Systems **94**(10), 1854–1862 (2011)

[160] Li, L., Chu, W., Langford, J., Schapire, R.E.: A contextual-bandit approach to personalized news article recommendation. In: Proceedings of the 19th international conference on World wide web. pp. 661–670 (2010)

[161] Li, T., Sun, J., Fei, L.: Application of multiple-criteria decision-making technology in emergency decision-making: Uncertainty, heterogeneity, dynamicity, and interaction. Mathematics **13**(5), 731 (2025). https://doi.org/10.3390/math13050731, https://doi.org/10.3390/math13050731

[162] Liang, H., Dong, Y., He, Y.: Interpretable preference learning and prediction: A data-driven method based on multiplicative multiattribute utility function with reference effects. Information Fusion p. 103311 (2025)

[163] Liao, H., He, Y., Wu, X., Wu, Z., Bausys, R.: Reimagining multi-criterion decision making by data-driven methods based on machine learning: A literature review. Information Fusion **100**, 101970 (2023)

[164] Liao, Z., Wang, B., Xia, X., Hannam, P.M.: Environmental emergency decision support system based on artificial neural network. Safety Science **50**(1), 150–163 (2012)

[165] Lim, M.C., Ayoko, G.A., Morawska, L., Ristovski, Z.D., Jayaratne, E.R., Kokot, S.: A comparative study of the elemental composition of the exhaust emissions of cars powered by liquefied petroleum gas and unleaded petrol. Atmospheric Environment **40**(17), 3111–3122 (2006)

[166] LimeSurvey GmbH: LimeSurvey: An Open Source Survey Tool. LimeSurvey GmbH (2025), https://www.limesurvey.org, version 6.x

[167] Lipton, Z.C.: The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery. Queue **16**(3), 31–57 (2018)

[168] Liu, B., Chen, Y., Shen, Y., Sun, H., Xu, X.: A complex multi-attribute large-group decision making method based on the interval-valued intuitionistic fuzzy principal component analysis model. Soft Computing **18**, 2149–2160 (2014)

[169] Liu, J., Jiang, D., Guo, L., Nan, J., Cao, W., Wang, P.: Emergency material location-allocation planning using a risk-based integration methodology for river chemical spills. Environmental Science and Pollution Research **27**(15), 17949–17962 (2020). https://doi.org/10.1007/s11356-020-08331-0

[170] Liu, K.: Gis-based mcdm framework combined with coupled multi-hazard assessment for site selection of post-earthquake emergency medical service facilities in wenchuan, china. International Journal of Disaster Risk Reduction **73**, 102873 (2022). https://doi.org/10.1016/j.ijdrr.2022.102873

[171] Liu, Q., Kou, Y., Huang, Z.: A comprehensive evaluation approach of navigation signal performance based on multi-attribute group decision making. In: China Satellite Navigation Conference (CSNC) 2016 Proceedings: Volume II. pp. 15–27. Springer (2016)

[172] Liu, T.Y.: Learning to Rank for Information Retrieval. Foundations and Trends in Information Retrieval, Springer (2009)

[173] Lu, S., Christie, G.A., Nguyen, T.T., Freeman, J.D., Hsu, E.B.: Applications of artificial intelligence and machine learning in disasters and public health emergencies. Disaster medicine and public health preparedness **16**(4), 1674–1681 (2022)

[174] Lundberg, S.M., contributors: Shap: Shapley additive explanations. https://github.com/shap/shap (2025), gitHub repository. Accessed 2025-09-27

[175] Lundberg, S.M., Erion, G., Chen, H., DeGrave, A., Prutkin, J.M., Nair, B., Katz, R., Himmelfarb, J., Bansal, N., Lee, S.I.: From local explanations to global understanding with explainable ai for trees. Nature machine intelligence **2**(1), 56–67 (2020)

[176] Lundberg, S.M., Lee, S.I.: A unified approach to interpreting model predictions. Advances in neural information processing systems **30** (2017)

[177] Lygouras, E., Santavas, N., Taitzoglou, A., Tarchanidis, K., Mitropoulos, A., Gasteratos, A.: Unsupervised human detection with an embedded vision system on a fully autonomous uav for search and rescue operations. Sensors **19**(16), 3542 (2019)

[178] Macharis, C., Bernardini, A.: Evaluation of transport projects in a multi-actor multi-criteria framework: The mamca method. Transport Policy **37**, 177–186 (2015)

[179] Maciąg, K., Urbanowicz, R.J.: Xcsf python package. https://pypi.org/project/xcsf/ (2024), version X.Y.Z, accessed 2024-07-16

[180] Mao, Q., Chen, J., Lv, J., Chen, S.: Emergency plan selection for epidemic prevention and control based on cumulative prospect theory and hybrid-information madm. Kybernetes **52**(6), 1903–1933 (2023). https://doi.org/10.1108/K-08-2021-0736

[181] Marchau, V.A., Walker, W.E., Bloemen, P.J., Popper, S.W.: Decision making under deep uncertainty: from theory to practice. Springer Nature (2019)

[182] Marsh, K., IJzerman, M., Thokala, P., Baltussen, R., Boysen, M., Kaló, Z., Longrenn, T., Mussen, F., Peacock, S., Watkins, J.: Multi-criteria decision analysis to support healthcare decisions. Pharmacoeconomics **32**(4), 345–365 (2014)

[183] Martins, A.P., Köbrich, M.V., Carstengerdes, N., Biella, M.: All's well that ends well? outcome bias in pilots during instrument flight rules. Applied Cognitive Psychology **37**(2), 433–442 (2023)

[184] Marzouk, M., Hassan, F.: Modeling evacuation and visitation proximity in museums using agent-based simulation. Journal of Building Engineering **56**, 104794 (2022)

[185] Meske, C., Bunde, E., Schneider, J., Gersch, M.: Explainable artificial intelligence: objectives, stakeholders, and future research opportunities. Information systems management **39**(1), 53–63 (2022)

[186] Minerva, R., Lee, G.M., Crespi, N.: Digital twin in the iot context: A survey on technical features, scenarios, and architectural models. Proceedings of the IEEE **108**(10), 1785–1824 (2020)

[187] Mitchell, T.M.: Machine Learning. McGraw-Hill (1997)

[188] Modibbo, U.M., Hassan, M., Ahmed, A., Ali, I.: Multi-criteria decision analysis for pharmaceutical supplier selection problem using fuzzy topsis. Management Decision **60**(3), 806–836 (2022)

[189] Moghtadernejad, S., Chouinard, L.E., Mirza, M.S.: Enhanced façade design: A data-driven approach for decision analysis based on past experiences. Developments in the Built Environment **5**, 100038 (2021)

[190] Mohammadnazari, Z., Mamoudan, M.M., Alipour-Vaezi, M., Aghsami, A., Jolai, F., Yazdani, M.: Prioritizing post-disaster reconstruction projects using an integrated multi-criteria decision-making approach: A case study. Buildings **12**(2), 136 (2022). https://doi.org/10.3390/buildings12020136

[191] Mohebbi, A., Achiche, S., Baron, L.: Multi-criteria fuzzy decision support for conceptual evaluation in design of mechatronic systems: a quadrotor design case study. Research in Engineering Design **29**, 329–349 (2018)

[192] Molnar, C.: Interpretable Machine Learning: A Guide for Making Black Box Models Explainable. Leanpub, 2 edn. (2022), https://christophm.github.io/interpretable-ml-book/

[193] Molnar, C., Casalicchio, G., Bischl, B.: Interpretable machine learning–a brief history, state-of-the-art and challenges. In: Joint European conference on machine learning and knowledge discovery in databases. pp. 417–431. Springer (2020)

[194] Molnar, C., Casalicchio, G., Bischl, B.: Interpretable machine learning – a brief history, state-of-the-art and challenges. arXiv preprint arXiv:2010.09337 (2020)

[195] Montgomery, D.C., Peck, E.A., Vining, G.G.: Introduction to Linear Regression Analysis. John Wiley & Sons (2012)

[196] Moustra, M., Avraamides, M., Christodoulou, C.: Artificial neural networks for earthquake prediction using time series magnitude data or seismic electric signals. Expert systems with applications **38**(12), 15032–15039 (2011)

[197] Murphy, K.P.: Probabilistic machine learning: an introduction. MIT press (2022)

[198] Nagy, M., ao Luís de Miranda, J., Popescu-Bodorin, N.: Decision making and robust optimization for information systems oriented to emergency events. International Journal of Computers Communications & Control **19**(6), 6861 (2024). https://doi.org/10.15837/ijccc.2024.6.6861

[199] von Neumann, J., Morgenstern, O.: Theory of Games and Economic Behavior. Princeton University Press (1947)

[200] Nguyen, C.T., Saputra, Y.M., Van Huynh, N., Nguyen, N.T., Khoa, T.V., Tuan, B.M., Nguyen, D.N., Hoang, D.T., Vu, T.X., Dutkiewicz, E., et al.: A comprehensive survey of enabling and emerging technologies for social distancing—part i: Fundamentals and enabling technologies. Ieee Access **8**, 153479–153507 (2020)

[201] Nuić, A., Poles, D., Mouillet, V.: Bada: An advanced aircraft performance model for present and future atm systems. International Journal of Adaptive Control and Signal Processing **24**(10), 891–911 (2010). https://doi.org/10.1002/acs.1176

[202] Ok, K., Okan, T., Yilmaz, E.: A comparative study on activity selection with multi-criteria decision-making techniques in ecotourism planning. Scientific Research and Essays **6**(6), 1417–1427 (2011)

[203] Onken, R.: The cognitive cockpit assistant systems cassy/cama. Tech. rep., SAE Technical Paper (1999)

[204] Online Flight Planner: Online flight planner. http://onlineflightplanner.org/, accessed: 2025-07-08

[205] Ortiz-Barrios, M.A., Alfaro-Saiz, J.J.: A hybrid fuzzy multi-criteria decision-making model to evaluate the overall performance of public emergency departments: A case study. International Journal of Information Technology & Decision Making **19**(6), 1485–1548 (2020). https://doi.org/10.1142/S0219622020500364

[206] Otay, I., Jaller, M.: Multi-expert disaster risk management & response capabilities assessment using interval-valued intuitionistic fuzzy sets. Journal of Intelligent & Fuzzy Systems **38**(1), 835–852 (2020). https://doi.org/10.3233/JIFS-179452

[207] Pagano, A., Giordano, R., Vurro, M.: A decision support system based on ahp for ranking strategies to manage emergencies on drinking water supply systems. Water Resources Management **35**(2), 613–628 (2021). https://doi.org/10.1007/s11269-020-02741-y

[208] Parkan, C., Wu, M.L.: Decision-making and performance measurement models with applications to robot selection. Computers Industrial Engineering **36**(3), 503–523 (1999)

[209] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.: Pytorch: An imperative style, high-performance deep learning library. In: Advances in Neural Information Processing Systems 32, pp. 8024–8035. Curran Associates, Inc. (2019), http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-libr pdf

[210] Pathan, A.I., Agnihotri, P.G., Said, S., Patel, D.: Ahp and topsis based flood risk assessment—a case study of the navsari city, gujarat, india. Environmental Monitoring and Assessment **194**(7), 509 (2022). https://doi.org/10.1007/s10661-022-10111-x

[211] Pegoraro, F., Santos, E.A.P., de Freitas Rocha Loures, E., Laus, F.W.: A hybrid model to support decision making in emergency department management. Knowledge-Based Systems **203**, 106148 (2020). https://doi.org/10.1016/j.knosys.2020.106148

[212] Petrillo, A., De Felice, F., Ortiz Barrios, M., Parra Negrete, K., Romero, B., Arenas, A.: An ahp-topsis integrated model for selecting the most appropriate tomography equipment. International Journal of Information Technology Decision Making **15** (06 2016). https://doi.org/10.1142/S021962201640006X

[213] Popper, S.W., Lempert, R.J., Bankes, S.C.: Shaping the future. Scientific American **292**(4), 66–71 (2005)

[214] Powers, D.M.W.: Evaluation: From precision, recall and f-measure to roc, informedness, markedness and correlation. Journal of Machine Learning Technologies **2**(1), 37–63 (2011)

[215] Ratner, A., Bach, S., Ehrenberg, H., Fries, J., Wu, S., Ré, C.: Snorkel: Rapid training data creation with weak supervision. The VLDB Journal **29**(2), 709–730 (2020)

[216] Rechenberg, I.: Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution. Frommann-Holzboog (1973)

[217] Reyes, J., Morales-Esteban, A., Martínez-Álvarez, F.: Neural networks to predict earthquakes in chile. Applied Soft Computing **13**(2), 1314–1328 (2013)

[218] Riquelme, H.: Do consumers know what they want? Journal of Consumer Marketing **18**(5), 437–448 (2001)

[219] Robbins, H., Monro, S.: A stochastic approximation method. The Annals of Mathematical Statistics pp. 400–407 (1951)

[220] Roy, B.: Classement et choix en présence de points de vue multiples. RIRO **2**, 57–75 (1968)

[221] Roy, B.: Electre iii: Un algorithme de classement fondé sur une représentation floue des préférences en présence de critères multiples. Cahiers du CERO **20**(1), 3–24 (1978)

[222] Roy, B.: Multicriteria Methodology for Decision Aiding. Kluwer Academic Publishers, Dordrecht (1996)

[223] Roy, B., Bertier, P.: La méthode electre ii: Une méthode de classement en présence de critères multiples. SEMA (Metra International) (1971), working Paper No. 142

[224] Roy, B., Hugonnard, J.: Ranking of suburban line extension projects on the paris metro system by a multicriteria method. Transportation Research **16A**(4), 301–312 (1982)

[225] Rudolph, F.M.: Diverter decision aiding for in-flight diversions. Tech. Rep. NASA-CR-185642, NASA Ames Research Center (1990), https://ntrs.nasa.gov/citations/19910003364

[226] Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning representations by back-propagating errors. Nature **323**, 533–536 (1986)

[227] Saaty, R.W.: The analytic hierarchy process—what it is and how it is used. Mathematical modelling **9**(3-5), 161–176 (1987)

[228] Saaty, T.L.: The Analytic Hierarchy Process. McGraw-Hill, New York (1980)

[229] Sadhu, T., Lahiri, S.K., Roy, J., Bhattacharjee, A., Chakrabarty, J.: Optimization of frying process for maintaining nutritional quality to satisfy consumers' sensory attributes: A novel application of multi-criteria decision-making approach. Journal of Multi-Criteria Decision Analysis **30**(1-2), 44–61 (2023)

[230] Sahay, R.R., Srivastava, A.: Predicting monsoon floods in rivers embedding wavelet transform, genetic algorithm and neural network. Water resources management **28**, 301–317 (2014)

[231] Sampson, J.R.: Adaptation in natural and artificial systems (john h. holland) (1976)

[232] Sánchez, L., Rodríguez-Fernández, V., Vasile, M.: Robust classification with belief functions and deep learning applied to stm. In: 2024 IEEE Congress on Evolutionary Computation (CEC). pp. 1–8. IEEE (2024)

[233] dos Santos, F.L.M., Tecchio, P., Ardente, F., Pekár, F.: User automotive powertrain-type choice model and analysis using neural networks. Sustainability **13**(2), 1–15 (2021)

[234] Schwalbe, G., Finzel, B.: A comprehensive taxonomy for explainable artificial intelligence: a systematic survey of surveys on methods and concepts. Data Mining and Knowledge Discovery **38**(5), 3043–3101 (2024)

[235] Schwartz, P.: The Art of the Long View: Planning for the Future in an Uncertain World. Doubleday (1996)

[236] Sexton, G.A.: Diverter: An ai-based decision aid for in-flight diversions. Tech. Rep. NASA-TM-102389, NASA Ames Research Center (1989), https://ntrs.nasa.gov/api/citations/19890014945/downloads/19890014945.pdf

[237] Shanteau, J.: How much information does an expert use? is it relevant? Acta psychologica **81**(1), 75–86 (1992)

[238] Shapiro, A., Dentcheva, D., Ruszczynski, A.: Lectures on stochastic programming: modeling and theory. SIAM (2021)

[239] Shapley, L.S.: A value for n-person games. Contributions to the Theory of Games **2**, 307–317 (1953)

[240] Shiva, J.S., Chandler, D.G., Kunkel, K.E.: Mapping heat wave hazard in urban areas: A novel multi-criteria decision making approach. Atmosphere **13**(7), 1037 (2022). https://doi.org/10.3390/atmos13071037

[241] Sirbiladze, G., Ghvaberidze, B., Midodashvili, B., Khutsishvili, I., Matsaberidze, B., Manjafarashvili, T.: New fuzzy madm approach for the temporary logistics hubs' selection preferences identification in disaster region. Bulletin of TICMI **27**(2), 67–80 (2023). https://doi.org/10.56128/ticmi.2023.27.2.67

[242] Slack, D., Hilgard, S., Jia, E., Singh, S., Lakkaraju, H.: Fooling lime and shap: Adversarial attacks on post hoc explanation methods. In: Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society. pp. 180–186 (2020)

[243] Soll, H., Proske, S., Hofinger, G., Steinhardt, G.: Decision-making tools for aeronautical teams: For-dec and beyond. Aviation Psychology and Applied Human Factors (2016)

[244] Soltanmohammadi, H., Osanloo, M., Bazzazi, A.A.: Deriving preference order of post-mining land-uses through mlsa framework: application of an outranking technique. Environmental Geology **58**, 877–888 (2009)

[245] Stević, Ž., Miškić, S., Vojinović, D., Huskanović, E., Stanković, M., Pamučar, D.: Development of a model for evaluating the efficiency of transport companies: Pca–dea–mcdm model. Axioms **11**(3), 140 (2022)

[246] Sun, K., Qian, T., Chen, T., Liang, Y., Nguyen, Q.V.H., Yin, H.: Where to go next: Modeling long-and short-term user preferences for point-of-interest recommendation. In: Proceedings of the AAAI conference on artificial intelligence. vol. 34, pp. 214–221 (2020)

[247] Sun, R., Gong, Z., Gao, G., Shah, A.A.: Comparative analysis of multi-criteria decision-making methods for flood disaster risk in the yangtze river delta. International Journal of Disaster Risk Reduction **51**, 101768 (2020). https://doi.org/10.1016/j.ijdrr.2020.101768

[248] Sundararajan, M., Taly, A., Yan, Q.: Axiomatic attribution for deep networks. arXiv preprint arXiv:1703.01365 (2017)

[249] Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. MIT Press (1998)

[250] Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. MIT Press (2018)

[251] Tax, N., Bockting, S., Hiemstra, D.: A cross-benchmark comparison of 87 learning to rank methods. Information processing & management **51**(6), 757–772 (2015)

[252] Tecle, A., Fogel, M., Duckstein, L.: Multicriterion selection of wastewater management alternatives. Journal of Water Resources Planning and Management **114**(4), 383–398 (1988)

[253] Thurstone, L.L.: A law of comparative judgment. In: Scaling, pp. 81–92. Routledge (2017)

[254] Triantaphyllou, E.: Multi-criteria decision making methods: A comparative study. Springer, Boston, MA (2000)

[255] Turing, A.M.: Computing machinery and intelligence. Mind **59**(236), 433–460 (1950)

[256] Tyagi, M., Kumar, P.: A hybrid approach using ahp-topsis for analyzing e- scm performance. Procedia Engineering **97**, 2195–2203 (06 2014). https://doi.org/10.1016/j.proeng.2014.12.463

[257] Tyagi, M., Kumar, P.: A hybrid approach using ahp-topsis for analyzing e- scm performance. Procedia Engineering **97**, 2195–2203 (06 2014). https://doi.org/10.1016/j.proeng.2014.12.463

[258] Tzeng, G.H., Huang, J.J.: Multiple Attribute Decision Making: Methods and Applications. Chapman and Hall/CRC (2011)

[259] Urbanowicz, R.J., Browne, W.N.: Introduction to learning classifier systems. Springer (2017)

[260] Vahdani, B., Mousavi, S., Tavakkoli-Moghaddam, R., Hashemi, H.: A new design of the elimination and choice translating reality method for multi-criteria group decision-making in an intuitionistic fuzzy environment. Applied Mathematical Modelling **37**(4), 1781–1799 (2013)

[261] Vaidya, O.S., Kumar, S.: Analytic hierarchy process: An overview of applications. European Journal of Operational Research **169**(1), 1–29 (2006)

[262] Van Dijk, E.E., Van Tartwijk, J., Van Der Schaaf, M.F., Kluijtmans, M.: What makes an expert university teacher? a systematic review and synthesis of frameworks for teacher expertise in higher education. Educational Research Review **31**, 100365 (2020)

[263] Van Ittersum, K., Pennings, J.M., Wansink, B., Van Trijp, H.C.: The validity of attribute-importance measurement: A review. Journal of Business Research **60**(11), 1177–1190 (2007)

[264] Van Laarhoven, P., Pedrycz, W.: A fuzzy extension of saaty's priority theory. Fuzzy Sets and Systems **11**(2), 229–241 (1983)

[265] Virtanen, P., Gommers, R., Oliphant, T.E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S.J., Brett, M., Wilson, J., Millman, K.J., Mayorov, N., Nelson, A.R.J., Jones, E., Kern, R., Larson, E., Carey, C.J., Polat, , Feng, Y., Moore, E.W., VanderPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E.A., Harris, C.R., Archibald, A.M., Ribeiro, A.H., Pedregosa, F., van Mulbregt, P., SciPy 1.0 Contributors: SciPy 1.0: Fundamental algorithms for scientific computing in python. Nature Methods **17**, 261–272 (2020). https://doi.org/10.1038/s41592-019-0686-2

[266] Visave, J.: Ai in emergency management: Ethical considerations and challenges. J. Emerg. Manag. Disaster Commun **5**, 165–183 (2024)

[267] Wack, P.: Scenarios: uncharted waters ahead. Harvard business review **63**(5), 72–89 (1985)

[268] Wang, H., Luo, P., Wu, Y.: Research on the location decision-making method of emergency medical facilities based on wsr. Scientific Reports **13**, 18011 (2023). https://doi.org/10.1038/s41598-023-44209-0

[269] Wang, J., Dietz, T., Carpenter, S.R., Alberti, M., Folke, C., Moran, E., Pell, A.N., Deadman, P., Kratz, T., Lubchenco, J.: Environmental management and decision making for sustainable development. Environmental Management **52**(3), 331–340 (2009)

[270] Wang, J., Zhao, Y., Balamurugan, P., Selvaraj, P.: Managerial decision support system using an integrated model of ai and big data analytics. Annals of Operations Research pp. 1–18 (2022)

[271] Wang, X., Zhang, C., Deng, J., Su, C., Gao, Z.: Analysis of factors influencing miners' unsafe behaviors in intelligent mines using a novel hybrid mcdm model. International Journal of Environmental Research and Public Health **19**(12), 7368 (2022). https://doi.org/10.3390/ijerph19127368

[272] Watson, A.A., Kasprzyk, J.R.: Incorporating deeply uncertain factors into the many objective search process. Environmental Modelling & Software **89**, 159–171 (2017)

[273] Weinstein, B.D.: What is an expert? Theoretical medicine **14**(1), 57–73 (1993)

[274] Wikipedia contributors: List of the busiest airports in germany. https://en.wikipedia.org/wiki/List_of_the_busiest_airports_in_Germany, accessed: 2025-07-08

[275] Wilson, S.W.: Classifier fitness based on accuracy. Evolutionary Computation **3**(2), 149–175 (1995). https://doi.org/10.1162/evco.1995.3.2.149

[276] Wilson, S.W.: Get real! xcs with continuous-valued inputs. In: International Workshop on Learning Classifier Systems. pp. 209–219. Springer (1999)

[277] Wilson, S.W.: Classifiers that approximate functions. Natural Computing **1**(2), 211–234 (2002)

[278] Wu, Q., Burges, C.J., Svore, K.M., Gao, J.: Adapting boosting for information retrieval measures. In: Information Retrieval. vol. 13, pp. 254–270 (2010)

[279] Würfel, J., Djartov, B., Papenfuß, A., Wies, M.: Intelligent pilot advisory system: The journey from ideation to an early system design of an ai-based decision support system for airline flight decks. AHFE 2023 (2023)

[280] Würfel, J., Flemisch, F.O.: Human system exploration for the ai-based flight deck decision support system ipas. In: In D. de Waard, D. Manzey, K. Brookhuis, F. Siebert, K. Karrer-Gauß, S. Winkler, F. Di Nocera, A. Toffetti, and T. Franke (2024). Proceedings of the Human Factors and Ergonomics Society Europe Chapter 2024 Annual Conference. ISSN 2333-4959 (online). Avail (2024)

[281] Würfel, J., Papenfuß, A., Wies, M.: Operationalizing ai explainability using interpretability cues in the cockpit: Insights from user-centered development of the intelligent pilot advisory system (ipas). In: International Conference on Human-Computer Interaction. pp. 297–315. Springer (2024)

[282] xcsf-dev: Xcsf learning classifier system (python implementation). `https://github.com/xcsf-dev/xcsf` (2025), accessed: 2025-08-29

[283] Xu, L., Wang, J., Ou, Y., Fu, Y., Bian, X.: A novel decision-making system for selecting offshore wind turbines with pca and d numbers. Energy **258**, 124818 (2022)

[284] Xu, W., Li, W., Proverbs, D., Chen, W.: An evaluation of the humanitarian supply chains in the event of flash flooding. Water **15**(18), 3323 (2023). `https://doi.org/10.3390/w15183323`

[285] Ye, A., Zhang, R., Wu, P., Xing, Y.: q-rung orthopair fuzzy topsis method and the application to information service quality evaluation in online health community. Journal of Intelligent & Fuzzy Systems **41**(2), 3697–3714 (2021)

[286] Zagorecki, A.T., Johnson, D.E., Ristvej, J.: Data mining and machine learning in the context of disaster and crisis management. International Journal of Emergency Management **9**(4), 351–365 (2013)

[287] Zambrano, L.A., Jadan, B.V.: Integrative multi-information fusion for enhanced risk assessment: A multi-criteria decision-making framework. Fusion Practice and Applications **14** (2024). `https://doi.org/10.54216/FPA.140104`

[288] Zeleny, M.: Multiple Criteria Decision Making. McGraw-Hill, New York (1982)

[289] Zhan, J., Sun, B., Zhang, X.: Pf-topsis method based on cpfrs models: An application to unconventional emergency events. Computers & Industrial Engineering **139**, 106192 (2020). `https://doi.org/10.1016/j.cie.2019.106192`

[290] Zhang, Q., Li, H.: Moea/d: A multiobjective evolutionary algorithm based on decomposition. IEEE Transactions on Evolutionary Computation **11**(6), 712–731 (2007)

[291] Zhang, W., Liu, X., Yu, W., Cui, C., Zheng, A.: Spatial-temporal sensitivity analysis of flood control capability in china based on madm-gis model. Entropy **24**(6), 772 (2022). `https://doi.org/10.3390/e24060772`

[292] Zhao, Z., Lu, H., Cai, D., He, X., Zhuang, Y.: User preference learning for online social recommendation. IEEE Transactions on Knowledge and Data Engineering **28**(9), 2522–2534 (2016)

[293] Zhen, Y., Song, Y., Yeung, D.Y.: Semiparametric preference learning. Tsinghua Science and Technology **19**(3), 257–264 (2014)

[294] Zhihong, T., Shirui, P., Xianyong, Z.: Research on the construction of smart city emergency management system under digital twin technology: Taking the practice of new coronary pneumonia joint prevention and control as an example. In: 2020 4th International Seminar on Education, Management and Social Sciences (ISEMSS 2020). pp. 146–151. Atlantis Press (2020)

[295] Zhong, K., Yang, Z., Xiao, G., Li, X., Yang, W., Li, K.: An efficient parallel reinforcement learning approach to cross-layer defense mechanism in industrial control systems. IEEE Transactions on Parallel and Distributed Systems **33**(11), 2979–2990 (2021)

[296] Zhou, Z.H.: A brief introduction to weakly supervised learning. National science review **5**(1), 44–53 (2018)

[297] Zitzler, E., Laumanns, M., Thiele, L.: Spea2: Improving the strength pareto evolutionary algorithm. TIK-report **103**, 1–21 (2001)

[298] Špák, M., Kalaš, M., Hromádko, J.: Enhancement of the diversion airport selection methodology. Transportation Research Procedia **51**, 373–380 (2020). https://doi.org/10.1016/j.trpro.2020.11.041, https://www.sciencedirect.com/science/article/pii/S2352146520308826

List of Abbreviations

ADM	Aeronautical Decision Making.
AHP	The Analytic Hierarchy Process.
AI	Artificial Intelligence.
AI-MCDM	Artificial Intelligence–Multi-Criteria Decision-Making.
AICIS	AI-Crew Interaction System.
AICOM	AI Core Module.
ATc	Air Traffic Control.
BADA	Base of Aircraft Data.
CRM	Crew Resource Management.
DA	Decision Analysis.
DAAS	Dynamic Alternate Airport Selection.
DECIDE	Detect, Estimate, Choose, Identify, Do, Evaluate.
DLR	German Aerospace Center (Deutsches Zentrum für Luft- und Raumfahrt).
DSS	Decision-Support System.
EA	Evolutionary Algorithms.
EASA	European Union Aviation Safety Agency.
ELECTRE	ELimination Et Choix Traduisant la REalité.
EMO	Evolutionary Multi-Objective Optimization.
ExACT-MCDM	Explainable, Adaptive, Context-aware, Time-critical Multi-Criteria Decision-Making.
FOR-DEC	Facts, Options, Risks & Benefits, Decision, Execution, Check.
IPAS	Intelligent Pilot Advisory System.
LCS	Learning Classifier Systems.
MADM	Multi-Attribute Decision Making.
MCDA	Multi-Criteria Decision Analysis.

MCDM	Multi-Criteria Decision Making.
MDP	Markov Decision Process.
ML	Machine Learning.
MODM	Multi-Objective Decision Making.
MOEA	Multi-Objective Evolutionary Algorithms.
MOMP	Multi-Objective Multiplexer Problem.
MOO	Multi-Objective Optimization.
MORDM	Many-Objective Robust Decision Making.
MORO	Many-Objective Robust Optimization.
NN	Neural Network.
PCA	Principal Component Analysis.
PFI	Permutation Feature Importance.
PIOSEE	Problem, Information, Options, Select, Execute, Evaluate.
RDM	Robust Decsion-making.
RL	Reinforcement Learning.
T-DODAR	Time, Diagnose, Options, Decide, Assign, Review.
TOPSIS	Technique for Order Preference by Similarity to Ideal Solution.
WSM	Weighted Sum Method.
XAI	Explainable Artificial Intelligence.

Author's Publications

[77] Djartov, B., Mostaghim, S., Papenfuß, A., Wies, M.: Description and first evaluation of an approach for a pilot decision support system based on multi-attribute decision making. In: 2022 IEEE Symposium Series on Computational Intelligence (SSCI). pp. 141–147. IEEE (2022)

[279] Würfel, J., Djartov, B., Papenfuß, A., Wies, M.: Intelligent pilot advisory system: The journey from ideation to an early system design of an ai-based decision support system for airline flight decks. AHFE 2023 (2023)

[76] Djartov, B., Mostaghim, S.: Multi-objective multiplexer decision making benchmark problem. In: Proceedings of the Companion Conference on Genetic and Evolutionary Computation. pp. 1676–1683 (2023)

[80] Djartov, B., Papenfuß, A., Wies, M.: Wings of wisdom: Learning from pilot decision data with interpretable ai models. In: International Conference on Human-Computer Interaction. pp. 241–256. Springer (2024)

[79] Djartov, B., Oswald, F.M., Jakobi, J.: Through the psychological lens: Unveiling biases in multi-criteria decision-making. In: International Conference on Human Interaction & Emerging Technologies IHIET (2024)

[78] Djartov, B., Mostaghim, S., Papenfuß, A., Wies, M.: A learning classifier system approach to time-critical decision-making in dynamic alternate airport selection. In: 2024 IEEE Congress on Evolutionary Computation (CEC). pp. 1–8. IEEE (2024)

[81] Djartov, B., Würfel, J.: Navigating Decisions in the Cockpit: The Intelligent Pilot Advisory System, pp. 305–325. Springer Nature Switzerland, Cham (2025). doi: 10.1007/978-3-031-83512-4_18, https://doi.org/10.1007/978-3-031-83512-4_18

List of Figures

List of Tables

Appendices

A Scenario types

Table A.1: Complete matrix of scenario types combining different weather conditions and technical failure severities, resulting in 60 different combinations (10 weather types × 6 failure types). Weather distributions show the probability of sampling from different intensity categories. For each available airport option, weather parameters are sampled independently using uniform distributions within the specified ranges.

Scenario Weather	**Failure Severity**	**Weather Distribution**	**Detailed Conditions (sampled per airport option)**
Very calm	Very light	45% clear, 30% light, 25% medium	Each available airport option samples weather parameters independently from this distribution. Clear conditions (45%): visibility uniformly sampled from 1600–2000m, cloud ceiling from 500–1000ft, dry runways, wind speeds follow wind field or multiplied by 3.5. Light conditions (30%): includes fog (visibility 75–500m) or light precipitation (rain/snow/thunderstorms) with visibility 1000–2000m, ceiling 200–800ft, wet or wet snow runways. Medium intensity (25%): visibility 700–1600m, ceiling 90–500ft, standing water or compacted snow on runways, winds up to 35–45kt. Technical problems: Single minor issue (70% minor fuel leak, 30% minor general warning). 15% probability of additional medical emergency.
Very calm	Light	45% clear, 30% light, 25% medium	Weather sampled independently per airport option. Predominantly clear scenarios (45%): dry runways, visibility 1600–2000m, good ceilings 500–1000ft. Light precipitation (30%): wet or wet snow runways, visibility 1000–2000m, ceiling 200–800ft. Occasional medium intensity (25%): standing water or compacted snow, visibility 700–1600m, ceiling 90–500ft. Wind speeds range from wind field values to uniformly sampled 35–45kt depending on scenario intensity. Technical problems: Single minor issue (20% minor general warning, 40% minor fuel leak, 40% moderate fuel leak). 15% probability of additional medical emergency.

Continued on next page

Table A.1 – continued from previous page

Scenario Weather	Failure Severity	Weather Distribution	Detailed Conditions (sampled per airport option)
Very calm	Moderate	45% clear, 30% light, 25% medium	Each airport independently samples from distribution with 45% probability of clear conditions. Clear (45%): dry runways, visibility 1600–2000m, ceiling 500–1000ft. Light scenarios (30%): wet or wet snow runways, visibility 1000–2000m, ceiling 200–800ft. Medium scenarios (25%): standing water or compacted snow, visibility 700–1600m, ceiling 90–500ft. Wind speeds vary from wind field patterns to 35–45kt uniformly sampled. Technical problems: Multiple minor issues (30% minor fuel leak, 50% moderate fuel leak, 20% major general warning). Combinations may include minor fuel leak with general warnings or major general warning with minor general warning. 20% probability of additional medical emergency.
Very calm	Severe	45% clear, 30% light, 25% medium	Independent weather sampling per available airport option. Clear conditions (45%): dry runways with excellent visibility 1600–2000m. Light precipitation (30%): wet or wet snow runways, visibility 1000–2000m. Medium intensity (25%): contaminated runways, visibility 700–1600m. Wind speeds vary from wind field values to 35–45kt. Technical problems: Severe issues (40% moderate fuel leak, 30% critical fuel leak, 30% engine out). May include combinations like engine out with fuel leaks or critical fuel leak with general warnings. 25% probability of additional medical emergency.
Very calm	Very severe	45% clear, 30% light, 25% medium	Weather distribution favors clear to light conditions with independent sampling per airport. Visibility ranges from 700–2000m depending on sampled intensity category. Cloud ceilings span 90–1000ft. Runway conditions range from dry (clear scenarios) to standing water or compacted snow (medium scenarios). Winds typically follow wind field patterns, multiply by 3.5, or reach 35–45kt uniformly sampled. Technical problems: Very severe issues (50% critical fuel leak, 50% engine out). May occur individually or in combination. 30% probability of additional medical emergency.

Continued on next page

Table A.1 – continued from previous page

Scenario Weather	Failure Severity	Weather Distribution	Detailed Conditions (sampled per airport option)
Very calm	Critical	45% clear, 30% light, 25% medium	Favorable weather distribution with each airport sampling conditions independently. 45% probability of dry runways with visibility 1600–2000m. 30% wet conditions with visibility 1000–2000m and ceiling 200–800ft. 25% medium contamination with visibility 700–1600m and standing water or compacted snow. Wind speeds from wind field to 35–45kt. Technical problems: Critical failures (60% critical fuel leak, 40% engine out). Often includes combinations of critical fuel leak with engine out or general warnings. 40% probability of additional medical emergency.
Calm	Very light	20% clear, 40% light, 40% medium	Balanced distribution with independent sampling per airport option. Clear scenarios (20%): dry runways, visibility 1600–2000m, ceiling 500–1000ft. Light conditions (40%): wet or wet snow runways, visibility 1000–2000m, ceiling 200–800ft, fog conditions possible. Medium intensity (40%): standing water or compacted snow, visibility 700–1600m, ceiling 90–500ft, winds up to 35–45kt. Technical problems: Single minor issue (70% minor fuel leak, 30% minor general warning). 15% probability of additional medical emergency.
Calm	Light	20% clear, 40% light, 40% medium	Each option independently samples with balanced light-medium distribution. Visibility ranges from 700–2000m depending on category. Cloud ceiling spans 90–1000ft. Runway conditions: dry (20%), wet or wet snow (40%), or standing water/compacted snow (40%). Wind speeds vary from wind field values, multiplied by 3.5, or up to 35–45kt uniformly sampled. Technical problems: Single minor issue (20% minor general warning, 40% minor fuel leak, 40% moderate fuel leak). 15% probability of additional medical emergency.
Calm	Moderate	20% clear, 40% light, 40% medium	Per-airport weather sampling with 20% probability of clear conditions. Clear (20%): dry runways, visibility 1600–2000m. Light precipitation (40%): wet or wet snow, visibility 1000–2000m. Medium intensity (40%): standing water or compacted snow, visibility 700–1600m, ceiling 90–500ft. Winds vary from wind field patterns to 35–45kt. Technical problems: Multiple minor issues (30% minor fuel leak, 50% moderate fuel leak, 20% major general warning) in combinations such as minor fuel leak with general warnings, or both general warning levels together. 20% probability of additional medical emergency.

Continued on next page

Table A.1 – continued from previous page

Scenario Weather	Failure Severity	Weather Distribution	Detailed Conditions (sampled per airport option)
Calm	Severe	20% clear, 40% light, 40% medium	Weather sampled independently per option, predominantly light to medium conditions. Visibility uniformly sampled from 700–2000m. Ceilings range 90–1000ft. Runways: dry (20%), wet or wet snow (40%), contaminated (40%). Wind speeds from wind field or up to 35–45kt. Technical problems: Severe issues (40% moderate fuel leak, 30% critical fuel leak, 30% engine out). Combinations include engine out with minor or moderate fuel leaks, or critical fuel leak with general warnings. 25% probability of additional medical emergency.
Calm	Very severe	20% clear, 40% light, 40% medium	Each airport independently samples from balanced distribution. 20% clear, 40% light, 40% medium intensity scenarios. Visibility ranges 700–2000m, ceilings 90–1000ft. Runway conditions vary from dry to standing water or compacted snow. Winds from wind field values to 35–45kt. Technical problems: Very severe issues (50% critical fuel leak, 50% engine out). May occur individually or in combinations requiring immediate emergency coordination. 30% probability of additional medical emergency.
Calm	Critical	20% clear, 40% light, 40% medium	Per-option weather sampling predominantly light-medium intensity. Visibility 700–2000m, ceiling 90–1000ft. Runway contamination likely (80% wet or worse conditions). Wind speeds range from wind field to 35–45kt. Each available airport has independently sampled weather parameters. Technical problems: Critical failures (60% critical fuel leak, 40% engine out). Often simultaneous critical fuel leak and engine out requiring immediate emergency response. 40% probability of additional medical emergency.
Mild	Very light	20% clear, 30% light, 45% medium, 5% heavy	Medium conditions dominant with independent sampling per airport. Clear/light scenarios: visibility 700–2000m. Medium/heavy scenarios: visibility 500–1600m. Cloud ceiling 90–800ft across categories. Runway conditions: dry (20%), wet or wet snow (30%), standing water or compacted snow (50%). Winds from wind field to 35–60kt, with sporadic directional shifts of 45–90° possible in extreme cases. Technical problems: Single minor issue (70% minor fuel leak, 30% minor general warning). 15% probability of additional medical emergency.

Continued on next page

Table A.1 – continued from previous page

Scenario Weather	Failure Severity	Weather Distribution	Detailed Conditions (sampled per airport option)
Mild	Light	20% clear, 30% light, 45% medium, 5% heavy	Per-option sampling favors medium intensity at 45%. Visibility ranges 500–2000m depending on sampled category. Ceiling 80–1000ft. Runway conditions predominantly wet to standing water or compacted snow. Wind speeds vary from wind field, multiplied by 3.5, or 35–60kt with possible sporadic directional shifts. Technical problems: Single minor issue (20% minor general warning, 40% minor fuel leak, 40% moderate fuel leak). 15% probability of additional medical emergency.
Mild	Moderate	20% clear, 30% light, 45% medium, 5% heavy	Each airport independently samples with medium scenarios most likely (45%). Medium scenarios: visibility 700–1600m, standing water or compacted snow. Light conditions (30%): visibility 1000–2000m, wet or wet snow runways. Clear (20%): dry runways. Heavy (5%): visibility 400–1000m, standing water or compacted snow, winds 35–60kt with sporadic shifts. Technical problems: Multiple minor issues (30% minor fuel leak, 50% moderate fuel leak, 20% major general warning) in various combinations. 20% probability of additional medical emergency.
Mild	Severe	20% clear, 30% light, 45% medium, 5% heavy	Weather distribution per airport option: predominantly medium with occasional heavy. Visibility 400–2000m depending on intensity. Ceiling 80–1000ft. Runways range from dry (20%) to severely contaminated (5% heavy scenarios). Winds from wind field to 35–60kt. Sporadic shifts of 45–90° occur in 5% of sampled scenarios. Technical problems: Severe issues (40% moderate fuel leak, 30% critical fuel leak, 30% engine out). Includes combinations like engine out with fuel leaks of varying severity. 25% probability of additional medical emergency.
Mild	Very severe	20% clear, 30% light, 45% medium, 5% heavy	Independent per-option sampling with medium intensity prevalent. 45% medium scenarios: visibility 700–1600m, standing water or compacted snow, winds up to 35–45kt. 30% light conditions, 20% clear scenarios. Occasional heavy conditions (5%): winds to 35–60kt with sporadic shifts. Technical problems: Very severe issues (50% critical fuel leak, 50% engine out). Combinations may include both critical fuel leak and engine out simultaneously. 30% probability of additional medical emergency.

Continued on next page

Table A.1 – continued from previous page

Scenario Weather	Failure Severity	Weather Distribution	Detailed Conditions (sampled per airport option)
Mild	Critical	20% clear, 30% light, 45% medium, 5% heavy	Each airport samples from distribution independently. Visibility ranges 400–2000m. Ceiling 80–1000ft. Runway conditions vary from dry (20%) to standing water or compacted snow (75%). Wind speeds from wind field to 35–60kt with sporadic directional shifts possible. Technical problems: Critical failures (60% critical fuel leak, 40% engine out). Often multiple severe problems simultaneously including critical fuel leak with engine out and potential general warnings. 40% probability of additional medical emergency.
Light	Very light	10% clear, 20% light, 40% medium, 15% heavy, 15% extreme	Medium intensity prevalent at 40% with independent per-airport sampling. Visibility 400–2000m depending on category. Ceiling 70–1000ft. Runways: dry (10%), wet or wet snow (20%), standing water or compacted snow (70%). Winds from wind field to 40–60kt. Extreme scenarios (15%) include sporadic shifts of 45–90°. Technical problems: Single minor issue (70% minor fuel leak, 30% minor general warning). 15% probability of additional medical emergency.
Light	Light	10% clear, 20% light, 40% medium, 15% heavy, 15% extreme	Each option samples independently with varied intensity distribution. 40% medium scenarios: visibility 700–1600m, standing water or compacted snow. 30% heavy/extreme: winds 35–60kt, visibility 400–1000m, sporadic shifts possible. 20% light: wet or wet snow runways. 10% clear: dry conditions. Wind speeds vary from wind field to extreme values with sporadic directional changes. Technical problems: Single minor issue (20% minor general warning, 40% minor fuel leak, 40% moderate fuel leak). 15% probability of additional medical emergency.
Light	Moderate	10% clear, 20% light, 40% medium, 15% heavy, 15% extreme	Per-airport weather sampling: predominantly medium to extreme conditions. Visibility 400–2000m. Ceiling 70–1000ft. Runway contamination common (90% wet or worse). Winds from wind field to 40–60kt. 15% probability per airport of sporadic directional shifts of 45–90°. Technical problems: Multiple minor issues (30% minor fuel leak, 50% moderate fuel leak, 20% major general warning) occurring in combinations. 20% probability of additional medical emergency.

Continued on next page

Table A.1 – continued from previous page

Scenario Weather	Failure Severity	Weather Distribution	Detailed Conditions (sampled per airport option)
Light	Severe	10% clear, 20% light, 40% medium, 15% heavy, 15% extreme	Each airport independently samples with challenging distribution. 40% medium intensity scenarios. 30% heavy/extreme: winds to 40–60kt with sporadic shifts, visibility 400–1000m. Visibility ranges 400–2000m overall. Ceilings 70–1000ft. Runways predominantly contaminated. Technical problems: Severe issues (40% moderate fuel leak, 30% critical fuel leak, 30% engine out). Combinations include engine out with fuel leaks or critical fuel leak with general warnings and potential additional issues. 25% probability of additional medical emergency.
Light	Very severe	10% clear, 20% light, 40% medium, 15% heavy, 15% extreme	Weather distribution per option: 40% medium, 15% heavy, 15% extreme. Visibility 400–2000m. Ceiling 70–1000ft. Runways with standing water or compacted snow (70% of scenarios). Wind speeds from wind field to 40–60kt. Sporadic shifts of 45–90° in extreme cases. Technical problems: Very severe issues (50% critical fuel leak, 50% engine out). Often both problems simultaneously affecting aircraft capability. 30% probability of additional medical emergency.
Light	Critical	10% clear, 20% light, 40% medium, 15% heavy, 15% extreme	Independent sampling per airport with varied intensity distribution. Visibility 400–2000m. Ceiling 70–1000ft. Runway conditions predominantly contaminated (90%). Winds range from wind field values to 40–60kt. 15% probability of sporadic shifts per sampled airport. Each option has independently sampled weather conditions. Technical problems: Critical failures (60% critical fuel leak, 40% engine out). Typically includes simultaneous critical fuel leak and engine out, potentially with additional general warnings. 40% probability of additional medical emergency.
Moderate	Very light	5% clear, 15% light, 30% medium, 20% heavy, 30% extreme	Extreme conditions prevalent at 30% with independent per-airport sampling. Visibility 400–2000m depending on intensity. Ceiling 70–1000ft. Runways: dry (5%), wet or wet snow (15%), standing water or compacted snow (80%). Winds from wind field to 40–60kt. 30% probability of sporadic shifts of 45–90° per airport. Technical problems: Single minor issue (70% minor fuel leak, 30% minor general warning). 15% probability of additional medical emergency.

Continued on next page

Table A.1 – continued from previous page

Scenario Weather	Failure Severity	Weather Distribution	Detailed Conditions (sampled per airport option)
Moderate	Light	5% clear, 15% light, 30% medium, 20% heavy, 30% extreme	Per-option sampling with balanced medium, heavy, and extreme conditions. 30% medium scenarios. 20% heavy conditions. 30% extreme scenarios. Visibility 400–2000m. Ceiling 70–1000ft. Runway contamination common (80%). Wind speeds highly variable: wind field to 40–60kt, sporadic directional shifts likely. Technical problems: Single minor issue (20% minor general warning, 40% minor fuel leak, 40% moderate fuel leak). 15% probability of additional medical emergency.
Moderate	Moderate	5% clear, 15% light, 30% medium, 20% heavy, 30% extreme	Each airport independently samples with challenging distribution. 50% heavy/extreme scenarios: winds 35–60kt, sporadic shifts of 45–90°, visibility 400–1000m. 30% medium, 20% light/clear scenarios. Severe runway contamination prevalent. Ceiling 70–1000ft. Technical problems: Multiple minor issues (30% minor fuel leak, 50% moderate fuel leak, 20% major general warning) in various combinations affecting multiple systems. 20% probability of additional medical emergency.
Moderate	Severe	5% clear, 15% light, 30% medium, 20% heavy, 30% extreme	Weather distribution per option: 30% extreme scenarios. Extreme category: winds to 40–60kt with sporadic shifts, visibility as low as 400–900m. Additionally 30% medium, 20% heavy scenarios. Visibility 400–2000m overall. Ceiling 70–1000ft. Runways predominantly standing water or compacted snow (80%). Technical problems: Severe issues (40% moderate fuel leak, 30% critical fuel leak, 30% engine out). Combinations include engine out with fuel leaks, critical fuel leak with general warnings, or multiple simultaneous failures. 25% probability of additional medical emergency.
Moderate	Very severe	5% clear, 15% light, 30% medium, 20% heavy, 30% extreme	Independent sampling per airport with 50% heavy/extreme conditions dominant. Visibility 400–2000m. Ceiling 70–1000ft. Severe runway contamination (80% contaminated scenarios). Winds from wind field to 40–60kt. 30% probability of sporadic shifts per airport. Technical problems: Very severe issues (50% critical fuel leak, 50% engine out). Often simultaneous critical fuel leak and engine out requiring immediate emergency decision-making. 30% probability of additional medical emergency.

Continued on next page

Table A.1 – continued from previous page

Scenario Weather	Failure Severity	Weather Distribution	Detailed Conditions (sampled per airport option)
Moderate	Critical	5% clear, 15% light, 30% medium, 20% heavy, 30% extreme	Each airport samples from challenging distribution. 30% extreme scenarios: sporadic wind shifts of 45–90°, winds 35–60kt, visibility 400–900m. Additionally 30% medium, 20% heavy, 15% light, rare clear (5%). Runway contamination throughout. Technical problems: Critical failures (60% critical fuel leak, 40% engine out). Typically simultaneous critical fuel leak with engine out, often combined with general warnings creating complex emergency scenarios. 40% probability of additional medical emergency.
Challenging	Very light	10% light, 20% medium, 25% heavy, 45% extreme	Extreme conditions dominant at 45% with independent per-airport sampling. Visibility 400–1800m. Ceiling 70–800ft. Runways: wet or wet snow (10%), standing water or compacted snow (90%). Winds from wind field to 40–60kt. 45% probability of sporadic shifts of 45–90° per option. Technical problems: Single minor issue (70% minor fuel leak, 30% minor general warning). 15% probability of additional medical emergency.
Challenging	Light	10% light, 20% medium, 25% heavy, 45% extreme	Each option independently samples with extreme scenarios most likely. 45% extreme: winds 35–60kt with sporadic shifts, visibility 400–900m. 25% heavy scenarios. 20% medium, 10% light conditions. Visibility 400–1800m overall. Ceiling 70–800ft. Severe runway contamination throughout (90%). Technical problems: Single minor issue (20% minor general warning, 40% minor fuel leak, 40% moderate fuel leak). 15% probability of additional medical emergency.
Challenging	Moderate	10% light, 20% medium, 25% heavy, 45% extreme	Weather distribution per airport: 70% heavy/extreme scenarios. Visibility 400–1800m. Ceiling 70–800ft. Runways with standing water or compacted snow (90%). Wind speeds from wind field to 40–60kt. Frequent sporadic directional shifts (45% probability). Technical problems: Multiple minor issues (30% minor fuel leak, 50% moderate fuel leak, 20% major general warning) in combinations affecting multiple aircraft systems. 20% probability of additional medical emergency.

Continued on next page

Table A.1 – continued from previous page

Scenario Weather	Failure Severity	Weather Distribution	Detailed Conditions (sampled per airport option)
Challenging	Severe	10% light, 20% medium, 25% heavy, 45% extreme	Independent sampling per option with extreme conditions prevalent. 45% extreme scenarios: winds to 40–60kt, sporadic shifts of 45–90°, visibility 400–900m. Visibility ranges 400–1800m. Ceilings 70–800ft. Runway contamination severe throughout. Technical problems: Severe issues (40% moderate fuel leak, 30% critical fuel leak, 30% engine out). Combinations include engine out with various fuel leak severities, or critical fuel leak with multiple general warnings. 25% probability of additional medical emergency.
Challenging	Very severe	10% light, 20% medium, 25% heavy, 45% extreme	Each airport samples with extreme scenarios at 45%. Extreme category includes sporadic wind shifts, winds 35–60kt, visibility as low as 400–900m. Overall visibility 400–1800m. Ceiling 70–800ft. Standing water or compacted snow (90%). Technical problems: Very severe issues (50% critical fuel leak, 50% engine out). Often both critical fuel leak and engine out occurring simultaneously creating severe operational constraints. 30% probability of additional medical emergency.
Challenging	Critical	10% light, 20% medium, 25% heavy, 45% extreme	Per-airport weather sampling: predominantly heavy/extreme at 70%. Visibility 400–1800m. Ceiling 70–800ft. Runways severely contaminated throughout. Winds from wind field to 40–60kt. 45% probability of sporadic shifts per option. Technical problems: Critical failures (60% critical fuel leak, 40% engine out). Typically simultaneous critical fuel leak and engine out, often with additional general warnings requiring maximum emergency response. 40% probability of additional medical emergency.
Adverse	Very light	15% medium, 25% heavy, 60% extreme	Extreme conditions dominant at 60% with independent per-airport sampling. Visibility 400–1600m. Ceiling 70–500ft. Runways with standing water or compacted snow (100%). Winds from wind field to 40–60kt. 60% probability of sporadic directional shifts of 45–90° per airport. Technical problems: Single minor issue (70% minor fuel leak, 30% minor general warning). 15% probability of additional medical emergency.

Continued on next page

Table A.1 – continued from previous page

Scenario Weather	Failure Severity	Weather Distribution	Detailed Conditions (sampled per airport option)
Adverse	Light	15% medium, 25% heavy, 60% extreme	Per-option sampling heavily weighted toward extreme scenarios. 60% extreme: winds 35–60kt with sporadic shifts, visibility 400–900m. 25% heavy scenarios. 15% medium conditions. Visibility 400–1600m overall. Ceiling 70–500ft. Severe contamination throughout all options. High winds with frequent directional changes. Technical problems: Single minor issue (20% minor general warning, 40% minor fuel leak, 40% moderate fuel leak). 15% probability of additional medical emergency.
Adverse	Moderate	15% medium, 25% heavy, 60% extreme	Each airport samples from severe distribution: 85% heavy/extreme scenarios. Visibility 400–1600m. Ceiling 70–500ft. Runways with standing water or compacted snow (100%). Wind speeds from wind field to 40–60kt. Sporadic shifts of 45–90° highly likely (60% probability). Technical problems: Multiple minor issues (30% minor fuel leak, 50% moderate fuel leak, 20% major general warning) in combinations. 20% probability of additional medical emergency.
Adverse	Severe	15% medium, 25% heavy, 60% extreme	Weather distribution per option: predominantly extreme at 60%. Extreme scenarios: winds to 40–60kt, sporadic shifts, visibility 400–900m. Visibility 400–1600m overall. Ceiling 70–500ft. All runways with standing water or compacted snow. Technical problems: Severe issues (40% moderate fuel leak, 30% critical fuel leak, 30% engine out). Combinations include engine out with fuel leaks, critical fuel leak with general warnings, requiring emergency procedures. 25% probability of additional medical emergency.
Adverse	Very severe	15% medium, 25% heavy, 60% extreme	Independent sampling per airport with extreme scenarios at 60%. Visibility as low as 400–900m in extreme cases. Winds to 40–60kt with sporadic shifts. Overall visibility 400–1600m. Ceiling 70–500ft. Severe contamination (100%). Technical problems: Very severe issues (50% critical fuel leak, 50% engine out). Often simultaneous critical fuel leak and engine out requiring immediate coordinated emergency response. 30% probability of additional medical emergency.

Continued on next page

Table A.1 – continued from previous page

Scenario Weather	Failure Severity	Weather Distribution	Detailed Conditions (sampled per airport option)
Adverse	Critical	15% medium, 25% heavy, 60% extreme	Each airport samples: predominantly extreme conditions at 60%. Sporadic wind shifts of 45–90° highly likely. Winds 35–60kt. Visibility 400–900m in extreme scenarios. All runways with standing water or compacted snow. Ceiling 70–500ft. Technical problems: Critical failures (60% critical fuel leak, 40% engine out). Typically simultaneous critical fuel leak and engine out, often with multiple general warnings creating maximum emergency scenario. 40% probability of additional medical emergency.
Severe	Very light	5% medium, 20% heavy, 75% extreme	Extreme conditions dominant at 75% with independent per-airport sampling. Visibility 400–1600m. Ceiling 70–500ft. Runways with standing water or compacted snow (100%). Winds consistently 35–60kt. 75% probability of sporadic shifts of 45–90° per airport. Technical problems: Single minor issue (70% minor fuel leak, 30% minor general warning). 15% probability of additional medical emergency.
Severe	Light	5% medium, 20% heavy, 75% extreme	Each option independently samples with extreme scenarios most likely. 75% extreme: winds 35–60kt with sporadic shifts, visibility 400–900m. 20% heavy scenarios. 5% medium conditions. Visibility 400–1600m overall. Ceiling 70–500ft. Severe contamination throughout all options. Sporadic wind shifts highly frequent. Technical problems: Single minor issue (20% minor general warning, 40% minor fuel leak, 40% moderate fuel leak). 15% probability of additional medical emergency.
Severe	Moderate	5% medium, 20% heavy, 75% extreme	Weather distribution per airport: 95% heavy/extreme conditions. Visibility 400–1600m. Ceiling 70–500ft. All runways with standing water or compacted snow. Wind speeds 35–60kt. Sporadic directional shifts of 45–90° expected in 75% of cases. Technical problems: Multiple minor issues (30% minor fuel leak, 50% moderate fuel leak, 20% major general warning) occurring in combinations. 20% probability of additional medical emergency.

Continued on next page

Table A.1 – continued from previous page

Scenario Weather	Failure Severity	Weather Distribution	Detailed Conditions (sampled per airport option)
Severe	Severe	5% medium, 20% heavy, 75% extreme	Independent sampling per option: predominantly extreme at 75%. Extreme scenarios: winds 35–60kt, sporadic shifts, visibility 400–900m. Visibility 400–1600m overall. Ceiling 70–500ft. All runways severely contaminated. Technical problems: Severe issues (40% moderate fuel leak, 30% critical fuel leak, 30% engine out). Combinations include engine out with various fuel leak severities or critical fuel leak with general warnings. 25% probability of additional medical emergency.
Severe	Very severe	5% medium, 20% heavy, 75% extreme	Each airport samples with 75% extreme conditions. Visibility as low as 400–900m in extreme scenarios. Winds 35–60kt with sporadic shifts of 45–90°. Visibility 400–1600m overall. Ceiling 70–500ft. Standing water or compacted snow (100%). Technical problems: Very severe issues (50% critical fuel leak, 50% engine out). Often both problems occurring simultaneously in severe weather environment. 30% probability of additional medical emergency.
Severe	Critical	5% medium, 20% heavy, 75% extreme	Per-airport weather sampling: extreme scenarios dominant at 75%. Sporadic wind shifts highly likely. Winds 35–60kt. Visibility 400–900m in extreme cases. All runways contaminated. Ceiling 70–500ft. Technical problems: Critical failures (60% critical fuel leak, 40% engine out). Typically simultaneous critical fuel leak and engine out, often with additional warnings requiring immediate emergency landing. 40% probability of additional medical emergency.
Extreme	Very light	15% heavy, 85% extreme	Almost exclusively extreme conditions at 85% with independent per-airport sampling. Visibility 400–1400m. Ceiling 70–500ft. Runways with standing water or compacted snow (100%). Winds 35–60kt. 85% probability of sporadic directional shifts of 45–90° per airport. Technical problems: Single minor issue (70% minor fuel leak, 30% minor general warning). 15% probability of additional medical emergency.

Continued on next page

Table A.1 – continued from previous page

Scenario Weather	Failure Severity	Weather Distribution	Detailed Conditions (sampled per airport option)
Extreme	Light	15% heavy, 85% extreme	Per-option sampling: 85% extreme scenarios most prevalent. Extreme category: winds 35–60kt with sporadic shifts, visibility 400–900m. 15% heavy scenarios. Visibility 400–1400m overall. Ceiling 70–500ft. Severe contamination throughout all options. Sporadic wind shifts standard. Technical problems: Single minor issue (20% minor general warning, 40% minor fuel leak, 40% moderate fuel leak). 15% probability of additional medical emergency.
Extreme	Moderate	15% heavy, 85% extreme	Each airport samples from distribution: predominantly extreme at 85%. Visibility 400–1400m. Ceiling 70–500ft. All runways with standing water or compacted snow. Wind speeds 35–60kt. Sporadic directional shifts of 45–90° expected (85% probability). Technical problems: Multiple minor issues (30% minor fuel leak, 50% moderate fuel leak, 20% major general warning) in various combinations. 20% probability of additional medical emergency.
Extreme	Severe	15% heavy, 85% extreme	Weather distribution per option: almost exclusively extreme at 85%. Extreme scenarios: winds 35–60kt, sporadic shifts, visibility 400–900m. Visibility 400–1400m overall. Ceiling 70–500ft. All runways severely contaminated. Technical problems: Severe issues (40% moderate fuel leak, 30% critical fuel leak, 30% engine out). Combinations include engine out with fuel leaks or critical fuel leak with general warnings in extreme weather. 25% probability of additional medical emergency.
Extreme	Very severe	15% heavy, 85% extreme	Independent sampling per airport with 85% extreme conditions. Visibility as low as 400–900m. Winds 35–60kt. Sporadic shifts of 45–90° standard. Visibility 400–1400m overall. Ceiling 70–500ft. Standing water or compacted snow (100%). Technical problems: Very severe issues (50% critical fuel leak, 50% engine out). Often simultaneous critical fuel leak and engine out in extreme weather conditions. 30% probability of additional medical emergency.

Continued on next page

Table A.1 – continued from previous page

Scenario Weather	Failure Severity	Weather Distribution	Detailed Conditions (sampled per airport option)
Extreme	Critical	15% heavy, 85% extreme	Each airport samples: predominantly extreme scenarios at 85%. Sporadic wind shifts of 45–90° expected. Winds 35–60kt. Visibility 400–900m in extreme cases. All runways contaminated. Ceiling 70–500ft. Independent weather per option. Technical problems: Critical failures (60% critical fuel leak, 40% engine out). Typically simultaneous critical fuel leak and engine out with potential additional general warnings in extreme weather environment. 40% probability of additional medical emergency.
Catastrophic	Very light	5% heavy, 95% extreme	Near-exclusively extreme conditions at 95% with independent per-airport sampling. Visibility 400–900m. Ceiling 70–500ft. Runways with standing water or compacted snow (100%). Winds consistently 35–60kt. 95% probability of sporadic shifts of 45–90° per airport. Technical problems: Single minor issue (70% minor fuel leak, 30% minor general warning). 15% probability of additional medical emergency.
Catastrophic	Light	5% heavy, 95% extreme	Each option independently samples: 95% extreme scenarios. Extreme category: winds 35–60kt with sporadic shifts of 45–90°, visibility 400–900m. 5% heavy scenarios. Visibility 400–900m overall. Ceiling 70–500ft. Severe contamination throughout all options. Sporadic wind shifts nearly guaranteed. Technical problems: Single minor issue (20% minor general warning, 40% minor fuel leak, 40% moderate fuel leak). 15% probability of additional medical emergency.
Catastrophic	Moderate	5% heavy, 95% extreme	Weather distribution per airport: almost exclusively extreme at 95%. Visibility 400–900m. Ceiling 70–500ft. All runways with standing water or compacted snow. Wind speeds 35–60kt. Sporadic directional shifts of 45–90° standard (95% probability). Technical problems: Multiple minor issues (30% minor fuel leak, 50% moderate fuel leak, 20% major general warning) in combinations. 20% probability of additional medical emergency.

Continued on next page

Table A.1 – continued from previous page

Scenario Weather	Failure Severity	Weather Distribution	Detailed Conditions (sampled per airport option)
Catastrophic	Severe	5% heavy, 95% extreme	Independent sampling per option: near-exclusively extreme at 95%. Extreme scenarios: winds 35–60kt, sporadic shifts, visibility 400–900m. Visibility 400–900m overall. Ceiling 70–500ft. All runways severely contaminated. Technical problems: Severe issues (40% moderate fuel leak, 30% critical fuel leak, 30% engine out). Combinations include engine out with fuel leaks or critical fuel leak with general warnings in catastrophic conditions. 25% probability of additional medical emergency.
Catastrophic	Very severe	5% heavy, 95% extreme	Each airport samples with 95% extreme scenarios. Visibility as low as 400–900m. Winds 35–60kt. Sporadic shifts of 45–90° nearly certain. Ceiling 70–500ft. Standing water or compacted snow (100%). Technical problems: Very severe issues (50% critical fuel leak, 50% engine out). Often simultaneous critical fuel leak and engine out in catastrophic weather environment. 30% probability of additional medical emergency.
Catastrophic	Critical	5% heavy, 95% extreme	Per-airport weather sampling: catastrophic conditions with 95% extreme scenarios. Sporadic wind shifts of 45–90° nearly guaranteed. Winds 35–60kt. Visibility 400–900m. All runways contaminated. Ceiling 70–500ft. Each option samples independently. Technical problems: Critical failures (60% critical fuel leak, 40% engine out). Typically simultaneous critical fuel leak and engine out, often with additional general warnings creating maximum emergency scenario requiring immediate landing in catastrophic weather. 40% probability of additional medical emergency.

B Hub Airports List

Table B.1: European Airlines Hub Airports with their geographical locations and hub types.

ICAO	Airport Name	Latitude	Longitude	Airline (Type)
EDDF	Frankfurt Airport	50.033	8.571	Lufthansa (Main)
EDDM	Munich Airport	48.354	11.786	Lufthansa (Main)
EDDH	Hamburg Airport	53.630	9.850	Lufthansa (Supporting)
EDDB	Berlin Brandenburg Airport	52.560	13.288	Lufthansa (Supporting)
LFPG	Paris Charles de Gaulle Airport	49.013	2.550	Air France (Main)
LFPO	Paris Orly Airport	48.723	2.380	Air France (Supporting)
EHAM	Amsterdam Schiphol Airport	52.309	4.764	Air France (Main)
LFBO	Toulouse Blagnac Airport	43.629	1.364	Air France (Supporting)
EGLL	London Heathrow Airport	51.471	-0.462	British Airways (Main)
EGKK	London Gatwick Airport	51.148	-0.190	EasyJet (Main); British Airways (Supporting)
EGFF	Cardiff Airport	51.383	-3.373	British Airways (Supporting)
EGPF	Glasgow Airport	55.872	-4.433	British Airways (Supporting)
EIDW	Dublin Airport	53.421	-6.270	Ryanair (Main)
EGSS	London Stansted Airport	51.885	0.235	Ryanair (Supporting)
EYKA	Kaunas Airport	54.964	23.906	Ryanair (Supporting)
LEZL	Seville Airport	36.844	-6.061	Ryanair (Supporting)
LIML	Milan Malpensa Airport	45.631	8.728	EasyJet (Main)
EGGW	Luton Airport	51.875	-0.368	EasyJet (Supporting)

Continued on next page

Table B.1 – continued from previous page

ICAO	Airport Name	Latitude	Longitude	Airline (Type)
LTFM	Istanbul Airport	41.275	28.752	Turkish Airlines (Main)
LTFJ	Sabiha Gökçen Airport	40.899	29.309	Turkish Airlines (Supporting)
EKCH	Copenhagen Airport	55.618	12.656	SAS (Main)
ESSA	Stockholm Arlanda Airport	59.650	17.939	SAS (Supporting)
ENGM	Oslo Gardermoen Airport	60.194	11.100	SAS (Supporting)
LEMD	Madrid Barajas Airport	40.472	-3.563	Iberia (Main)
LEBL	Barcelona Airport	41.297	2.078	Iberia (Supporting)
LIRF	Rome Fiumicino Airport	41.794	12.250	Alitalia (Main)
LIML	Milan Linate Airport	45.445	9.277	Alitalia (Supporting)
LHBP	Budapest Ferenc Liszt Airport	47.437	19.265	Wizzair (Main)
EPWA	Warsaw Chopin Airport	52.166	20.967	Wizzair (Main)
EPKT	Katowice International Airport	50.474	19.080	Wizzair (Supporting)
LRCL	Cluj-Napoca International Airport	46.785	23.686	Wizzair (Supporting)
LYBE	Belgrade Nikola Tesla Airport	44.819	20.962	Air Serbia (Main)

C Model Hyperparameters

Hyperparameter	Classification	Ranking
Model Type	Linear Weighted Sum	Linear Weighted Sum
Input Dimension	19 (context:4 + option:15)	19 (context:4 + option:15)
Output Dimension	1 (score per option)	1 (score per option)
Number of Parameters	19 (learnable weights)	19 (learnable weights)
Number of Options	5	5 / 10 / 15
Optimizer	Adam	Adam
Learning Rate	0.001	0.001
Loss Function	Cross-Entropy	ListNet (NDCG approximation)
Loss Temperature	–	1.0
Batch Size	32	32
Training Epochs	5	5
Weight Initialization	Random Normal	Random Normal
Training Samples (5 opt)	200,000	200,000
Training Samples (10/15 opt)	–	50,000
Holdout Samples (5 opt)	50,000	50,000
Holdout Samples (10/15 opt)	–	1,000

Table C.1: Comparison of Weighted Sum hyperparameters for classification and ranking tasks. Both use identical linear architectures with 19 learnable parameters, differing only in loss function: Cross-Entropy for classification and ListNet (softmax cross-entropy over relevance distributions) for ranking.

	Simple (Interpretable)		Full (Performance)	
Hyperparameter	**Classification**	**Ranking**	**Classification**	**Ranking**
Objective	multi:softmax	rank:ndcg	multi:softmax	rank:ndcg
Number of Estimators	20	20	5,000	5,000
Max Depth	2	2	8	8
Learning Rate (η)	0.3	0.3	0.01	0.02
Subsample	1.0	1.0	0.8	0.9
Column Sample by Tree	1.0	1.0	0.8	0.85
L1 Regularization (α)	0.1	0.1	0.1	–
L2 Regularization (λ)	0.1	0.1	1.0	–
Min Child Weight	5	5	3	2
Gamma (γ)	0.1	0.1	0.1	0.05
Early Stopping Rounds	10	10	50	100

Table C.2: Comparison of XGBoost hyperparameters between interpretable (Simple) and performance-optimized (Full) model variants for both classification and ranking tasks.

	Simple (Interpretable)		Full (Performance)	
Hyperparameter	**Classification**	**Ranking**	**Classification**	**Ranking**
Population Size	20	20	650	650
Input Dimension	79	19	79	19
Output Dimension	5	1	5	1
Maximum Trials	400,000	600,000	400,000	600,000
Performance Trials	3,000	3,000	3,000	3,000
Loss Function	One-Hot	MSE	One-Hot	MSE
Error Threshold (e_0)	0.012	0.01	0.010	0.008
Learning Rate (α)	1	0.1	0.75	0.15
Fitness Decay (β)	0.08	0.2	0.1	0.15
Min Experience (ν)	5	10	5	10
Fitness Threshold (δ)	0.1	0.1	0.1	0.1
Deletion Threshold (θ_{del})	38	20	32	25
Condition Min	0.0	-1.0	0.0	-1.0
Condition Max	1.0	1.0	1.0	1.0
Condition Spread Min	0.33	0.1	0.28	0.08
Condition η	0.09	0.01	0.09	0.01
Prediction x_0	1.0	0.0	1.0	0.0
Prediction η	0.12	0.01	0.09	0.02
Prediction η_{min}	10^{-6}	10^{-8}	10^{-6}	10^{-8}
Early Stopping Patience	10,000	10,000	10,000	10,000

Table C.3: Comparison of LCS (XCSF) hyperparameters between interpretable (Simple) and performance-optimized (Full) model variants for classification and ranking tasks for the representative (Medium) dataset.

Hyperparameter	Classification (SimpleNN)	Ranking (ListNet)
Input Dimension	79 (context:4 + options:15×5)	19 (context:4 + option:15)
Output Dimension	5 (class labels)	1 (relevance score)
Hidden Layers	2	2
Hidden Dimension	32	32
Activation Function	ReLU	ReLU
Optimizer	Adam	Adam
Learning Rate	0.01	0.001
Batch Size	64	64
Training Epochs	100	100
Loss Function	Cross-Entropy	ListNet (KL Divergence)
Regularization	None	None
Early Stopping	Accuracy plateau ($\Delta < 0.001$)	NDCG@1 plateau ($\Delta < 0.001$)
Number of Options	5	5 / 10 / 15
Cross-Validation Folds	5	5
Random State	42	42
Training Samples	200,000	200,000

Table C.4: Comparison of Neural Network hyperparameters for classification (SimpleNN) and ranking (ListNet) tasks. Both architectures prioritize interpretability with minimal hidden layers and no regularization techniques such as Dropout or BatchNorm.