MSc thesis in Engineering and Policy Analysis

Thinking Outside the Box: A Critical Evaluation of Oblique Decision Tree Algorithms for Scenario Discovery

Jasper Tycho ter Horst

2025

TU Delft

MSc thesis in Engineering and Policy Analysis

Thinking Outside the Box: A Critical Evaluation of Oblique Decision Tree Algorithms for Scenario Discovery

Jasper Tycho ter Horst
Student Number: 5115078

July 2025

A thesis submitted to the Delft University of Technology in partial fulfillment of the requirements for the degree of Master of Engineering and Policy Analysis

Jasper Tycho ter Horst: *Thinking Outside the Box: A Critical Evaluation of Oblique Decision Tree Algorithms for Scenario Discovery* (2025)

The work in this thesis was created under:

Policy Analysis
Multi-Actor Systems
Faculty of Technology, Policy & Management
Delft University of Technology

Chair:	Prof. Dr. J.H. Kwakkel
First Supervisor:	Prof. Dr. J.H. Kwakkel
Second Supervisor:	Prof. Dr. M.E. Warnier
Advisor:	Dr. P. Steinmann

Executive Summary

In an era shaped by systemic risks and long-term challenges such as climate change, technological disruption and geopolitical instability, policymakers must make consequential decisions without knowing what the future holds. This condition, known as deep uncertainty, arises when the relationships between actions and outcomes are contested and traditional prediction methods no longer apply. To address this, analysts have turned to computational tools such as scenario discovery, which explores thousands of simulated futures to identify combinations of factors that lead to policy success or failure. It provides decision-makers with understandable *scenarios*: data-grounded narratives of the future that highlight critical vulnerabilities and opportunities. These scenarios help enable the design of strategies that are robust and adaptive.

The effectiveness of scenario discovery, however, depends on the algorithms used to find these scenarios. Among the most widely used approaches are PRIM and CART, which are valued for producing simple rules that are easy to communicate, as they set thresholds on one factor at a time. Yet, their fundamental limitation is that the boundaries of real-world problems are often not aligned with these simple, independent factors. Instead, they are shaped by more complex interactions and trade-offs between variables. To partially address this, the more advanced PCA-PRIM method was introduced. By applying a single, global rotation to the data, it can better capture some of these relationships. This added flexibility, however, comes at a significant cost to interpretability. Furthermore, its reliance on a single, global rotation for the entire dataset is a structural limitation, as the important relationships within a problem may have different orientations in different regions of the uncertainty space.

This thesis evaluates whether a more advanced class of algorithms, known as oblique decision trees, can overcome this limitation. These methods are considered theoretically and geometrically more powerful because they generate flexible, locally adapted rules that account for combinations of multiple variables. This capability should, in principle, enable them to capture complex relationships more precisely. The potential motivated the central research question of this thesis: to what extent can oblique decision tree algorithms improve scenario discovery compared to established methods?

To answer this question a three-phase research design was chosen. The first phase benchmarked six different oblique tree algorithms on a novel set of synthetic geometric problems, systematically testing their performance to select the most robust candidate. The second phase conducted a detailed comparative analysis, pitting the top-performing oblique algorithm, HHCART(D), against the established methods, PRIM, CART, and PCA-PRIM, on challenging two-dimensional problems to evaluate their respective strengths in partitioning and interpretation. The final phase applied all four methods to a high-dimensional, real-world model of the European Union's energy transition, providing a test of their practical utility.

The findings from this evaluation are clear and serve as a strong cautionary note for the field. The theoretical promise of the tested oblique algorithms was not realised in practice. While the algorithm could, in some cases, match the statistical performance of established methods, it consistently produced scenarios that were more fragmented, more complex, and harder to understand. Most tellingly, when faced with the real-world policy problem, the algorithm failed to find meaningful relationships between factors. Instead, it produced highly complex rules that, upon inspection, were merely disguised versions of simple, one-factor rules. This outcome illustrates a key drawback: scenarios became harder to interpret without yielding corresponding gains in insight or performance.

The findings of this thesis carry implications for both research and practice in scenario discovery. It makes three key contributions. First, it presents the first systematic, benchmark-driven evaluation of oblique decision tree algorithms for scenario discovery. Second, it delivers an extensible, open-source Python framework that enables reproducible comparisons across methods. Third, it introduces a new set of stylised benchmark problems designed to expose the limitations of existing methods. However, the study focused on algorithms that construct trees step-by-step using local decisions and do not simplify their splits by selecting only the most informative variables. Future research should explore methods that optimise the tree structure more globally and allow each split to use a small, most relevant subset of features. Such approaches may offer a better trade-off between flexibility and interpretability. For practitioners and policymakers, the central conclusion reaffirms the value of established methods like PRIM, whose simplicity and communicability remain essential for decision-making under deep uncertainty.

Acknowledgements

If you are reading this, it means the simulations finally finished, the code finally ran without errors, and I have finally come out of my nerdy rabbit hole.

I would like to thank Dr. P. Steinmann for his guidance throughout this project. Our weekly meetings helped keep the work focused and on track. I am also grateful to Prof. dr. J.H. Kwakkel and Prof. dr. M.E. Warnier for their valuable feedback and support. I thank the DelftBlue team for providing the computing resources that made the experiments in this thesis possible. I also thank Torsha Majumder for sharing her code on oblique decision trees, which served as the foundation for several of the algorithms evaluated in this work.

I am deeply grateful to my friends and family for their unwavering support throughout this journey. Their encouragement, perspective, and occasional distractions helped me maintain balance and reminded me that there is more to life than debugging Python code. For the curious (or the brave), all code developed for this thesis is available online.[1]

Jasper Tycho ter Horst
Rotterdam, 2025

[1] See Section A for the repository link.

Contents

Acronyms

1 Introduction

1.1 Decision-Making Under Deep Uncertainty and the Role of Scenario Discovery

In many policy domains, decision-makers must act under conditions of deep uncertainty: situations where the system structure is contested, the probability distributions of key parameters are unknown, and stakeholders disagree on what outcomes matter most (Lempert, 2003). These conditions arise in areas such as climate adaptation, pandemic response, and long-term energy transitions, where traditional optimisation and probabilistic forecasting often fail to provide strategies that remain robust across a wide range of futures (Constantino & Weber, 2021; Hadjisotiriou et al., 2023; Paredes-Vergara et al., 2024).

To support decisions in these contexts, computational frameworks have emerged that enable the systematic exploration of large ensembles of simulated futures. Among these, *scenario discovery* has become a used approach for identifying combinations of uncertain inputs that lead to specific policy-relevant outcomes (Lempert et al., 2008). Instead of predicting the most likely future, scenario discovery asks under what conditions a particular outcome emerges. This inversion of conventional analysis allows analysts to identify vulnerabilities and opportunities, and to develop strategies that perform well across a diverse range of plausible futures.

Scenario discovery has been applied across a wide range of domains and is now integral to several decision-making methodologies under deep uncertainty. It has supported (policy-)analyses in climate change and water management (Haasnoot et al., 2011), resource scarcity (Kwakkel et al., 2013), transportation and logistics (Halim et al., 2016), the energy transition (Kwakkel & Pruyt, 2013), and societal responses to climate change (Greeven et al., 2016). It also underpins robust approaches such as Robust Decision Making (Groves & Lempert, 2007), Many Objective Robust Decision Making (Kasprzyk et al., 2013), and Dynamic Adaptive Policy Pathways (Haasnoot et al., 2013), offering a structured means to assess vulnerabilities and opportunities, and support the design of robust or adaptive policies across a wide range of plausible futures.

1.2 Structural Limitations in Existing Scenario Discovery Methods

Scenario discovery relies on the use of statistical machine learning algorithms for rule induction, with the aim of identifying combinations of uncertain inputs associated with specific policy-relevant outcomes (Kwakkel & Cunningham, 2016). Currently, the main algorithm used for this purpose is the Patient Rule Induction Method (PRIM) (Friedman & Fisher, 1999), although other algorithms such as Classification and Regression Trees (CART) (Breiman et al., 1984) are occasionally applied (Kwakkel & Jaxa-Rozen, 2016; Lempert et al., 2008). Both PRIM and CART rely on axis-aligned threshold rules applied independently to individual input variables. These rules construct hyperrectangular regions in the input space that are easy to interpret and communicate, making them attractive for decision support. However, their axis-aligned structure limits geometric flexibility. As Dalal et al. (2013) observe, "regions [of interest] are often not well described by a hyper-rectangle", and may instead take a triangular form or lie along some other axes. This constraint limits the ability of such algorithms to capture rotated or irregular regions efficiently, often requiring deep or fragmented rule structures to approximate non-orthogonal boundaries (Dalal et al., 2013).

One algorithm that seeks to address this geometric limitation is Principal Component Analysis–based Patient Rule Induction Method (PCA-PRIM), which departs from the axis-aligned structure by applying a global rotation of the input space before executing the PRIM algorithm (Dalal et al., 2013). This transformation is based on principal component analysis and allows the method to perform axis-aligned splits in the rotated space, which correspond to linear partitions in the original space. As a result, PCA-PRIM can better capture rotated structures and has shown improved performance relative to standard PRIM. However, because the transformation is global, the method cannot adapt to local variation in

the input space. As the authors note, PCA-PRIM performs poorly on irregular shapes such as "a regular pentagon, a ring, a circle, or some strange concave shape". Moreover, its decision rules are defined over linear combinations of input variables, which reduces interpretability and complicates communication with decision-makers (Kwakkel & Cunningham, 2016; Steinmann, 2018).

These limitations point to a fundamental challenge in current scenario discovery practice: the trade-off between geometric flexibility and interpretability. Axis-aligned algorithms such as PRIM and CART offer intuitive descriptions, but often fail to capture decision-relevant regions that deviate from rectangular geometries. PCA-PRIM introduces greater geometric flexibility by rotating the input space, but applies a single, global transformation that cannot adapt to local variation and often sacrifices interpretability. What is currently lacking is a systematic investigation of whether more geometrically flexible algorithms can achieve better performance while preserving the interpretability required for effective decision support.

This thesis addresses that gap by evaluating oblique decision tree algorithms for scenario discovery. These models define decision boundaries through linear combinations of input features, allowing for rotated and locally adapted partitions (Breiman, 2001; Murthy et al., 1994). This added flexibility may improve alignment with the structure of decision-relevant regions. However, while their increased expressiveness holds potential to improve region identification, it may also make the resulting decision rules more difficult to intepret and communicate in a policy context. To date, no evaluation has tested whether oblique decision tree algorithms can achieve a balance between performance and interpretability in scenario discovery. This study offers such an evaluation.

1.3 Research Questions

Building on the research gap outlined in the preceding section, the main research question guiding this thesis is:

Research Question

To what extent can oblique decision tree algorithms improve scenario discovery compared to established methods?

To structure this investigation, the research is organised into three empirical phases. The design of these phases and the corresponding sub-questions is informed by established evaluation criteria for scenario discovery, as well as by insights from the literature regarding the potential and limitations of oblique decision tree algorithms.

Together, the three phases operationalise the central research question by addressing it through the following sub-questions:

SQ1: *Which oblique decision tree algorithm demonstrates the best performance on benchmark datasets?*

This first phase evaluates a diverse set of oblique decision tree algorithms on benchmark datasets with known structures. The goal is to identify a single candidate for comparison based on overall performance across evaluation criteria.

SQ2: *How does the selected oblique decision tree algorithm compare to established scenario discovery methods when applied to benchmark datasets?*

This phase evaluates the selected oblique decision tree alongside PRIM, PCA-PRIM, and CART on representative benchmark problems. The comparison focuses on scenario discovery performance in terms of coverage and density, with close attention to interpretability through the number, shape, and complexity of the resulting partitions.

SQ3: *How does the selected oblique decision tree algorithm compare to established scenario discovery methods in identifying relevant and interpretable regions within a high-dimensional policy model?*

In the final phase, the selected algorithm is applied to a high-dimensional policy model and compared with PRIM, PCA-PRIM, and CART. This stage evaluates whether the algorithm can preserve interpretability while effectively identifying decision-relevant regions in a complex and realistic scenario discovery setting.

1.4 Research Structure

This thesis is organised into eight chapters. Chapter 2 establishes the conceptual foundation by reviewing scenario discovery methods and introducing oblique decision trees and their potential. Chapter 3 outlines the research design, including datasets, algorithms, and evaluation criteria. Chapters 4 to 6 report the results of the three research phases: Phase 1 benchmarks six oblique tree algorithms on on a set of synthetic benchmark problems; Phase 2 compares the selected method, to PRIM, PCA-PRIM, and CART on a subset of the synthetic benchmark problems; Phase 3 applies all four methods to a high-dimensional energy transition model. Chapter 7 highlights the key findings, relevance of the study, limitations, and potential for future research. Finally, Chapter 8 answers the three sub-questions and the main research question.

2 Literature Review

This chapter provides the background and motivation for the research. It begins by reviewing the shift from narrative scenario planning to simulation-based approaches, and outlines the limitations that led to the development of scenario discovery (Section 2.1). It then introduces the scenario discovery framework, including its methodology, evaluation criteria, and reliance on rule-induction algorithms (Section 2.2). The structural limitations of established algorithms such as PRIM, CART, and PCA-PRIM are then examined (Section 2.2.2). This motivates the introduction of oblique decision trees as a more flexible alternative, along with a discussion of their potential and challenges (Section 2.3). The chapter concludes by identifying the central research objective and the need for a structured empirical investigation (Section 2.4).

2.1 From Narrative Scenarios to Model-Based Exploration

Scenario planning emerged in the mid-twentieth century as a novel approach to strategic thinking under conditions of profound uncertainty. Initially developed at the RAND Corporation during the 1950s to aid the US Department of Defense in anticipating nuclear and geopolitical contingencies, it was later adapted for corporate strategy by Pierre Wack and colleagues at Royal Dutch Shell, who applied it to prepare for systemic shocks such as the 1973 oil crisis (Amer et al., 2013; Bradfield et al., 2005; Wack, 1985). Among the various methods that evolved, the intuitive logic approach became dominant, emphasising the construction of a small number of internally consistent narrative scenarios grounded in qualitative assessments of driving forces and critical uncertainties (Amer et al., 2013; Bradfield et al., 2005).

However, traditional scenario planning has long faced criticism for its methodological limitations. As Lempert et al. (2008) argue, such approaches "lacked an appropriate analytic foundation for inclusion in quantitative decision analyses". In complex and high-stakes policy settings, reliance on expert judgement and manual construction of scenario axes risks omitting key uncertainties or misrepresenting their interactions (Groves & Lempert, 2007; Lempert, 2003). The common practice of selecting two orthogonal axes, for example, often excludes relevant dimensions and cannot adequately represent the full uncertainty space. Furthermore, the inherently subjective process of crafting narrative scenarios makes the results highly dependent on the skill and credibility of the scenario team (Amer et al., 2013). As Walker et al. (2013) aptly summarises: "these traditional methods all founder on the same shoals: an inability to grapple with the long term's multiplicity of plausible futures. Any single guess about the future will likely prove wrong".

To address these shortcomings, simulation-based approaches began to emerge in the 1990s (Bankes, 1993). These methods employed computational models to generate large ensembles of plausible futures by systematically varying uncertain input parameters. In practice, these simulation ensembles were typically explored either informally, through expert selection of illustrative *what-if* cases, or more formally, through stakeholder-led processes that prioritised a subset of runs (Lempert et al., 2008). While representing a methodological advance over purely narrative approaches, such practices retained important limitations. As Groves and Lempert (2007) note, these methods still struggled with two foundational challenges: how to select a small number of scenarios that meaningfully represent a high-dimensional uncertainty space, and how to incorporate probabilistic or structural uncertainty into the scenario selection process.

These limitations are particularly critical in settings characterised by deep uncertainty, circumstances in which analysts do not know, or cannot agree on, the structure of the system, the probability distributions of key inputs, or the relevant criteria for evaluating outcomes (Lempert, 2003). Under such conditions, traditional scenario planning and ad hoc simulation analysis often led to an understatement of uncertainty in pursuit of tractable answers, thereby risking the production misleading conclusions (Lempert et al., 2006). What was needed instead were analytical tools capable of reasoning across a wide range of plausible futures, without requiring consensus on assumptions or normative goals.

2.2 The Scenario Discovery Framework

Scenario discovery was developed in the mid-2000s at the RAND Corporation as a structured, model-based alternative to narrative-driven scenario planning and ad hoc simulation exploration (Bryant & Lempert, 2010; Groves & Lempert, 2007; Lempert et al., 2006, 2008). It was designed to address the limitations of earlier approaches in supporting robust decision-making under conditions of deep uncertainty. Rather than constructing a few hypothetical futures a priori and observing their consequences, scenario discovery begins with a large ensemble of simulation cases and systematically works backwards to identify the input conditions most strongly associated with outcomes of interest (Bryant & Lempert, 2010). This reversal of logic allows analysts to pinpoint which uncertainties matter most for policy success or failure, without presupposing agreement on how the world works or what goals are normative.

To operationalise this, scenario discovery uses rule induction or machine learning algorithms to identify structured regions within the input space that correspond to outcome-relevant clusters. These regions provide compact, interpretable summaries of relationships between uncertain inputs and outcomes of interest. As Bryant and Lempert (2010) explain, the aim is to "summarize sets of plausible future states of the world that illuminate key vulnerabilities [and opportunities] in proposed policies and to describe these scenarios in a manner useful for decision-makers and stakeholders". In doing so, the method directly addresses two long-standing challenges: how to systematically select a small number of scenarios from a vast space of possibilities, and how to meaningfully incorporate probabilistic or structural uncertainty into policy analysis (Lempert et al., 2008).

The scenario discovery process typically unfolds in four or five sequential stages, each designed to move from a high-dimensional ensemble of simulation results to an interpretable set of decision-relevant rules (Bryant & Lempert, 2010; Lempert et al., 2008):

1. **Sampling the Input Space**
   The analysis begins by constructing a comprehensive experimental design that systematically varies all uncertain input parameters. Each sampled combination is run through a simulation model to produce a corresponding policy-relevant output. Sampling is generally done through Latin Hypercube Sampling (Kwakkel, 2017; van Droffelaar, 2020).

2. **Classifying Outputs**
   The simulation results are classified into binary categories, typically *of interest* and *not of interest*, based on an outcome threshold relevant to the policy context. These outcomes may represent undesirable conditions, such as policy failure or risk, but equally, they may describe desirable targets, such as achieving policy success, profitability thresholds, or resilience indicators.

3. **Identifying Box Sets**
   A rule induction algorithm is applied to the input-output data to identify candidate subregions of the input space that are strongly associated with the outcome of interest. The specific algorithms used are discussed in Section 2.2.2.

4. **Evaluating and Selecting Boxes**
   The analyst evaluates the candidate boxes using performance criteria such as coverage, density, and interpretability, these are discussed in Section 2.2.1. A preferred box is then selected based on how well it balances these criteria.

5. **(Optional) Iteration to Find Box Families**
   If a single box does not fully capture the outcome of interest, additional boxes can be identified iteratively. Points from earlier boxes are either excluded or reclassified as not of interest to discourage overlap (Guivarch et al., 2016).

In summary, scenario discovery marks a fundamental shift in the treatment of uncertainty. Rather than relying on speculation or expert judgment alone, it employs computational and statistical methods to uncover the uncertainties that shape decision-relevant outcomes, thereby helping to navigate deeply uncertain futures.

The Example of Scenario Discovery Steps box below provides a simplified, narrative illustration of how scenario discovery might be applied in practice, using an agricultural policy context to demonstrate the core steps.

Example of Scenario Discovery Steps

A policymaker seeks to understand under which conditions crop yields become critically low, using a simulation model of agricultural productivity.

Step 1: The analyst systematically samples the input space, varying weather patterns, soil moisture, and technology adoption rates, and records both the input conditions and their associated yields.

Step 2: The outputs are classified using a threshold that separates acceptable yields from critically low ones.

Step 3: A rule induction algorithm is applied on the data to identify conditions under which critically low outcomes occur. For example, it might reveal that such outcomes tend to arise when weather variability exceeds a certain threshold and soil moisture falls below a certain threshold.

Step 4: After evaluating performance metrics, the analyst selects the box or set of boxes that best capture the conditions associated with poor yields, offering a targeted basis for policy design.

2.2.1 Metrics for Evaluating Scenario Quality

As highlighted above, scenario discovery seeks to identify subspaces of the input domain that are strongly associated with outcomes of policy interest. To be useful, however, these subspaces must satisfy three principal criteria, as outlined by Lempert et al. (2008):

1. They should capture a high proportion of outcomes of interest in the dataset (*coverage*).
2. They should contain a high proportion of outcomes of interest relative to all cases within the region (*density*).
3. They should remain simple to interpret and use in policy contexts (*interpretability*).

The remainder of this section elaborates on each criterium and discusses how they interact in practice.

Coverage

Coverage quantifies the extent to which a scenario box set captures cases associated with a specified outcome of interest (Bryant & Lempert, 2010; Lempert et al., 2008). It measures the proportion of all outcome-relevant cases in the dataset that fall within the selected box(es). A coverage value of 1 indicates that all relevant cases are captured; a value of 0 indicates that none are included. Thus, the metric evaluates how fully the identified conditions represent the phenomenon of interest.

To formalise coverage, let X be the full set of cases, $X^I \subseteq X$ the subset of policy-relevant cases, and $B = \{B_1, \ldots, B_n\}$ the set of boxes. Then, coverage is defined as:

$$\text{Coverage}(B; X^I) = \frac{\text{Number of policy-relevant cases in the boxes}}{\text{Total number of policy-relevant cases}} = \frac{\left|\{x \in X^I \mid x \in \bigcup_{i=1}^{n} B_i\}\right|}{|X^I|} \quad (2.1)$$

Importantly, coverage should not be assessed in isolation. A box set with high coverage may still perform poorly if it also includes large numbers of irrelevant cases. The second metric, density, addresses this trade-off.

Density

Density evaluates the selectivity of a scenario box set by measuring the proportion of enclosed cases that correspond to the outcome of interest (Bryant & Lempert, 2010; Lempert et al., 2008). It is a precision-oriented metric: a value of 1 implies that nearly all cases within the box are outcome-relevant, whereas lower values indicate greater contamination by irrelevant cases. Decision-makers should find this measure important if they would like each scenario to be a strong predictor of the cases of interest (Bryant & Lempert, 2010).

To formalise density, let X be the full set of cases, $X^I \subseteq X$ the subset of policy-relevant cases, and $B = \{B_1, \ldots, B_n\}$ the set of boxes. Then, density is defined as:

$$\text{Density}(B; X, X^I) = \frac{\text{Number of policy-relevant cases in the boxes}}{\text{Total number of cases in the boxes}} = \frac{\left|\bigcup_{B_i \in B} x_j^I \mid x_j^I \in B_i\right|}{\left|\bigcup_{B_i \in B} x_j \mid x_j \in B_i\right|} \quad (2.2)$$

Coverage and density together characterise the statistical quality of a scenario box set. However, even when both metrics are high, a box set may still be too complex to interpret or communicate effectively, hindering its application in policy analysis. This motivates the third key criterion: interpretability.

Interpretability

Interpretability refers to the extent to which decision-makers can readily understand and apply the results of a scenario discovery analysis (Lempert et al., 2008). It is a prerequisite for policy relevance: even statistically robust scenarios have limited value if they cannot be communicated clearly or used meaningfully in decision processes. As Hadjimichael et al. (2024) emphasise, outputs must not only represent uncertainty accurately but also be cognitively accessible and practically actionable for their intended audiences.

While difficult to measure directly, interpretability is often approximated using two indicators: the number of boxes in the solution and the number of input parameters constrained within each box (Lempert et al., 2008). Based on his foundational work in narrative scenario planning, Schwartz (1997) argued that scenario sets should ideally contain three to four scenarios, each structured around two to three key uncertainties. Bryant and Lempert (2010) and Lempert et al. (2008) adopted and formalised this principle for scenario discovery, suggesting that solutions adhering to this guideline are more likely to be interpretable and actionable in policy contexts.

Trade-offs and Evaluation Strategy

Scenario discovery inherently involves trade-offs between coverage, density, and interpretability. As Lempert et al. (2008) note, gains in one criterium often come at the expense of another. Expanding box boundaries to increase coverage typically lowers density, as more irrelevant cases are included. Conversely, tightening box boundaries to improve density may exclude relevant cases, thereby reducing coverage. Interpretability introduces a further constraint: achieving high coverage and density often necessitates more complex rule sets, such as additional boxes or more thresholded dimensions.

These trade-offs imply that no single box set will simultaneously maximise coverage, density, and interpretability. Analysts therefore examine a range of candidate solutions along the Pareto frontier (Kwakkel, 2019), and apply expert judgement to select those most appropriate for the policy context (Bryant & Lempert, 2010). The value of scenario discovery lies in its ability to make these trade-offs explicit, enabling transparent and informed discussion of which uncertainties matter most and under what conditions policy-relevant outcomes arise.

2.2.2 Rule Induction Algorithms for Scenario Discovery

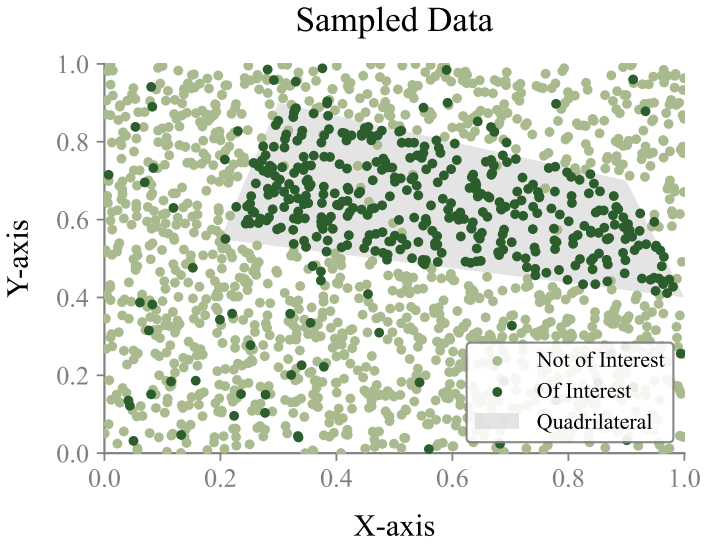
Selecting scenario boxes that balance coverage, density, and interpretability depends critically on the underlying rule-induction algorithm. These algorithms define how the input space is partitioned and thus shape which trade-offs are achievable in practice. While scenario discovery has been implemented using various techniques, two of the most widely established methods are the Patient Rule Induction Method (PRIM) and Classification and Regression Trees (CART) (Breiman, 2001; Friedman & Fisher, 1999). In addition, a globally rotated extension of PRIM, known as PCA-PRIM, has been proposed to improve geometric flexibility (Dalal et al., 2013).

This section examines how these algorithms construct decision-relevant regions and what structural assumptions they impose. To provide a clear conceptual comparison, each method is demonstrated on a stylised two-dimensional test case (Figure 2.1). The design of the shape follows the insight of Dalal et al. (2013), who argue that "in many situations, a triangular scenario or one that lies along some other axes than those of the model inputs may best illuminate a policy's vulnerabilities [or opportunities]". The example therefore defines the target region as a rotated quadrilateral: simple in shape, but misaligned with the coordinate axes.

The dataset consists of uniformly sampled points within a unit square, with a rotated quadrilateral embedded as the target region. Inside this quadrilateral, 95% of cases are of interest, and 5% are not; outside it, the proportion is reversed. This introduces noise in the labels, while the misalignment with the coordinate axes creates a setting in which axis-aligned methods are unable to accurately capture the underlying structure. The purpose is not to conduct an in-depth comparison of algorithmic performance, but to illustrate how different rule-induction methods construct decision regions to approximate the target subspace.

Each of the following sections presents one algorithm, combining a concise explanation with a visual demonstration on the test case. These examples highlight how the algorithms differ in constructing and representing outcome-relevant regions.

Figure 2.1
Synthetic two-dimensional scenario discovery problem. A rotated quadrilateral is embedded within a unit square as the target region. Inside the quadrilateral, 95% of points are of interest and 5% are not; outside, the proportions are reversed.



Patient Rule Induction Method

The Patient Rule Induction Method (PRIM), introduced by Friedman and Fisher (1999), is a sequential algorithm designed to identify subregions of the input space where the outcome variable is unusually high or low. It begins with an axis-aligned box that encloses the entire dataset and iteratively refines this region through a procedure called *peeling*. At each step, small candidate slices, typically removing α percent from either end of each input variable, are evaluated, and the one that most increases the *box mean* is selected. PRIM can be described as a *patient* or *non-greedy* hill climbing procedure (Kwakkel & Cunningham, 2016), because rather than committing to global partitioning, PRIM incrementally improves a single box.

To further refine the box after peeling, Friedman and Fisher (1999) propose a complementary bottom-up *pasting* strategy. This step reverses the peeling logic: it attempts to enlarge the box by adding small slices at the margins, one variable at a time. A candidate expansion is accepted only if it improves the box mean. Although pasting typically yields modest improvements, in some cases it can significantly recover relevant cases that were prematurely excluded during peeling. Together, these two procedures, top-down peeling and bottom-up pasting, allow PRIM to construct compact, interpretable boxes that concentrate a high proportion of outcome-relevant cases.

PRIM was adapted for scenario discovery because it offered a transparent, interactive way to explore trade-offs between coverage, density, and interpretability (Lempert et al., 2008). For this purpose, the method was applied to binary-classified outputs: simulation results were recoded such that cases of interest were assigned a value of 1, and all others a value of 0. Under this transformation, the *box mean* became interpretable as *density*, the proportion of outcomes of interest inside the box. Although the optimisation procedure remained unchanged, its meaning shifted: the box mean no longer represented a

general outcome average, but rather the concentration of decision-relevant points. PRIM continued to maximise this revised density metric, halting when the *support*, also called *minimum mass*, dropped below a user-specified threshold.

PRIMs visual outputs were redesigned to support this reinterpretation and to make trade-offs between scenario discovery metrics visible. The original peeling trajectory, plotted as box mean versus support, was reformulated to show density against coverage (Groves & Lempert, 2007). Here, *coverage* refers to the proportion of all relevant (label 1) cases in the dataset that fall within the current box. Each point on the trajectory corresponds to a candidate box from the peeling sequence, illustrating how density and coverage evolve across iterations. A colour gradient indicates the number of constrained input dimensions, providing a visual proxy for interpretability. This visualisation enables users to assess, at a glance, how much of the outcome of interest is retained (coverage), how much of the box is outcome-relevant (density), and how many dimensions the the box restricts (interpretability).

Figure 2.2 illustrates how PRIM operates on the quadrilateral test case, using a peeling parameter of $\alpha = 0.05$. Figure 2.2a shows the initial box enclosing the full dataset. In Figure 2.2b, a 5% slice is removed from the left edge to increase density. Figure 2.2c visualises the continued peeling process as a sequence of nested boxes. Figure 2.2d presents the coverage-density trajectory, where each point corresponds to a box from the sequence. In this example, colours are used consistently across the spatial and trajectory plots to represent the same sequence of boxes, enabling direct visual comparison. The trajectory's upward and leftward movement reflects the core trade-off in scenario discovery: increasing density requires peeling, which removes both relevant and irrelevant cases, thereby reducing coverage.

While PRIM provides a transparent and interpretable means of identifying decision-relevant regions, its geometric flexibility is limited. The method can only construct axis-aligned partitions, which makes it poorly suited to capturing rotated or non-orthogonal regions (Dalal et al., 2013). This limitation is evident in Figure 2.2, where the rectangular boxes fail to align with the orientation of the rotated quadrilateral. Because PRIM can only remove axis-aligned slices, it must exclude both relevant and irrelevant cases to improve density, making it impossible for any single box to achieve both high coverage and high density in this setting.

PCA-PRIM: Global Rotation of the Input Space

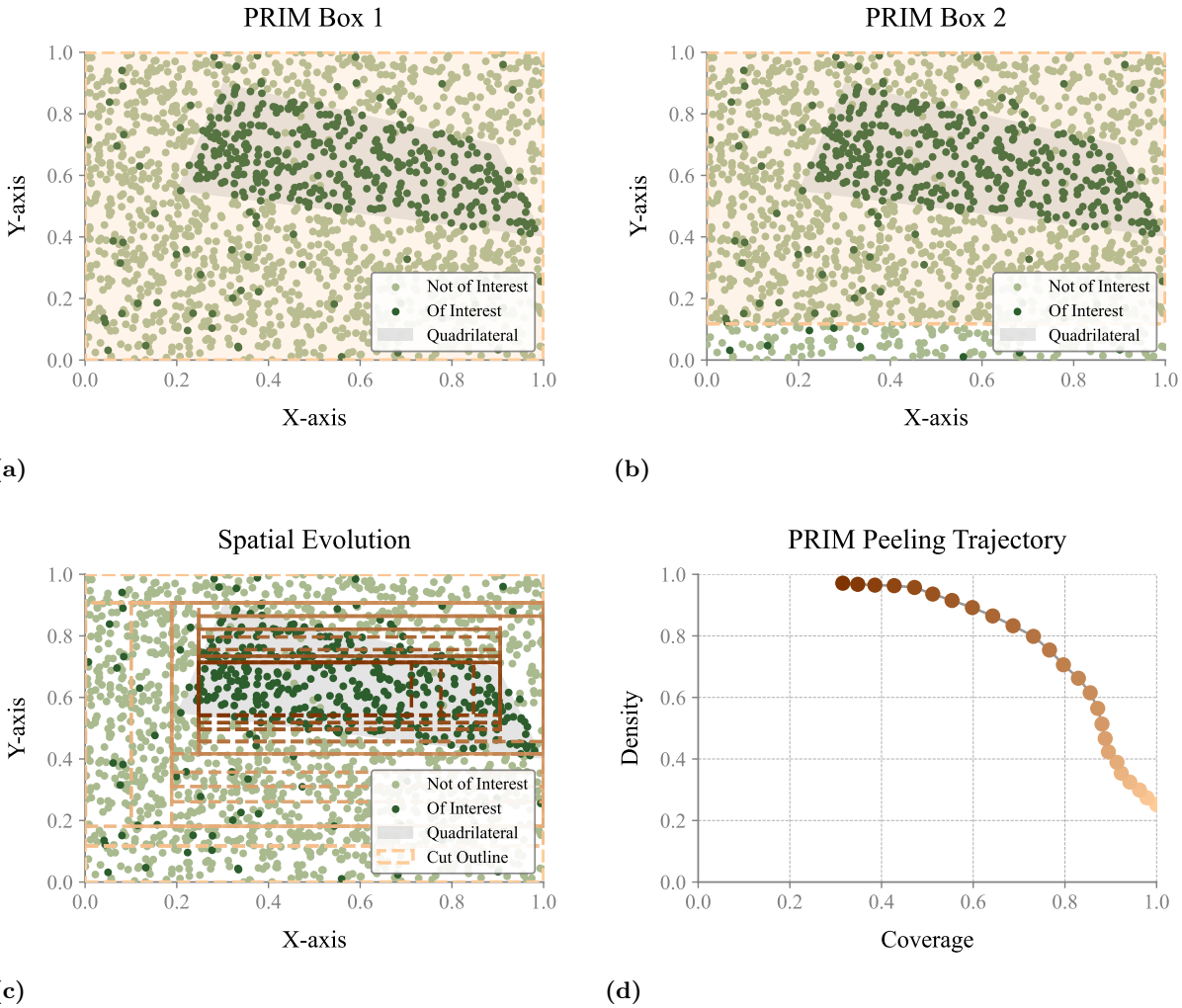
Principal Component Analysis–based Patient Rule Induction Method (PCA-PRIM), introduced by Dalal et al. (2013), was developed to address a key limitation of PRIM: its inability to efficiently capture decision-relevant regions that are not aligned with the coordinate axes. To overcome this, PCA-PRIM applies a Principal Component Analysis (Principal Component Analysis (PCA)) rotation before box construction begins. As shown in Figure 2.3a, this transformation reorients the input space so that its axes, the principal components, align with directions of greatest variance among the cases of interest. In the rotated space, PRIM then performs its standard axis-aligned peeling, generating rectangular boxes such as box 14 (Figure 2.3b). When these boxes are mapped back to the original coordinate system, they are no longer aligned with the original axes (Figure 2.3c), enabling the algorithm to approximate regions that standard PRIM would struggle to enclose.

This simple transformation improves both coverage and density when the region of interest is not axis-aligned. Dalal et al. (2013) report that PCA-PRIM increased the F1-metric, the harmonic mean of coverage and density, by 37% on synthetic benchmarks and by 16% in an application to a policy simulation model. A similar pattern is visible in Figure 2.3d, where PCA-PRIM outperforms standard PRIM on the rotated quadrilateral test case. Across the peeling trajectory, it identifies boxes that retain higher coverage while achieving higher density.

Despite these strengths, PCA-PRIM introduces two important limitations. First, it only applies a single global rotation to the entire input space. While effective for aligning the dominant orientation of a dataset, this approach cannot adapt to regions with varying directions. As Figure 2.3c demonstrates, even after rotation, box 37 fails to align with all four edges of the rotated quadrilateral. As Dalal et al. (2013) observe, a global transformation cannot capture the structure of more complex geometries, including "a regular pentagon, a ring, a circle, or some strange concave shape". The algorithm remains axis-aligned in the rotated space, and this rigidity limits its ability to approximate scenarios with multiple orientations or local curvature.

Second, PCA-PRIM reduces interpretability by replacing the original input variables with synthetic axes – linear combinations of features that do not correspond to any single policy-relevant input. In Figure 2.3c,

Figure 2.2
PRIM peeling process on the rotated quadrilateral, with peeling parameter $\alpha = 0.05$. Panel (a) shows the initial box enclosing the entire dataset. Panel (b) shows the first peel, in which 5% of the data is removed from one edge to increase density. Panel (c) shows the sequence of nested boxes generated through successive peeling steps. Panel (d) plots the corresponding peeling trajectory, with each point representing a box from the sequence and illustrating the trade-off between coverage and density.



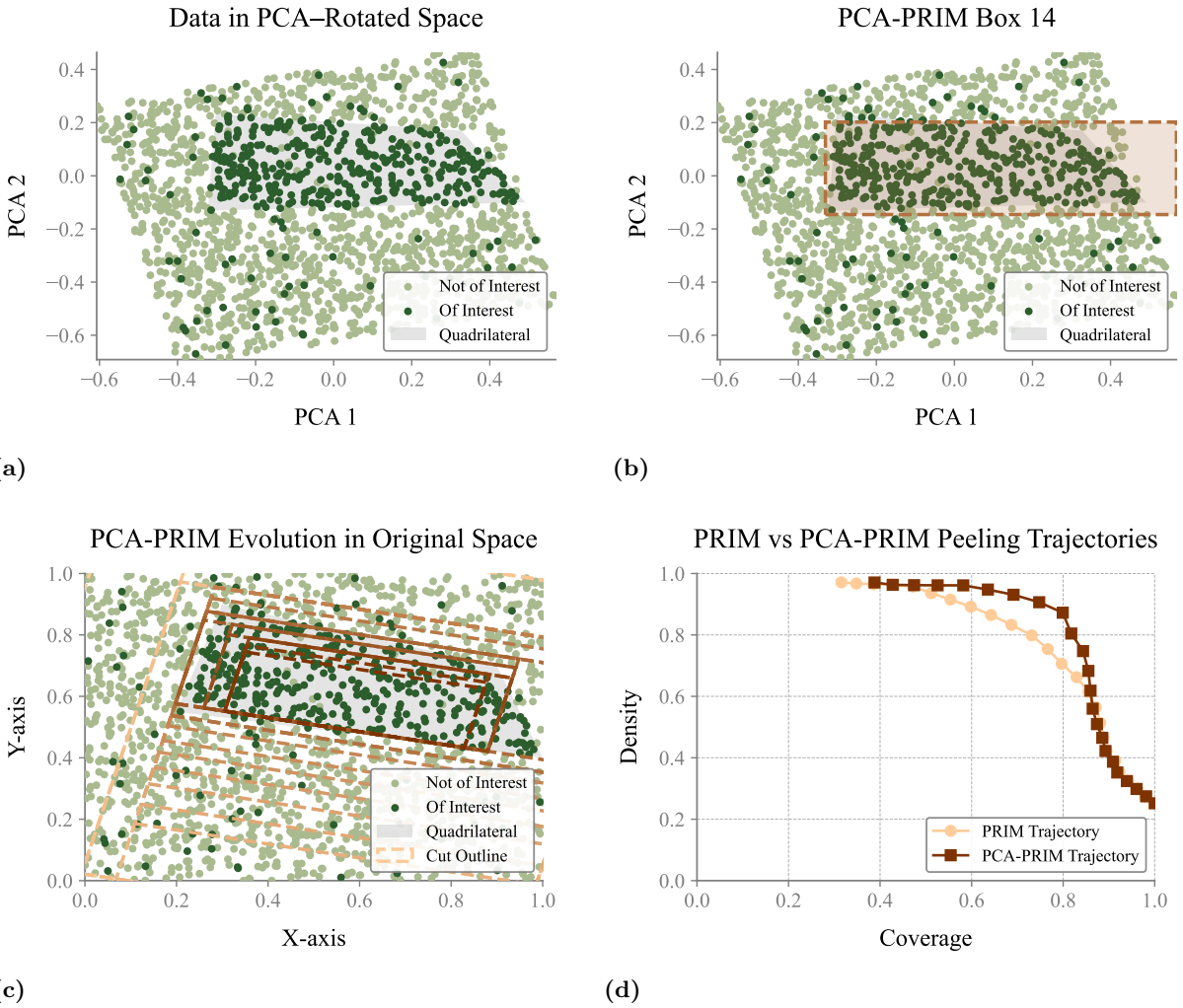
the boundary of box 14 no longer aligns with any individual variable, complicating the task of translating model outputs into actionable insights. This challenge has also been highlighted in the literature. For instance, Kwakkel et al. (2013) report that although PCA-PRIM improved the accuracy of scenario discovery in a system dynamics model, it hindered communication with non-technical stakeholders. Similar concerns are raised by Kwakkel and Cunningham (2016) and Steinmann (2018).

These limitations might be the reason that PCA-PRIM has seen limited adoption in applied studies. The unconstrained version has been used in only a handful of cases, including European Union electricity modelling (Hamarat et al., 2014; Loonen et al., 2013), system dynamics simulations (Pruyt et al., 2014), and market analysis of the copper industry (Hamarat, Pruyt, & Loonen, 2013). A constrained version, where PCA is applied within predefined groups of variables to retain interpretability, has been used once, also in the copper market context (Kwakkel et al., 2013). These examples suggest that while PCA-PRIM has performance advantages, its practical utility depends heavily on the interpretability demands of the use case.

In summary, PCA-PRIM enhances PRIMs ability to approximate non-axis-aligned regions by applying a global rotation of the input space prior to box construction. This enables axis-aligned peeling to better fit

rotated geometries, often improving coverage and density compared to standard PRIM. However, because the rotation is global and static, the method cannot adapt to regions with varying orientations. Moreover, its use of synthetic axes complicates interpretability, as decision rules no longer map transparently onto original input variables.

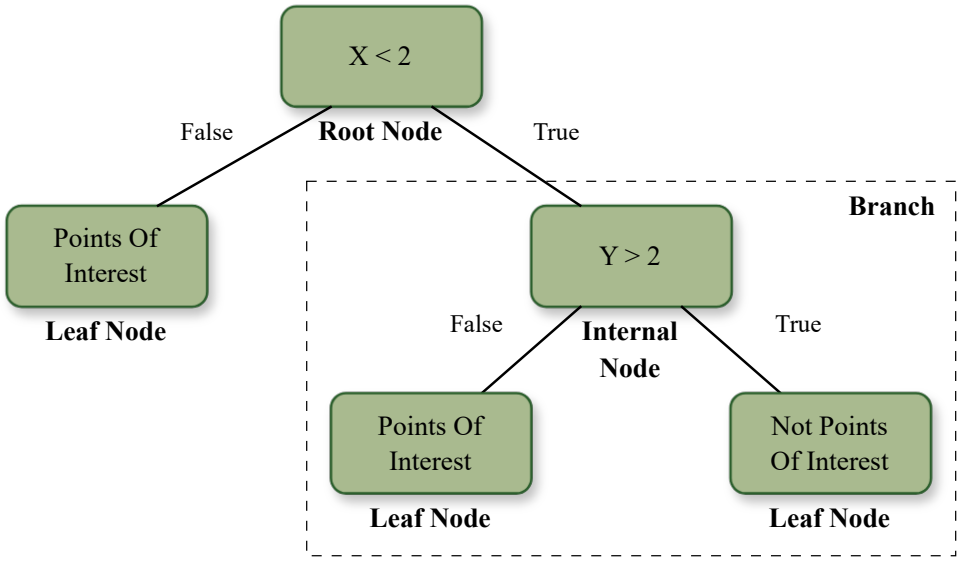
Figure 2.3
PCA-PRIM peeling process on the rotated quadrilateral, with peeling parameter $\alpha = 0.05$. Panel (a) shows the input dataset after PCA-based rotation, with the new axes aligned to the principal components of the cases of interest. Panel (b) shows candidate box 14, generated through axis-aligned peeling in the rotated space. Panel (c) visualises the sequence of boxes mapped back to the original input space, where the boxes are no longer aligned with the original axes. Panel (d) plots the peeling trajectories of standard PRIM and PCA-PRIM, illustrating how PCA-PRIM achieves improved trade-offs between coverage and density.



Classification and Regression Trees

Decision trees are a family of predictive models used in supervised learning, valued for their versatility, robustness, and interpretability (Costa & Pedreira, 2023). They partition the input space into regions defined by simple decision rules to classify or predict outcomes. The model adopts a hierarchical, tree-like structure (Figure 2.4) consisting of a root node, internal decision nodes, branches, and leaf nodes. Starting from the root, the tree applies a sequence of decision rules at each internal node, with each rule based on one or more input variables (e.g., $X < 2$). If the condition is satisfied, the data follow one branch; otherwise, they follow the other. This process continues recursively until the data reach a terminal leaf node, which assigns a final classification or prediction. The resulting tree provides a transparent and interpretable mapping from input conditions to predicted outcomes.

Figure 2.4
A conceptual representation of a binary decision tree.



Classification and Regression Trees (CART), introduced by Breiman et al. (1984), is one of the earliest and most influential algorithms within this family. It constructs trees by recursively applying binary, axis-aligned splits in a greedy fashion: at each node, it evaluates all candidate splits on each input variable and selects the split that yields the largest immediate reduction in impurity, typically measured by the Gini index or misclassification error. Candidate splits correspond to midpoints between sorted values of each variable. This process continues down each branch until a stopping criterion is met, such as a minimum node size, maximum depth, or purity threshold. The resulting tree partitions the input space into mutually exclusive, axis-aligned regions.

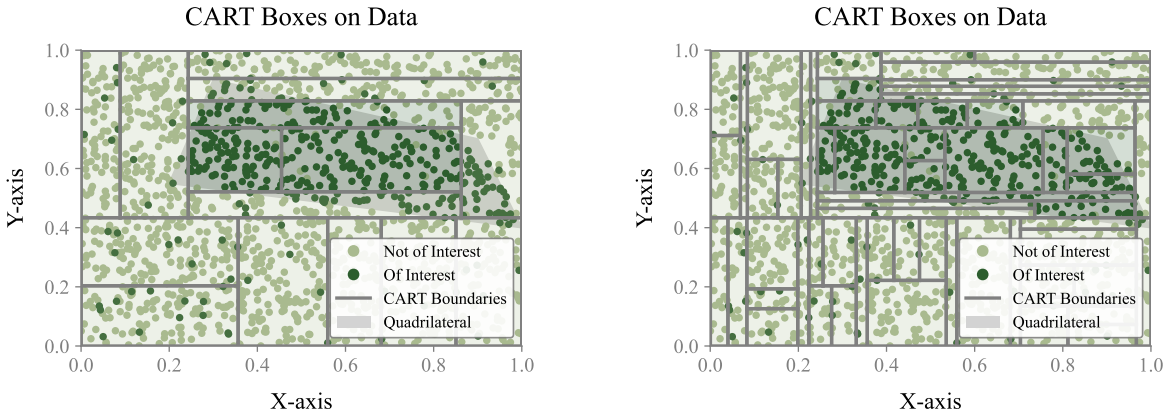
In scenario discovery, CART was first introduced by Lempert et al. (2008) as an alternative to PRIM for identifying decision-relevant input regions. In this context, each terminal node of the tree is interpreted as a box, defined by the cumulative restrictions imposed by the sequence of splits along the path to that node. Boxes are classified as *of interest* or *not of interest* based on the fraction of points of interest they contain. Unlike PRIM, which supports interactive exploration of trade-offs between coverage, density, and interpretability for box selection, CART operates in a fully automated manner, selecting splits greedily to maximise immediate impurity reduction.

However, CART faces two core limitations when used for scenario discovery under deep uncertainty. First, its axis-aligned split structure can force the model to construct an unnecessarily complex partition of the input space. When the outcome-relevant region is rotated or depends on interactions among variables, CART can only approximate it by combining many narrow, axis-aligned boxes (Dalal et al., 2013). This often leads to a proliferation of small, highly specific boxes (Lempert et al., 2008), which reduces interpretability and typically requires additional analyst effort to merge adjacent boxes into coherent decision-relevant scenarios (Lempert et al., 2008).

Second, the greedy nature of CART further limits its effectiveness. At each node, the algorithm selects the split that most reduces impurity locally, without regard for the global structure of the data or the long-term consequences of earlier decisions (Friedman & Fisher, 1999). This can result in unstable trees that miss broader patterns, especially in higher-dimensional settings, and the cumulative effect of many such local decisions can further diminish interpretability (Lempert et al., 2008).

Furthermore, Lempert et al. (2008) compared CART and PRIM in scenario discovery applications and found that while CART can perform comparably to PRIM in simple, low-dimensional problems, it performs less well in higher-dimensional contexts and offers less opportunity for interactive exploration of trade-offs. As a result, achieving interpretable and decision-relevant outputs with CART often requires substantially more analyst intervention.

Figure 2.5
Demonstration of the CART algorithm on test data under two minimum mass thresholds. Panel (a) applies a minimum mass of 5%, producing fewer, broader regions with a combined coverage of 0.63 and density of 0.89. Panel (b) reduces the minimum mass to 1%, resulting in more and narrower boxes with increased coverage of 0.80 and density of 0.92.



(a) **(b)**

Table 2.1
CART Boxes Summary Table with a minimum mass of 5%, combined coverage of 0.63 and combined density of 0.89.

Box	X range	Y range	Coverage	Density	Predicted Class
0	[0.00, 0.36]	[0.00, 0.20]	0.03	0.12	0
1	[0.00, 0.36]	[0.20, 0.43]	0.02	0.06	0
2	[0.36, 0.56]	[0.00, 0.43]	0.01	0.03	0
3	[0.56, 0.68]	[0.00, 0.43]	0.00	0.00	0
4	[0.68, 0.85]	[0.00, 0.43]	0.01	0.05	0
5	[0.85, 1.00]	[0.00, 0.43]	0.02	0.09	0
6	[0.00, 0.09]	[0.43, 1.00]	0.01	0.07	0
7	[0.09, 0.24]	[0.43, 1.00]	0.01	0.04	0
8	[0.24, 0.86]	[0.43, 0.52]	0.09	0.45	0
9	[0.24, 0.45]	[0.52, 0.74]	0.20	0.98	1
10	[0.45, 0.86]	[0.52, 0.74]	0.29	0.93	1
11	[0.24, 0.86]	[0.74, 0.83]	0.14	0.68	1
12	[0.86, 1.00]	[0.43, 0.83]	0.09	0.46	0
13	[0.24, 1.00]	[0.83, 0.90]	0.04	0.20	0
14	[0.24, 1.00]	[0.90, 1.00]	0.01	0.05	0

Methodological Trade-offs

The preceding comparison highlights that existing rule-induction methods for scenario discovery struggle to balance coverage, density, and interpretability when decision-relevant regions deviate from axis alignment. PRIM and CART, both constrained to axis-aligned splits, can represent such regions only through fragmented combinations of many boxes, undermining interpretability. PCA-PRIM offers greater geometric flexibility through global rotation, improving performance in rotated settings, but at the cost of interpretability and adaptability to local variations.

Taken together, these contrasts reveal a central limitation of current scenario discovery methods: none can simultaneously represent geometrically complex decision regions and maintain the simplicity needed for effective decision support. This motivates the exploration of oblique decision trees, a class of decision tree algorithms that construct splits along linear combinations of input variables. The following section introduces this family of methods and examines them in more depth.

2.3 Oblique Decision Trees for Enhancing Scenario Discovery

The limitations of established methods motivate the exploration of a more geometrically flexible alternative: oblique decision tree algorithms. This section introduces this class of models by first explaining the conceptual shift from axis-aligned to oblique splits and illustrating the geometric efficiency this offers. It then surveys algorithmic strategies for constructing oblique splits, highlighting the computational challenges and common design trade-offs. Finally, it evaluates the potential of oblique decision trees for scenario discovery, focusing on their expected gains in coverage and density, and the interpretability challenges they pose. The section concludes by discussing sparsity as a promising avenue for improving interpretability .

2.3.1 From Axis-Aligned Splits to Oblique Splits

Oblique decision trees depart from standard decision tree algorithms by replacing axis-aligned splits with decision boundaries defined by linear combinations of input features. Each internal node introduces a split of the form:

$$\mathbf{w}^\top \mathbf{x} \leq v, \tag{2.3}$$

where $\mathbf{w} \in \mathbb{R}^d$ is a weight vector, $\mathbf{x}$ is the input vector, and v is a threshold (or bias coefficient). This defines a hyperplane that divides the space into two regions. Because the weights in $\mathbf{w}$ can assign importance to multiple features at once, the resulting split is not restricted to alignment with any single axis.

This change in split structure has important consequences for geometric efficiency of the algorithms. In problems where the decision boundary depends on interactions, such as when both X_1 and X_2 must be simultaneously high, oblique trees can isolate the relevant region with a single split. An axis-aligned tree, by contrast, would require a series of horizontal and vertical splits to approximate the same condition, leading to deeper and more fragmented trees (Murthy et al., 1994).

Figure 2.6 provides a stylised illustration of this principle. It revisits the rotated quadrilateral benchmark introduced in earlier sections, which proved challenging for axis-aligned and globally rotated methods. PRIM and PCA-PRIM were unable to fully capture the region without sacrificing coverage or density, while CART required a proliferation of small boxes to approximate the shape. In contrast, the figure shows that four oblique splits could theoretically be sufficient to enclose the whole quadrilateral in one region.

While the splits in the figure are manually defined rather than generated by a trained model, they do underscore the geometric efficiency that oblique decision trees can theoretically offer. In settings where policy-relevant outcomes are caused by linear interactions between variables, this flexibility may prove essential to achieving high-quality scenarios.

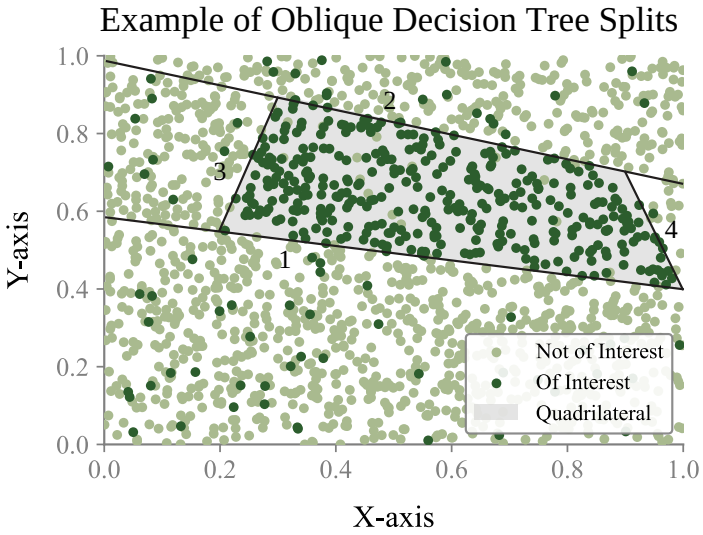
2.3.2 Strategies for Constructing Oblique Splits

While oblique splits offer greater geometric flexibility, finding the optimal split at each node is a far more complex task than in an axis-aligned tree. The process of finding a truly optimal decision tree is what computer scientists call *NP-hard*, a class of problems for which no efficient solution is known to exist (Carreira-Perpiñán & Tavallali, 2018). In practical terms, this means that as the number of uncertainties and data points grows, the time required to find the perfect tree explodes, quickly becoming unfeasible even for the most powerful computers.

Faced with this computational barrier, almost all practical tree-building algorithms that have stood the test of time, are (in spite of their suboptimality) greedy (Carreira-Perpiñán & Tavallali, 2018). The term *greedy* means that at each step of building the tree, the algorithm makes the decision that is locally optimal, without considering the downstream consequences for the global structure of the tree. Specifically, when creating a new split, the algorithm only seeks to find the single hyperplane that best reduces the impurity for the data at that immediate node. Once this locally best split is made, it is fixed permanently. The algorithm does not look ahead to see if a less optimal split now might enable better splits later, nor does it look back to revise earlier splits. As seen in the survey by Cañete-Sifuentes et al. (2021), most existing algorithms follow this greedy, node-by-node construction strategy.

The central challenge within this greedy framework is finding the best local split, specifically the parameters for the hyperplane $\mathbf{w}^\top \mathbf{x} \leq v$. For axis-aligned trees, this is straightforward and can be solved exactly

Figure 2.6
Four oblique lines enclosing the rotated quadrilateral in two dimensions. Although not derived from a trained model, these manually placed boundaries show that an oblique decision tree could, in principle, represent the region using a single node.



by enumerating all possible splits. For oblique trees, however, minimizing the impurity is much harder because it is a non-differentiable function of the weights (Carreira-Perpiñán & Tavallali, 2018). A tiny, continuous change to the weights in **w** can cause a group of data points to cross the boundary, making the impurity score jump unpredictably. Because there is no clear gradient or direction to follow towards a better solution, standard optimisation methods fail (Carreira-Perpiñán & Tavallali, 2018).

To address this, different algorithms adopt different strategies. Analytical methods compute w in closed form (Yildiz & Alpaydin, 2005), for example using projections derived from linear discriminant analysis or least-squares regression (Cañete-Sifuentes et al., 2021). These are fast, deterministic, and stable, but they are limited in the shapes they can capture. Iterative methods, by contrast, attempt to directly optimise the impurity criterion (Yildiz & Alpaydin, 2005), for example through local search, perturbing one weight at a time (e.g., coordinate descent), or sampling from stochastic neighbourhoods (e.g., simulated annealing or evolutionary algorithms) (Cañete-Sifuentes et al., 2021). While more flexible, these methods are sensitive to initialisation and computationally expensive. Hybrid approaches combine elements of both, for instance by fixing w analytically and tuning v through an iterative method.

It is important to note, however, that the reliance on greedy induction in the methods described above is a known limitation that can lead to suboptimal tree structures. To address this, some recent, more advanced methods have been proposed that depart from the greedy paradigm (Zharmagambetov et al., 2021). These non-greedy approaches, such as the Tree Alternating Optimisation method of Carreira-Perpiñán and Tavallali (2018), attempt to perform a global optimisation over the tree's parameters and structure. By iteratively refining the entire model rather than fixing splits one by one, such methods can produce more accurate, compact, and interpretable trees Zharmagambetov et al. (2021). These global optimisation techniques, while powerful, are typically more computationally expensive.

2.3.3 Evaluating Oblique Decision Tree Algorithms for Scenario Discovery

Having detailed their construction, this section evaluates the expected utility of oblique decision tree algorithms for scenario discovery. This evaluation requires a different lens than their typical assessment in machine learning. While overall predictive accuracy is not the primary goal, the model properties that lead to it are closely related to the specific criteria of scenario discovery: coverage, density, and interpretability (Lempert et al., 2008). This section will therefore explore the expected performance of oblique trees against these criteria, drawing on insights from the existing literature and exploring how their utility in this context could be improved.

Expected Coverage and Density Gains

The primary motivation for exploring oblique trees is their potential to improve on the core statistical metrics of scenario discovery: coverage and density. This expectation stems from their documented success in the broader classification literature. For problems with non-axis-aligned structures, empirical comparisons consistently show that oblique trees outperform their axis-aligned counterparts, achieving higher predictive accuracy while generating more compact models with fewer nodes and shallower depth (Li et al., 2003; Murthy et al., 1994; Wickramarachchi et al., 2016). This superior ability to define decision boundaries accurately and efficiently is expected to translate directly into scenarios with higher purity (density) and completeness (coverage), making it a highly desirable property for scenario discovery applications.

Further indication of this potential comes from the scenario discovery literature itself. As highlighted before, Dalal et al. (2013) demonstrated that applying a single global rotation to the input space improved the performance of PRIM, resulting in increased coverage and density. Although that study did not evaluate oblique decision trees, it shows that aligning split geometry with the orientation of decision boundaries improves scenario quality. Oblique trees are expected to improve upon this global rotation by allowing locally adaptive, branch-specific hyperplanes, and may therefore offer further improvements over established methods when the structure of the outcome-relevant region is not axis-aligned.

The Criticality of Interpretability

This potential for improved statistical performance, however, is met with a severe and potentially disqualifying challenge: a loss of interpretability. Unlike their axis-aligned counterparts, oblique splits are defined by linear combinations of features. This produces decision rules that can be difficult to interpret, validate, or communicate to stakeholders, particularly in high-dimensional policy settings. For example, a scenario boundary might be defined by the rule:

$$0.6 \cdot \text{GDP Growth} - 0.4 \cdot \text{Fuel Price} + 1.2 \cdot \text{Investment Risk} \leq 2.5, \tag{2.4}$$

While mathematically precise, such a rule lacks the intuitive clarity of a simple, single-variable threshold.

This challenge is compounded by another common limitation of oblique decision trees: most do not perform feature selection at the split level (Cañete-Sifuentes et al., 2021). Instead, they construct each decision rule using all available input variables, a characteristic known as creating *dense* splits. In a high-dimensional policy model with dozens of uncertain factors, this means a single decision rule could involve a weighted combination of all of them, making the resulting scenario logic nearly impossible to grasp. For applications such as scenario discovery, where the primary goal is not just to classify but to understand which uncertainties matter, this inherent complexity could severely undermine the applicability of the findings.

The Role of Sparsity in Improving Interpretability

One promising approach to managing the complexity of oblique decision rules is to encourage or enforce sparsity. Sparsity is the principle of constraining an algorithm to use only a small subset of the most influential features for any given decision rule. Several methods can enforce this, including heuristic search, mixed-integer optimisation, or L_1-based regularisation (Blanquero et al., 2020; Cañete-Sifuentes et al., 2021). By generating simpler decision logic, these approaches can produce a more explainable model while preserving the flexibility that makes oblique trees effective.

The result is a more interpretable, *sparse* rule that pinpoints a specific interaction between a few critical variables. This aligns well with the analytical objective of scenario discovery. Importantly, this gain in interpretability need not come at the expense of accuracy. Studies have found that applying sparsity constraints leads to only minimal reductions in predictive performance while dramatically improving model transparency (Blanquero et al., 2020; Heath et al., 1993). Given this favourable trade-off, sparsity is a highly desirable feature for any oblique tree algorithm intended for policy-relevant analysis.

2.4 Synthesis of Literature Insights

The preceding review has shown limitations of established scenario discovery methods, and introduced oblique decision trees as a promising extension. By enabling locally adaptive, oblique splits, they offer the potential to improve scenario discovery performance, specifically in terms of coverage and density. It has

to be highlighted that at the same time, their might be problems with interpretability, which has be be evaluated in this study.

To date, no systematic evaluation has been conducted to assess whether oblique decision trees can achieve a desirable balance between performance and interpretability in scenario discovery. Addressing this gap requires a structured empirical investigation. The next chapter therefore introduces a three-phase research design that operationalises this objective.

3 Methods

This chapter details the empirical strategy designed to assess whether oblique decision trees can enhance scenario discovery without sacrificing the interpretability crucial for policy analysis. The methodology is organised into three sequential phases, each built to address one of the research sub-questions introduced in Chapter 1. This framework progresses from controlled synthetic benchmarks to a complex, high-dimensional policy application, systematically evaluating the potential of oblique decision tree algorithms by consistently focusing on the core scenario discovery metrics of coverage, density, and interpretability. The design builds on the precedent set by Lempert et al. (2008), who combined algorithmic benchmarking with a policy model to compare PRIM and CART, providing a robust foundation for the comparative analysis that follows.

Phase 1: Identified the most robust and scalable oblique decision tree algorithm for scenario discovery. This was achieved by benchmarking six candidate methods on a set of synthetic classification problems designed to expose the limitations of established scenario discovery approaches. The benchmark problems varied in geometric shape and were tested under increasing levels of boundary noise, dimensional noise, and changing sample sizes. The evaluation focused on key scenario discovery metrics, including coverage, density, and training time, with tree depth used as a proxy for interpretability.

Phase 2: Compared the best-performing oblique algorithm, Householder Classification and Regression Tree using the dominant eigenvector of each class (HHCART(D)), against the established methods of PRIM, PCA-PRIM, and CART on benchmark problems. Prior to this comparison, the impact of structural regularisation parameters on the complexity of HHCART(D)'s partitions was assessed. The insights from this assessment, were used to configure HHCART(D) for the main comparison, allowing for a more meaningful evaluation against the typically compact outputs of the established methods. The final comparison focused on coverage and density, with interpretability judged by the number of positive subspaces and the complexity of the split expressions.

Phase 3: Evaluated the practical applicability and relative performance of the selected oblique algorithm, HHCART(D), against the established methods, PRIM, PCA-PRIM, and CART, in a high-dimensional policy context. All four methods were applied to a simulation model of the European Union energy system to identify scenarios leading to a high share of renewables. While the established methods were applied to the full set of uncertainties, the HHCART(D) approach involved a feature selection step to identify relevant inputs for the regularised algorithm. The scenarios from all methods were then benchmarked on coverage and density, with interpretability assessed using the same diagnostics as in Phase 2, to enable a final evaluation of their practical viability.

Having outlined the overall research strategy, this chapter now proceeds to detail the methodology for each phase. The following sections will elaborate on the specific experimental design, datasets, algorithms, and evaluation criteria used.

3.1 Phase 1: Benchmarking Oblique Decision Tree Algorithms

The goal of Phase 1 was to identify the most robust and scalable oblique decision tree algorithm for scenario discovery. This required determining which method most consistently isolates decision-relevant subspaces in classification problems that are structurally challenging for established scenario discovery methods. Performance in this phase was defined in terms of three criteria: high scenario coverage, high density, and low tree depth. The latter was taken as a proxy for interpretability, under the assumption that shallower trees produce less splits and therefore more comprehensible trees. Runtime was also analysed to give inside about the tractability of algorihtms for scenario discovery.

To support this evaluation, a two-stage experimental design was implemented. In the first stage, six oblique decision tree algorithms were benchmarked on eight synthetic benchmark problems, each constructed to target one or more known limitations of established scenario discovery methods. Specifically, the benchmark shapes were designed to test algorithmic robustness to: rotation of decision boundaries relative to the coordinate axes, local directional variation that undermines the efficacy of global alignment strategies like PCA, and non-linear or curved regions that cannot be easily approximated by axis-aligned or linear splits in the input space. To further stress-test the algorithms, each shape was evaluated under increasing levels of boundary noise, simulating conditions in which input-output relationships exhibit noisy, overlapping class boundaries.

The second stage focused on the top-performing algorithms from the initial benchmarks, evaluating whether their performance remained consistent under increased dimensionality and data volume. Specifically, irrelevant input features were added to test the algorithms' sensitivity to noisy variables and to examine the sparsity of the resulting decision rules. Additionally, the number of samples per dataset was scaled to assess runtime scalability and generalisation at larger problem sizes. These extensions aimed to simulate conditions characteristic of real-world scenario discovery tasks, where high-dimensional input spaces and big datasets are common, and to determine whether the algorithms could maintain strong performance without performance loss or excessive runtimes. Together, these tests enabled a systematic assessment of both robustness and runtime scalability.

The remainder of this section details the full experimental setup used in Phase 1. It begins by introducing the set of synthetic benchmark problems (Section 3.1.1) and the specific boundary and dimensional noise conditions they were subjected to (Section 3.1.2). The discussion then presents the candidate algorithms selected for this study (Section 3.1.3) and the two-stage experimental design to evaluate them (Section 3.1.4). Finally, the section outlines the performance criteria used for the assessment (Section 3.1.5) and summarises the computational infrastructure (Section 3.1.6).

3.1.1 Synthetic Benchmark Design

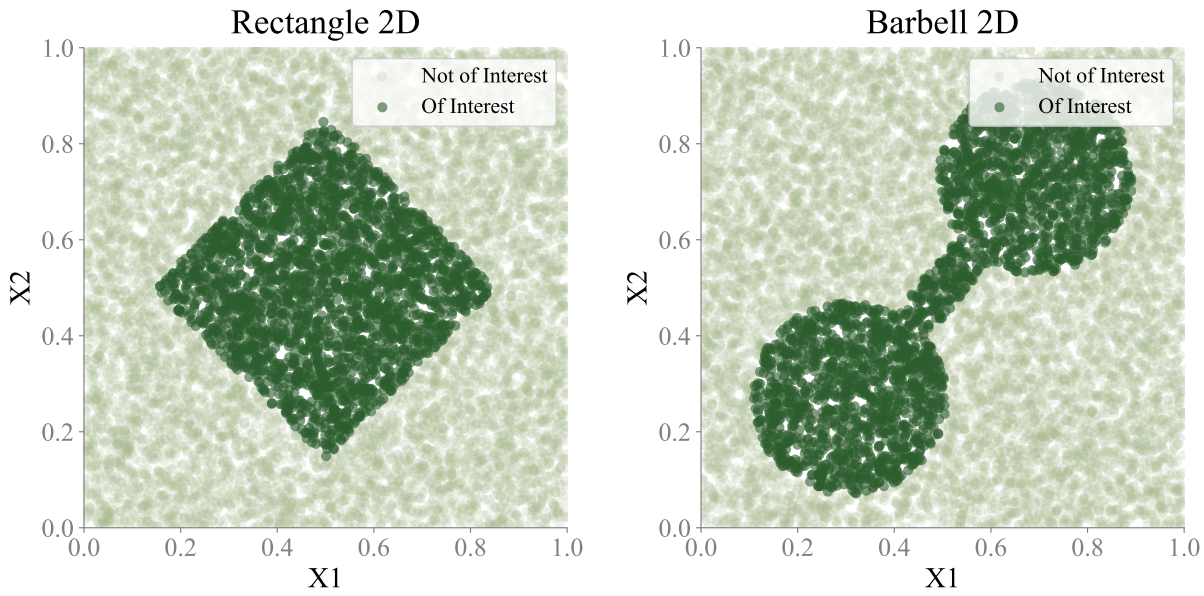
A rigorous evaluation of new algorithms requires a comprehensive set of benchmark problems. While existing studies have provided valuable comparisons, they often rely on a limited set of test cases, such as those introduced in the original evaluation of scenario discovery methods by Lempert et al. (2008). To provide a more systematic and conclusive test, this research introduces a new, expanded set of synthetic benchmarks. The central aim of this new set is to specifically target the known geometric limitations of methods like PRIM, CART, and PCA-PRIM, in order to determine if oblique decision trees offer clear benefits for handling more complex relationships.

To achieve this, this phase employed a set of eight synthetic binary classification tasks. Each benchmark defines a parametric decision region in either a two- or three-dimensional input space; data points falling inside a region are assigned a label of 1 (decision-relevant), while those outside are assigned a label of 0. The set comprises five two-dimensional shapes (rectangle, barbell, radial segment, sine wave, and star) and three three-dimensional shapes (barbell, radial segment, and saddle). Together, these were selected to represent three key geometric challenges for established algorithms:

- **Axis misalignment:** Decision boundaries are rotated with respect to the coordinate axes. This undermines axis-aligned methods such as PRIM and CART, which partition space using axis-parallel splits only.
- **Directional misalignment:** Decision boundaries follow multiple orientations that vary locally and are not confined to axis-aligned directions. This prevents axis-aligned methods such as PRIM and CART from capturing the structure. Global rotation approaches such as PCA-PRIM also fail, since one global transformation cannot resolve locally differing directions, as highlighted by Dalal et al. (2013).
- **Nonlinearity:** Decision regions exhibit curved, non-linear boundaries. Because both PRIM and PCA-PRIM rely on linear or rotated box approximations, they are structurally incapable of capturing such non-linear forms.

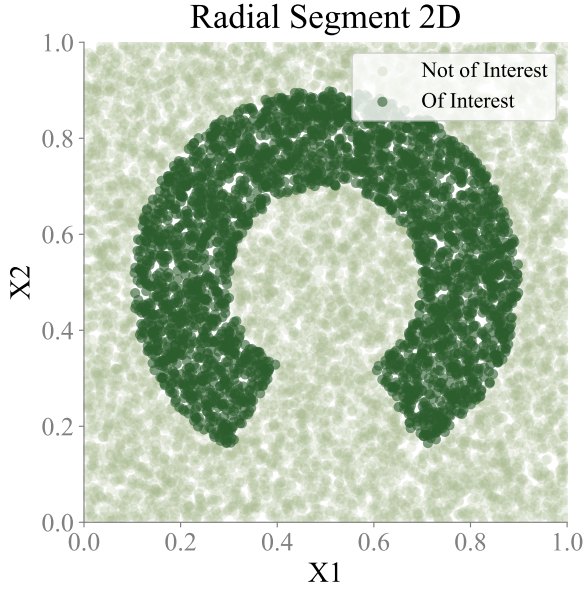
Figure 3.1 illustrates the full set of benchmark shapes. Table 3.1 provides a complete breakdown of the geometric properties tested by each shape, while their formal mathematical definitions and parameter settings are detailed in Appendix B.

Figure 3.1
Visualisations of the eight synthetic shapes used in Phase 1. Each shape defined a binary classification task and exhibited geometric properties that were expected to challenge traditional scenario discovery methods (see Table 3.1. Panels (a)–(e) display the five 2D shapes: a rectangle, a barbell, a radial segment, sine Wave, and five-point star. Panels (f)–(h) show the three 3D shapes: a barbell, a radial segment, and a saddle.

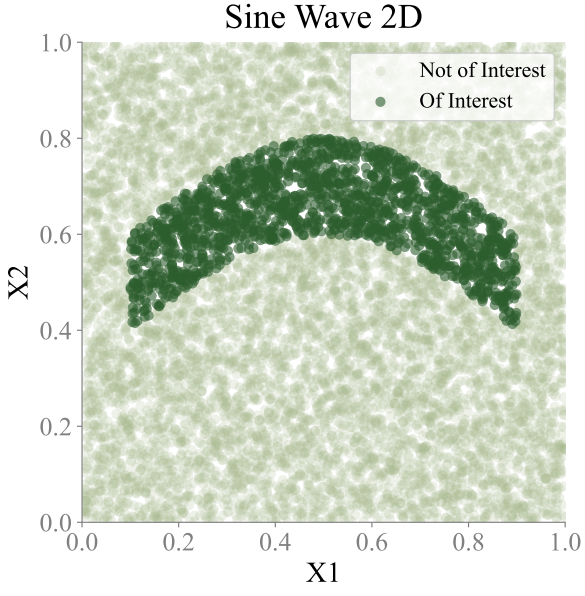


(a)

(b)

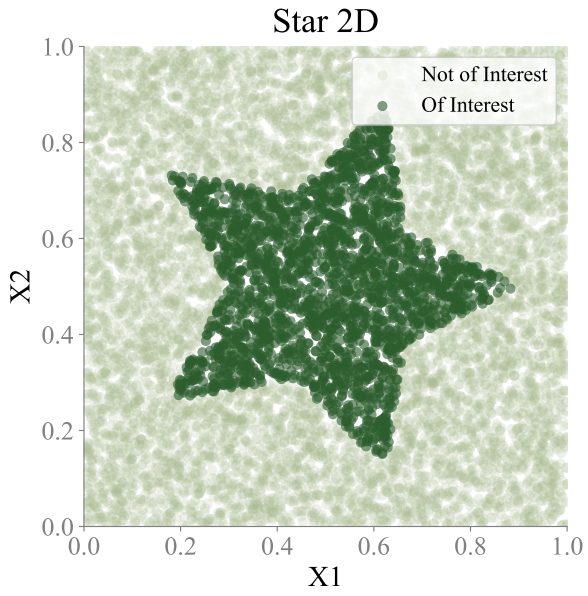


(c)

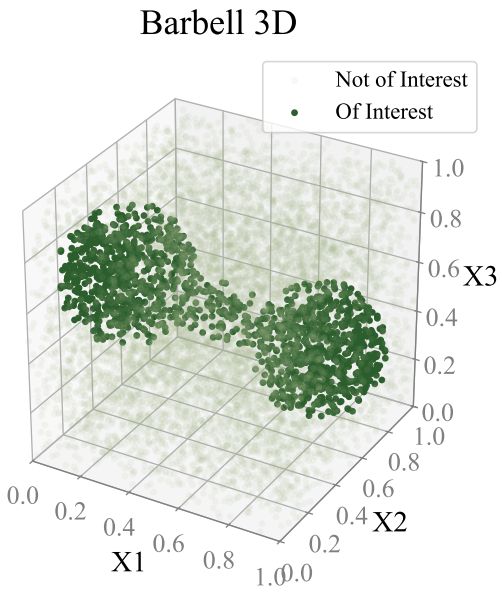


(d)

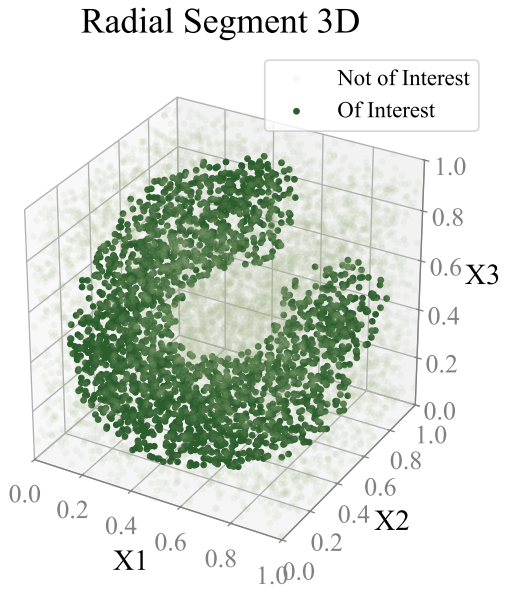
Figure 3.1 (continued)



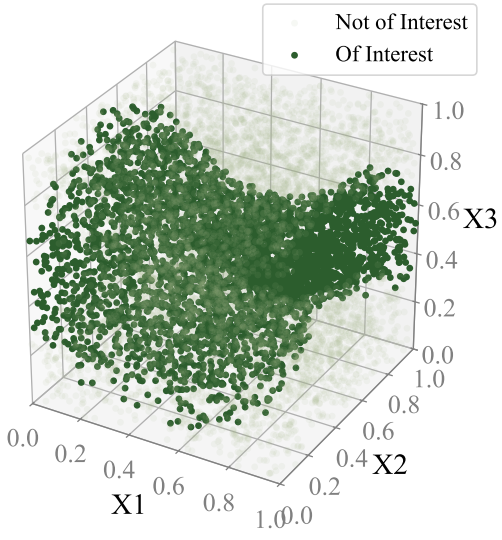
(e)



(f)



(g)



(h)

Table 3.1
Geometric properties of the synthetic shapes used in Phase 1. A checkmark indicates the presence of a property that is expected to challenge axis-aligned or globally rotated decision tree methods. Properties include axis misalignment (rotated boundaries), directional misalignment (locally varying orientations), and nonlinearity. The final column indicates which established methods were expected to perform poorly on each shape.

Shape	Axis Misalignment	Directional Misalignment	Nonlinearity	Likely to Struggle
2D Rectangle	✓			PRIM, CART
2D Barbell	✓	✓	✓	(PCA-)PRIM, CART
2D Radial Segment	✓	✓	✓	(PCA-)PRIM, CART
2D Sine Wave	✓	✓	✓	(PCA-)PRIM, CART
2D Star	✓	✓		(PCA-)PRIM, CART
3D Barbell	✓	✓	✓	(PCA-)PRIM, CART
3D Radial Segment	✓	✓	✓	(PCA-)PRIM, CART
3D Saddle	✓	✓	✓	(PCA-)PRIM, CART

3.1.2 Boundary and Dimensional Noise Design

In addition to the structural complexity of the benchmark shapes themselves, two forms of noise were introduced to test algorithmic robustness under more realistic data conditions: boundary noise and dimensional noise. Each type was designed to simulate distinct challenges commonly found in scenario discovery applications, allowing for a more rigorous evaluation of the candidate methods.

Boundary Noise

A key challenge in scenario discovery is that the decision boundaries produced by real-world models are often *noisy*. This characteristic, visually evident in studies by Gerst et al. (2013) and Kwakkel (2017), arises from three primary forms of model behaviour. First, many simulation models contain stochastic elements, meaning the same set of inputs can lead to different outcomes across separate runs. Second, the highly non-linear relationships between inputs and outputs can create complex and unpredictable boundaries. Third, complex models are often highly sensitive to critical thresholds or *tipping points*, where a minor variation in an input can trigger a major shift in system behaviour. As a result of these effects, the boundary separating one class of outcomes from another is rarely a clean line, but is instead a zone where successful and unsuccessful outcomes are mixed.

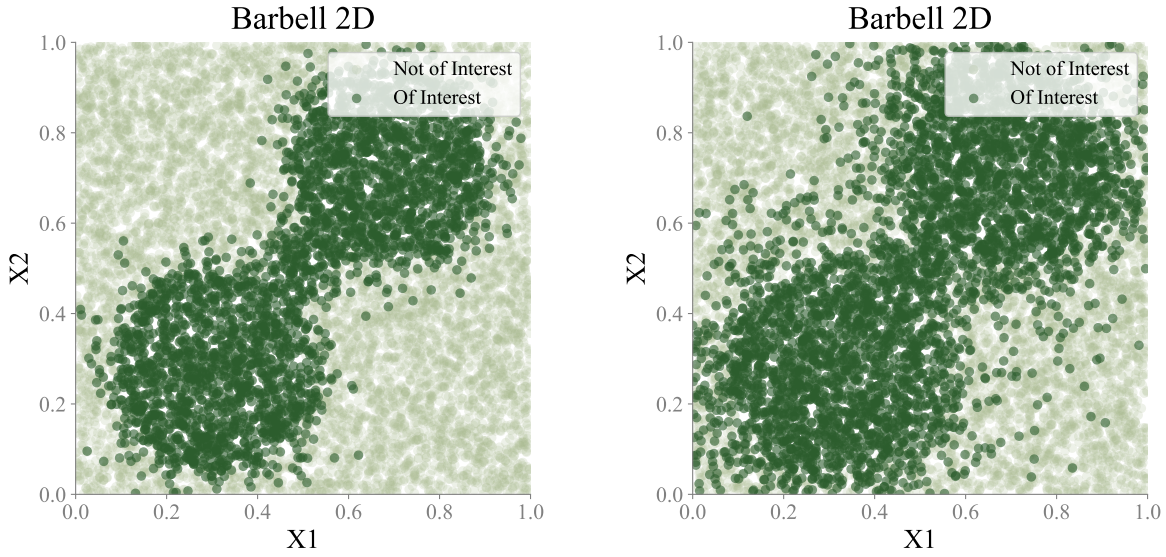
While this phenomenon is well-recognized in applied studies, it has not been systematically incorporated into the synthetic benchmarks used for evaluating scenario discovery algorithms. To create a more realistic and rigorous test environment, this research therefore introduces a formal mechanism for emulating this zone of mixed outcomes. Boundary noise is added to each synthetic shape by probabilistically re-labeling points just outside the true decision region from class 0 to class 1. The likelihood of a point being re-labeled is designed to decay exponentially with its Euclidean distance from the true boundary, a rate governed by the parameter $\lambda \in [0,1]$. Higher values of λ produce a wider and more ambiguous boundary zone, as shown in Figure 3.2. The full implementation details are provided in Appendix B.

Dimensional Noise

Following the precedent of earlier benchmarking studies (Dalal et al., 2013; Lempert et al., 2008; van Droffelaar, 2020), the second form of noise introduced was dimensional noise. This simulates another key challenge of real-world scenario discovery: high-dimensional input spaces that contain many uninformative variables. This test was designed to evaluate an algorithm's robustness and its ability to identify the truly influential variables and maintain performance when confronted with irrelevant inputs that have no causal relationship with the outcome.

To implement this, a set of irrelevant features was added to each dataset after the initial class labeling was complete. Each of these new features was generated by sampling independently from a uniform distribution over $[0, 1]$, ensuring it was statistically independent of the class label.

Figure 3.2
Visual effect of boundary noise on the 2-dimensional barbell shape. Panel (a) shows a dataset with low boundary noise ($\lambda = 0.03$), where only a few points near the decision boundary are relabelled. Panel (b) shows higher boundary noise ($\lambda = 0.07$), resulting in more points being relabelled.



(a) **(b)**

3.1.3 Selected Algorithms and Inclusion Criteria

Having established the design of the benchmark problems, this section introduces the six oblique decision tree algorithms that were selected for evaluation. The goal of the selection was not to conduct an exhaustive survey, but to assemble a methodologically diverse and representative set of algorithms. This allows for a focused comparison of how well different split induction strategies, including analytical, iterative, and combined methods, handle the challenges described previously.

The selection process was guided by three practical inclusion criteria:

- **Methodological diversity:** Algorithms were chosen to represent different strategies for constructing oblique splits, particularly in how they define the weight vector and threshold at each node.
- **Implementation accessibility:** Only models with available Python implementations were considered. This was essential for compatibility with the benchmarking framework, which required exporting trained trees into a shared, standardized format.
- **Structural simplicity:** The selection was limited to single-tree classifiers. Ensemble models, neural network methods, and other methods with non-transparent internal logic were excluded to increase interpretability required for scenario discovery.

Applying these criteria resulted in a final benchmark set of six algorithms, all adapted for this study from two open-source Python repositories. Four core candidates were drawn from the work of Majumder (2021), representing a variety of split-induction strategies: HHCART(A), HHCART(D), MOC1, and WODT. In addition, RandCART was included as a benchmark variant from the same source. The set was completed with a sixth algorithm, RidgeCART, adapted from the repository of Mao (2023).

The following sections proceed to describe the core mechanism of each of these six selected algorithms, grouped by their split optimisation strategies. A comparative summary of these methods is provided in Table 3.2.

Projection-Based Splitting Methods

A primary challenge in constructing oblique splits is the complexity of searching for an optimal hyperplane in a high-dimensional space. Several of the algorithms evaluated in this study – Householder Classification and Regression Tree (HHCART), Ridge Classification and Regression Tree (RidgeCART), and Randomised Classification and Regression Tree (RandCART) – address this by simplifying the problem: they first project the input data onto a single direction $w \in \mathbb{R}^d$, and then perform a simpler one-dimensional search for an optimal split point. All three methods use a CART-segmentor for this second step, which exhaustively evaluates all midpoints between the sorted, projected data points to find the threshold v that maximally reduces impurity. The key distinction between these algorithms, therefore, is how they determine the projection direction w, which reflects different assumptions about what structure in the data should guide the split.

HHCART(A) and HHCART(D)

The HHCART, introduced by Wickramarachchi et al. (2016), derives its projection directions from the internal geometric structure of the data. At each node, it analyses the principal axes of variation within each class by performing an eigendecomposition on the class-specific covariance matrices. The resulting eigenvectors are then used to construct Householder reflection matrices, which provide the candidate projection directions. To ensure robustness, the best oblique split found through this process is then compared to the best possible axis-aligned split, and the algorithm selects whichever of the two yields a greater reduction in impurity. Two variants of this algorithm were evaluated: the exhaustive HHCART(A), which tests reflections derived from all non-zero eigenvectors, and the more computationally efficient HHCART(D), which simplifies this search by using only the dominant eigenvector from each class. According to its authors, this simplification retains most of the performance benefits of the full approach, making it a potentially more practical choice for larger problems. The implementation used in this study was adapted from Majumder (2021), as detailed in Appendix C.

RidgeCART

In contrast to other methods, RidgeCART uses a supervised learning approach to generate candidate splits, based on the algorithm proposed by Mao (2023). While the original algorithm was designed for regression, it was adapted for the classification task in this thesis. At each decision node, a Ridge Regression model is fitted to predict the binary class labels. The learned coefficients ($w \in \mathbb{R}^d$) are used to create a new linear projection of the data. The algorithm then compares this projection in a direct competition with the original, axis-aligned features, and the single best split is chosen from this combined set. The inclusion of an L_2 regularisation term is a key feature that stabilises the learned projection, helping to avoid overfitting and improve robustness.

RandCART

To provide a simple baseline for comparison, RandCART uses a purely stochastic method to generate its projection directions, as described by Majumder (2021). This algorithm does not learn from the data. Instead, it generates a random orthonormal rotation matrix and applies it to the entire feature space. It then performs an exhaustive search for the best axis-aligned split across all axes of this new, randomly rotated coordinate system. The performance of this computationally efficient method is therefore dependent on chance, and its ability to find an effective split naturally decreases as the dimensionality of the input space increases.

Iterative and Optimised Methods

In contrast to projection-based methods, which reduce split selection to a one-dimensional thresholding problem, a second class of oblique decision trees constructs splits by directly exploring candidate hyperplanes. These methods do not rely on axis-aligned scans or input space transformations. Instead, they iteratively evaluate oblique splits by adjusting both the orientation and position of the decision boundary to minimise a split quality criterion. Two such algorithms were used in this study: a modified version of Oblique Classifier 1 (MOC1), which uses a structured greedy search over randomly perturbed hyperplanes, and the Weighted Oblique Decision Tree (WODT), which employs gradient-based optimisation of a smooth entropy-based impurity function.

Modified Oblique Classifier 1

The original Oblique Classifier 1 (OC1) algorithm, introduced by Murthy et al. (1994), performs greedy local search for oblique decision boundaries by iteratively perturbing the weights of a linear classifier. At each node, the method applies coordinate-wise updates to the weight vector, evaluates split candidates based on impurity, and uses probabilistic acceptance of non-improving moves to escape local minima. Random restarts from different initial hyperplanes are used to increase robustness, and the final decision rule is defined by the best-performing hyperplane across all trials.

The implementation used in this study, referred to as the Modified Oblique Classifier 1 (MOC1), preserves the overall structure of OC1 but introduces several adaptations to improve computational efficiency and simplify the search procedure. Most notably, MOC1 replaces coordinate-wise perturbations with full-vector directional perturbations sampled from a unit hypersphere. It performs a structured grid search over weight and bias perturbations, evaluating a discrete set of candidate splits per restart. Rather than using probabilistic acceptance or stochastic tie-breaking, MOC1 deterministically selects the globally best split identified across all perturbation trials and restarts.

These modifications aim to retain the core principle of perturbation-driven oblique split induction while reducing the computational overhead associated with fine-grained local search and probabilistic move selection. The design of MOC1 draws inspiration from the implementation by Majumder (2020), but reintroduces several foundational elements from Murthy's original algorithm, including multiple restarts and explicit bias term exploration. Full implementation details and justification for these changes are provided in Appendix C.

Weighted Oblique Decision Tree

The Weighted Oblique Decision Tree (WODT) algorithm, introduced by Yang et al. (2019), formulates the split selection problem as a continuous optimisation task over a differentiable objective function. Rather than selecting discrete thresholds or relying on axis-parallel initialisations, WODT optimises a weighted entropy function using gradient-based methods. This approach assigns soft probabilistic weights to instances based on a sigmoid-transformed linear score, thereby allowing a smooth partitioning of the data at each node.

The split parameters are jointly optimised via L-BFGS to minimise the objective function, which reflects the weighted sum of entropies in the left and right child nodes. After optimisation, the probabilistic split is hardened into a binary decision boundary defined by the learned hyperplane. Unlike most oblique tree methods, WODT requires no separate threshold scan or midpoint enumeration; both the orientation and position of the split are learned directly through optimisation.

The implementation used in this study was adapted from the version provided by Majumder (2020), which itself incorporated elements of the original WODT formulation by Yang et al. (2019). No substantial algorithmic modifications were made beyond integration into the benchmarking framework.

Split Evaluation Criteria

To ensure consistency in split selection across algorithms, a standard impurity-based criterion was applied throughout Phase 1. All evaluated methods, except for WODT, used Gini impurity to evaluate the quality of candidate splits. Originally introduced by Breiman et al. (1984) as part of the CART framework, the Gini index measures the probability that a randomly selected instance would be misclassified if it were labelled according to the node's class distribution (Bertsimas & Dunn, 2017). It ranges from 0 (perfect purity) to 1 (maximum impurity), and is minimised when child nodes are homogeneous in class composition.

Gini impurity was selected due to its widespread adoption in decision tree algorithms (Mienye & Jere, 2024) and its computational efficiency. Empirical comparisons have shown that alternative criteria, such as information gain, gain ratio, and chi-squared, yield similar predictive performance, with no consistent advantage across tasks (Mingers, 1989). As a result, using Gini was chosen for split evaluation.

The only exception is the WODT, which was retained in its original form. Unlike the other methods, WODT uses a continuous entropy-based loss function optimised through gradient descent (Yang et al., 2019). Substituting a discrete impurity measure like Gini would violate the assumptions of its optimisation procedure. For this reason, WODT was excluded from the standardisation to preserve its functionality.

Table 3.2
Comparison of evaluated decision tree algorithms by split formulation and optimisation strategy. Each method is categorised by the type of split it supports (axis-aligned, oblique, or both), the procedure used to determine the orientation of the split (i.e., the weight vector w), and the approach used to identify the optimal threshold (bias term v). The table also reports whether the method incorporates stochastic components during training, and lists the primary source or implementation reference.

Algorithm	Split Type	Weight Vector Search (w)	Bias Term Search (v)	Stochastic	Primary Source
HHCART(A)	Axis-aligned and oblique	**Analytical:** using eigenvectors from each class-specific covariance matrix to define candidate directions; each direction is reflected onto a coordinate axis using Householder transformation	**Analytical:** scanning thresholds via CART-style midpoints in each reflected space and the original space; the best split (oblique or axis-aligned) is selected	No	Adapted from Majumder (2021), based on Wickramarachchi et al. (2016)
HHCART(D)	Axis-aligned and oblique	**Analytical:** using the dominant eigenvector from each class-specific covariance matrix to define a single Householder reflection direction	**Analytical:** scanning thresholds via CART-style midpoints in each reflected space and the original space; the best split (oblique or axis-aligned) is selected	No	Adapted from Majumder (2021), based on Wickramarachchi et al. (2016)
RidgeCART	Axis-aligned and oblique	**Analytical:** computing direction using ridge regression; projection vector defines new axis for split search	**Analytical:** scanning thresholds using CART-style midpoints in the projected space and the original space; the best split is selected	No	Mao (2023)
RandCART	Oblique	**Analytical:** sampling random orthogonal rotation matrices via QR decomposition; projection axis selected within rotated space	**Analytical:** scanning thresholds using CART-style midpoints along rotated axes	Yes (random rotation)	Majumder (2020, 2021)
MOC1	Oblique	**Iterative:** multiple random restarts with full-vector perturbations; for each candidate direction, impurity is evaluated after bias shifting	**Iterative:** bias term shifted along hyperplane normal for each perturbed direction; best $(\mathbf{w}, v)$ pair selected per node	Yes (random restarts and perturbation of vector)	Adapted from Majumder (2021), with inspiration from Murthy et al. (1994)
WODT	Oblique	**Iterative:** computing direction using numerical optimisation of a smooth entropy-based classification loss	Jointly optimised with weight vector via smooth entropy-based classification loss	Yes (random initialisation of weights)	Adapted from Majumder (2021), based on Yang et al. (2019)

3.1.4 Two-Stage Experimental Design

Having introduced the benchmark shapes, noise conditions, and algorithmic candidates, this section explains how these elements were combined in a two-stage experimental design to determine the most suitable oblique decision tree for scenario discovery. The overarching goal was to identify a method that could not only handle diverse geometric challenges but also remain robust under the complex data conditions typical of real-world policy models.

Stage 1: Benchmarking Across Shapes and Boundary Noise

The first stage aimed to identify algorithms capable of producing high-quality, interpretable scenarios across the set of stylised benchmark problems. By testing the candidate algorithms on a wide variety of geometric shapes and noise conditions, this stage was designed to probe for generalisability. The goal was to identify methods that performed well consistently, rather than those that might excel on only a single type of problem.

Evaluation focused on three standard scenario discovery criteria: coverage, density, and interpretability. Coverage indicates how much of the outcome class is captured, while density measures how selectively this is achieved. Interpretability was operationalised using tree depth, which influences the number and complexity of subspaces in the resulting decision tree. Preference was given to algorithms that achieved strong coverage and density at shallow depths, since deeper trees typically introduce fragmentation and reduce interpretability. For example, a depth 3 tree can produce up to 8 subspaces, while a depth 5 tree may generate up to 32. Such complexity can obscure scenario logic and make results harder to communicate.

To implement this evaluation, the six candidate algorithms, along with standard CART as an axis-aligned baseline, were applied to all eight benchmark shapes. The four levels of boundary noise ($\lambda \in \{0.00, 0.03, 0.05, 0.07\}$ were chosen to meet several criteria. The number of levels was selected to balance a sufficient coverage of uncertainty with the practical need to contain the total number of experimental runs. The values themselves were chosen to span a modest but meaningful range, from a clean baseline ($\lambda = 0.00$ to a zone of moderate ambiguity. Higher levels of noise were deliberately avoided, as they risked erasing the underlying geometric logic of the shapes.

Trees were trained from depths 1 to 8 to observe the full performance trajectory. To account for stochasticity, each run was repeated five times for non-deterministic algorithms, while deterministic models were run once. Hyperparameters were held constant across all experiments to ensure a consistent basis for comparison (see Parameter Settings in Appendix C). In addition to scenario quality, computational runtime was tracked to assess the tractability of each method.

Based on this evaluation, two algorithms, MOC1 and HHCART(D), were selected for further testing in Stage 2. These methods consistently delivered similarly high scenario quality at low depths while maintaining tractable runtimes across benchmark conditions.

Stage 2: Robustness to Dimensional Noise and Sample Size

The second stage submitted the top-performing candidates from Stage 1, MOC1 and HHCART(D), to a series of stress tests designed to evaluate their robustness. The primary goal was to determine if their high performance would degrade when confronted with more realistic data complexities. Specifically, this phase assessed their performance persistence against two common challenges in scenario discovery: increasing sample size and the presence of high-dimensional data with irrelevant features (dimensional noise).

This two-stage design provides a rigorous basis for algorithm selection. The process was designed to be potentially iterative; had the top candidates shown significant performance degradation, the next-best algorithms from Stage 1 would have been subjected to the same robustness tests. By confirming that an algorithm with the best initial performance (Stage 1) also shows no significant degradation under these more complex conditions (Stage 2), it can be confidently selected as the most suitable candidate. This provides strong evidence of its superiority, as there is no reason to expect that other candidates, which started from a lower performance baseline, would outperform it under these more challenging conditions.

Experiments focused exclusively on the 3D Radial Segment. This shape was selected because it is the most structurally demanding of the benchmark problems, featuring disconnected subregions, combined with all three identified challenges: axis misalignment, directional misalignement and nonlinearity. Its

three-dimensional form also adds complexity beyond the 2D shapes used in Stage 1. To isolate the effects of dataset size and dimensionality, boundary noise was fixed at $\lambda = 0.03$.

Two stress tests were applied independently to evaluate robustness to sample size and dimensional noise. First, sample size was varied across six levels $(1000, 2000, 3000, 5000, 7500, 10000)$ to assess whether performance degraded with smaller datasets or runtime grew excessively with larger ones. Second, dimensional noise was introduced by appending $k \in \{0, 2, 7, 12, 17, 22, 27\}$ irrelevant features. This helped test each method's ability to ignore uninformative inputs. MOC1 was executed three times for each combination using fixed random seeds to account for stochastic variability, while HHCART(D), being deterministic, was run once per configuration.

Performance was assessed by tracking how coverage and density evolved across tree depths as sample size decreased or the number of irrelevant features increased. This allowed evaluation of each algorithm's robustness to reduced data availability and growing input dimensionality. The central criterion was whether high-quality scenario regions, defined by high coverage and density, could still be identified without requiring deeper trees.

In parallel, two additional metrics were recorded to assess tractability and interpretability: runtime and sparsity. Runtime was monitored to examine how computational costs scaled with larger datasets or higher-dimensional input spaces. Sparsity was evaluated to provide further insight into interpretability. A lower number of features per split corresponds to simpler decision boundaries and greater transparency in the resulting scenarios. To quantify this, the average number of features used per split was recorded for each tree depth.

3.1.5 Evaluation Criteria

As highlighted in Section 3.1.4, algorithmic performance was assessed primarily through coverage, density, and tree depth. Additionaly runtime was included to test tractability, and the average number of features per split were added to test sparsity. This section formalises these metrics.

1. **Coverage:** High coverage indicates the model successfully includes *cases of interest* within its selected subspaces. It is defined as the proportion of *cases of interest* correctly predicted as such by the model (see Section 2.2.1). If the model labelled none of the identified regions as relevant, coverage was assigned a value of 0.0.
2. **Density:** High density reflects the model's ability to concentrate *cases of interest* in areas identified as of interest, avoiding irrelevant cases. It is defined as the proportion of cases labeled as relevant that were in fact *cases of interest* (see Section 2.2.1). When the model labelled none of the identified regions as relevant, density was assigned a value of 0.0.
3. **Tree Depth:** Used as a proxy for interpretability. Since all tested algorithms used full-dimensional oblique splits, sparsity was not expected to be a distinguishing factor. Therefore, lower tree depth was preferred as an indicator of tree interpretability. Tree depth starts from 0 at the root node.
4. **Average Features per Split:** The average number of non-zero weights in the split vector for a tree of depth x.
5. **Runtime:** Total time to fit the model.

Accuracy and F-score were excluded from the primary evaluation to avoid redundancy. Accuracy displayed negligible variation across algorithms and therefore lacked discriminative power (see Appendix D). The F-score, being a harmonic mean of coverage and density, was omitted to avoid conflation of these two critical metrics.

3.1.6 Benchmark Execution and Post-Processing

A modular benchmarking pipeline was developed to ensure consistent, reproducible evaluation of all algorithm-shape combinations. The system automated model training, metric computation, and result logging, with experiments executed in parallel on DelftBlue. Post-processing and visualisation were performed separately using the saved metric files, enabling separate analysis.

Python Execution and Model Evaluation

The benchmarking pipeline was implemented as a modular Python script that executed each experiment in a consistent and reproducible manner. For every configuration, defined by the selected algorithm, benchmark shape, maximum depth, random seed, and output settings, the script trained the specified oblique decision tree classifier on the synthetic dataset, with runtime measured using Python's performance counter. After training, the model was converted into a standardised internal format that abstracted away implementation-specific details while preserving all relevant split and leaf attributes. This allowed uniform downstream evaluation across algorithms. Scenario discovery metrics, such as coverage and density, were then computed and saved alongside structural properties such as tree depth and feature usage. All outputs, including metrics and metadata, were saved to structured `.csv` files, with the converted trees optionally stored as `.pkl` files for post hoc analysis. Reproducibility was ensured through explicit seed control, with stochastic algorihtms refactored to take seeds.

Parallel Execution on DelftBlue

All experiments were executed in parallel on the DelftBlue high-performance computing cluster using SLURM job arrays. A Python script generated the full set of configurations by enumerating all combinations of algorithm, dataset, tree depth, random seed, and noise level relevant to each experimental stage. These configurations were stored as individual lines in a job list file, which was indexed using the SLURM array task ID to retrieve and run each configuration in isolation. This setup enabled efficient distribution of runs across CPU cores and ensured that each execution was independently reproducible. For each run, a single `.csv` file was written to a shared output directory, containing the full set of evaluation metrics and configuration metadata. Filenames were automatically constructed to encode key details such as algorithm name, depth, and seed. SLURM logs were saved separately to support diagnostics and debugging.

Result Aggregation and Visualisation

All evaluation and visualisation steps were based entirely on stored metric outputs, enabling a fully decoupled, reproducible post-processing workflow that did not require access to raw model objects. After completing all runs on DelftBlue, the resulting `.csv` files were downloaded and concatenated into a unified results tables using a post-processing script. These tables served as the input for all subsequent analysis and plotting. Plots primarily traced how coverage, density, accuracy, and runtime evolved with tree depth, and were grouped by experimental dimensions such as shape type, boundary noise, sample size, and feature count. Additional plots summarised runtime and feature usage patterns across dimensionality settings.

3.2 Phase 2: Comparison on Benchmark Shapes

Phase 2 investigated how HHCART(D), the algorithm selected in Phase 1, structures scenario partitions on two-dimensional benchmark problems, and how its performance on coverage, density, and interpretability compares to established scenario discovery methods. This phase served both to deepen understanding of HHCART(D)s partitioning behaviour and to assess whether its promising coverage–density results could be achieved without compromising interpretability.

The analysis proceeded in two parts. First, the impact of regularisation parameters, specifically, minimum purity and minimum mass, on HHCART(D)s partition complexity was examined. Second, the behaviour of the algorithm was compared to three established scenario discovery methods: PRIM, PCA-PRIM, and CART.

Interpretability in this phase was assessed using two indicators: the number of subspaces classified as class 1 and the structural complexity of the splits. The former reflects the fragmentation of the scenario space, while the latter distinguishes between axis-aligned and oblique splits, with the latter being more complex to interpret. This focus marks a shift from relying solely on tree depth or feature sparsity. This is because the regularisation applied in this section decouples tree depth from subspace count by removing unnecessary splits.

3.2.1 Regularisation Effects on HHCART(D) Partition Structure

The first part of the Phase 2 analysis examined whether structural regularisation could improve the interpretability of partitions generated by HHCART(D). Early results on the two-dimensional benchmark problems revealed that the algorithm's default configuration often produced highly fragmented solutions: numerous small subspaces were created, and nodes with high class purity were repeatedly split (see Section 5.1.1). This over-partitioning increased the number of regions classified as class 1 without yielding any strong gains in coverage or density, thereby undermining interpretability. To mitigate this effect, two forms of pre-split constraint were tested: a *minimum mass threshold* and a *minimum purity threshold*.

The minimum mass threshold specifies the minimum number of samples required in a node to allow further splitting, expressed either as an absolute count or as a fraction of the full dataset.[1] Increasing this threshold reduced the creation of small subspaces only at a small loss to coverage and density. However, it did not prevent the algorithm from continuing to split already homogeneous nodes. To address this, a second constraint was introduced: the minimum purity threshold. This parameter blocks further splitting when the dominant class already exceeds a specified proportion of the node, effectively halting tree growth in already well-classified regions. Unlike minimum mass, which regulates partition size, this constraint targets unnecessary complexity arising from overfitting to noise or marginal gains.

To explore the effect of these constraints they were tested on benchmark shapes with noise levels of $\lambda = 0.05$. Minimum purity thresholds of 90% and 100%, and minimum mass thresholds of 1% and 5% were tested. These tests were not intended to find optimal parameter settings, as this is expected to differ per case, but to provide qualitative insight into how the two constraints influence tree complexity, coverage, and density. Results are presented in Section 5.1.

3.2.2 Comparison of HHCART(D) and Established Scenario Discovery Methods

The second part of Phase 2 evaluated HHCART(D)s partitioning behaviour in direct comparison to the three established scenario discovery methods. While Phase 1 established that HHCART(D) can achieve promising results on coverage and density at low tree dephts relative to other oblique decision trees, it remained essential to assess whether the created subspaces by this algorithm could be intepreted.

The comparison used two-dimensional benchmark problems to enable direct visual assessment of scenario partitions. Two shapes were selected to represent contrasting partitioning challenges. The rotated rectangle was chosen because there is a simple 4-split solution that can reflect the underlying shape. The rotated barbell was used because it introduces a more fragmented and harder to approximate target concept, combining two disjoint circular regions linked by a narrow connecting segment.

The comparative evaluation focused on two aspects of partition structure. First, the trade-off between coverage and density was analysed to determine whether methods achieved high performance on both metrics simultaneously or required compromises. Second, the interpretability of the resulting partitions was evaluated in terms of split structure (axis-aligned or oblique) and number of subspaces classified as class 1. The latter directly corresponds to the number of scenario boxes identified in PRIM and PCA-PRIM.

The parameter settings for this comparison were as follows. Minimum mass was set at 5% across all methods; other parameters were varied as appropriate to each method. For HHCART(D), minimum purity was tuned per shape based on visual inspection of the resulting partitions and coverage–density trajectories. PRIM and PCA-PRIM used a peeling and pasting fraction of 5%, a density threshold of 0.8, and the Guivarch update function.[2] Scenario boxes for PRIM and PCA-PRIM were selected manually from the peeling trajectory based on coverage–density trade-offs. CART did not require any additional configuration beyond the minimum mass.

This comparative analysis provided a critical assessment of whether HHCART(D) can deliver interpretable scenario partitions while maintaining competitive coverage and density.

[1] Unlike the Exploratory Modeling and Analysis (EMA) Workbench implementations of PRIM and CART, which enforce the minimum mass on the resulting box or branches, HHCART(D) applies the threshold to the parent node before splitting.

[2] This modification avoids overlap between successive boxes by marking covered points as no longer of interest rather than removing them entirely. This allows PRIM and PCA-PRIM to produce disjoint partitions similar to those generated by the decision trees, improving comparability.

3.2.3 Visualisation Framework for HHCART(D)

To support this analysis, a dedicated Python class, `HHCartD`, was developed as a scenario discovery-focused implementation of the HHCART(D) algorithm. The class enables systematic training of oblique tree algorithms up to a specified maximum depth, and retains pruned versions at all depths to support depth-wise evaluation. It standardises the calculation and storage of scenario discovery metrics across depths, including coverage, density, and accuracy. It also provides an integrated set of visualisation tools for detailed inspection of tree structure and partition geometry.

Each trained `HHCartD` model is saved as a self-contained output package, comprising the trained tree objects, input data, full depth-wise metrics, and run metadata. This ensures complete reproducibility and comparability across all Phase 2 experiments. Internally, the tree object stores the exact split structure at every trained depth, allowing depth-specific retrieval, evaluation, and visualisation.

Specifically, the following visualisations are enabled by the package:

- **Decision boundary plots**: Oblique split boundaries in 2-dimensional input space shown at each tree depth.
- **Decision region plots**: The 2-dimensional input space with the classification of the regions based on the boundaries at the respective tree depth.
- **Coverage-density trade-off plots**: Coverage and density trajectories across tree depths or number of subspaces classified as 1.
- **Tree structure plots**: Node-based diagrams including oblique split equations, sample sizes, and class predictions.
- **Node size plots**: Sample count per node at each tree depth.
- **Scenario performance plots**: Accuracy, coverage, and density trajectories across tree depths or number of subspaces classified as class 1.
- **Split construction plots**: Visualisations for a selected node showing the data subset, the dominant eigenvector of the chosen class, the rotation applied by HHCART(D), the axis-aligned split in the rotated space, and the resulting oblique split mapped back to the original coordinates.

Both the regularisation analysis and the comparative evaluation in Phase 2 used this unified `HHCartD` implementation. As such, all results presented reflect consistent training, pruning, evaluation, and visualisation logic, ensuring that comparisons across regularisation settings and against established scenario discovery methods are fully standardised.

3.2.4 Visualisation Framework for Established Methods

A dedicated visualisation framework was developed to enable side-by-side comparison of two-dimensional scenario partitions generated by PRIM, PCA-PRIM, and CART. Box selection for PRIM and PCA-PRIM continued to use the standard functionality of the Exploratory Modeling and Analysis (EMA) Workbench. However, to support the comparative analysis in Phase 2, it was necessary to implement an additional visualisation tool that could display multiple boxes per method in a common coordinate space, report their coverage and density, and show the corresponding decision rules.

3.3 Phase 3: Comparison on Policy Model

The final phase of the research assesses the practical utility of the oblique decision tree approach in a realistic, high-dimensional policy context. Having been selected as the most robust candidate from the initial benchmarking, HHCART(D) is again compared against the established scenario discovery methods. The primary objective is to analyse its relative strengths and weaknesses regarding coverage, density, and, crucially, interpretability. By applying the different methods to the same complex case study, this phase provides a clear, evidence-based assessment of whether the theoretical flexibility offered by oblique trees translates into a practical advantage for policy analysis under deep uncertainty.

3.3.1 Case Study: The Energy Transition in the European Union

To provide a challenging and realistic test case, this research employs a well-established System Dynamics model of the energy transition within the European Union (EU) (Forrester, 1997; Loonen et al., 2013). The analysis uses the specific model configuration from Hamarat et al. (2014), which adapts the foundational model by Loonen et al. (2013) to operate under the Emissions Trading Scheme (ETS) policy. This model was chosen because it embodies the characteristics of a complex system under deep uncertainty; its structure contains the systemic interactions, delays, and feedback loops that produce the non-linear and path-dependent behaviour that challenges established scenario discovery methods (Hamarat, Kwakkel, & Pruyt, 2013; Loonen et al., 2013). The model's extensive use in prior research confirms its status as a vetted and appropriate case study for testing novel analytical approaches (Hamarat et al., 2014; Kwakkel, 2019; Kwakkel & Cunningham, 2016; Kwakkel & Jaxa-Rozen, 2016; Loonen et al., 2013).

The model is comprehensive in scope, representing the EUs power sector as seven interconnected geographical regions and simulating the market competition between nine distinct power generation technologies, from conventional sources to renewables (Loonen et al., 2013). Implemented in Vensim, it is composed of 33 ordinary differential equations and hundreds of auxiliary variables that capture the system's core dynamics (Hamarat et al., 2014). The system's behaviour is explored across 46 uncertain factors that represent technological, economic, and policy uncertainties. These uncertainties are a mix of continuous, integer, and categorical variables, creating a complex, high dimensional, and heterogeneous input space for the analysis (Hamarat et al., 2014). Table 3.3 summarises the main categories of these uncertainties.

Table 3.3
Specification of the uncertainties explored, taken from Hamarat et al. (2014).

Name	Description
Economic lifetime	For each technology, the average lifetimes are not known precisely. Different ranges for the economic lifetimes are explored for each technology.
Learning curve	It is uncertain for different technologies how much costs will decrease with increasing experience. Different progress ratios are explored for each technology.
Economic growth	It is deeply uncertain how the economy will develop over time. Six possible developments of economic growth behaviours are considered.
Electrification rate	The rate of electrification of the economy is explored by means of six different electrification trends.
Physical limits	The effect of physical limits on the penetration rate of a technology is unknown. Two different behaviours are considered.
Preference weights	Investor perspectives on technology investments are treated as being deeply uncertain. Growth potential, technological familiarity, marginal investment costs, and carbon abatement are possible decision criteria.
Battery storage	For wind and PV solar, the availability of (battery) storage is difficult to predict. A parametric range is explored for this uncertainty.
Time of nuclear ban	A forced ban for nuclear energy in many EU countries is expected between 2013 and 2050. The time of the nuclear ban is varied between 2013 and 2050.
Price–demand elasticity	A parametric range is considered for price–demand elasticity factors.

3.3.2 Application of Competing Algorithms

The application of the four competing algorithms began with a common data preparation stage. First, an ensemble of simulation runs was generated, after which the outputs were reclassified into binary outcomes to define the *outcomes of interest* for the analysis (Section **??**). Following this preparation, the methodological approach diverged. The established methods were suitable for direct application to the full 46-dimensional uncertainty space. In contrast, the HHCART(D) algorithm, which does not inherently produce sparse rules, required a two-step process of feature selection and subsequent model fitting to ensure its results would be interpretable and splits would not include all 46 input variables.

Established Methods

The application of the established methods required consideration of two primary methodological challenges: ensuring a consistent analytical baseline for all algorithms, and managing the unique operational characteristics of each approach. To establish a consistent baseline, all three methods were implemented via the EMA Workbench and applied to the full 46-dimensional dataset. A common minimum mass constraint of 5% was enforced across all algorithms. Furthermore, to generate comparable sets of non-overlapping scenarios from the peeling-based methods, both PRIM and PCA-PRIM employed the update procedure suggested by Guivarch et al. (2016).

With this common framework in place, method-specific characteristics were then addressed. For PRIM, the challenge of handling the dataset's heterogeneous uncertainties was managed by using the EMA Workbench's lenient objective function, which prevents biases toward categorical variables (Kwakkel & Jaxa-Rozen, 2016). For PCA-PRIM, the key consideration was preserving the integrity of categorical variables during dimensionality reduction, which was achieved by excluding them from the PCA rotation step (Kwakkel et al., 2013). Finally, for CART, its known tendency for over-fragmentation (Lempert et al., 2008) was managed with a custom post-processing framework that automatically pruned the tree by merging similar leaf nodes and enabled clear visualisation of the results. Furthermore, the EMA workbench implementation of CART automatically performs internal dummy encoding for categorical variables, allowing it to handle heterogeneous inputs without requiring explicit preprocessing.

HHCART(D) with Feature Selection

In contrast to the direct application of the established methods, the HHCART(D) algorithm was implemented through a two-step process designed to enhance interpretability. The two steps were the following: feature selection using Global Sensitivity Analysis, followed by scenario discovery using selected subsets of features with HHCART(D).

Before both steps, all categorical variables were transformed using fixed one-hot (dummy) encoding. This was necessary because oblique decision trees require numeric inputs for linear combinations; unordered categorical variables lack inherent structure and cannot be directly incorporated without introducing incorrect relationships. Although HHCART supports categorical data via CRIMCOORD transformations (Loh & Shih, 1997; Wickramarachchi et al., 2016), these node-specific projections vary across the tree and lack global interpretability. Dummy encoding was therefore used to ensure increased interpretability of HHCART(D) outputs.

Feature Selection via Global Sensitivity Analysis

As established in the literature and confirmed by the algorithm's design, oblique decision trees like HHCART(D) do not inherently produce sparse decision rules. In high-dimensional policy applications, this can lead to complex and fragmented partitions that are difficult to interpret. To address this and enhance the interpretability of the final model, a feature selection step is required. Feature selection reduces the dimensionality of the input data by selecting a subset of the original variables and eliminating those that are not relevant (Guyon & Elisseeff, 2003). This process makes the resulting model more interpretable and the output data easier to analyse (Jaxa-Rozen & Kwakkel, 2018).

Given the model's complexity, as discussed in Section 3.3.1, *global sensitivity analysis* is the most appropriate methodological framework for identifying influential variables. Simpler approaches, such as *one-at-a-time sensitivity analysis*, are insufficient in this context because they assess variables only along isolated dimensions. As a result, they fail to capture interaction effects (Jaxa-Rozen & Kwakkel, 2018), which may produce misleading conclusions about variable influence. Global sensitivity analysis addresses this limitation by adopting a global perspective: it evaluates each variable's influence across the full joint distribution of all other parameters, allowing interaction effects and nonlinear contributions to be properly identified (Jaxa-Rozen & Kwakkel, 2018; Q. Liu & Homma, 2009).

While several global sensitivity analysis methods exist, there is often a trade-off between their computational cost and the information they provide (Jaxa-Rozen & Kwakkel, 2018). A key metric in this context is Sobol's total effect indices, which quantify a variable's influence through both its direct effects and its higher-order interactions (Saltelli et al., 2008; Sobol, 2001). However, the cost of calculating these indices can quickly become prohibitive for complex simulation models (Jaxa-Rozen & Kwakkel, 2018). Tree-based ensemble methods, and in particular the Extra-Trees algorithm, offer a practical and robust alternative. Empirical results show that Extra-Trees can accurately approximate the relative importance of these key

Sobol indices with a much smaller computational budget (Jaxa-Rozen & Kwakkel, 2018). Furthermore, these techniques perform well even with relatively small sample sizes, are effective for nonlinear problems where input interactions are significant, and can handle both numerical and categorical data (Jaxa-Rozen & Kwakkel, 2018; Louppe, 2014).

The Extra-Trees algorithm identifies influential variables by constructing an ensemble of fully grown decision trees that collectively capture the relationships between input variables and model outcomes (Geurts et al., 2006). Rather than fitting a single tree, the method builds many trees using randomisation at two levels: it selects a random subset of variables at each node and draws random split thresholds for these variables. Among these candidates, the algorithm selects the split that most effectively reduces node impurity, which is measured by the Gini index for classification tasks (Louppe, 2014). By averaging across the entire ensemble, Extra-Trees produces a robust estimate of each variable's contribution to reducing impurity, summarised through the Mean Decrease in Impurity score (Louppe, 2014). This approach mitigates the instability and bias of single decision trees and enables the efficient estimation of variable importance in high-dimensional, nonlinear, and interaction-driven models (Jaxa-Rozen & Kwakkel, 2018).

Variable selection in this study was performed using the Extra-Trees implementation available in the EMA Workbench (Kwakkel, 2017), to identify influential variables leading to high-renewable features. The algorithm was configured following the recommendations of Jaxa-Rozen and Kwakkel (2018). Specifically, 250 trees were used, with no maximum depth, and one-third of the available features considered at each split. The minimum number of samples required to split a node was set to 2, with at least 1 sample required per leaf. Bootstrapping and out-of-bag estimation were disabled to ensure alignment with the typical usage of Extra-Trees for sensitivity analysis (Jaxa-Rozen & Kwakkel, 2018). A fixed random seed of 42 ensured reproducibility. This configuration was designed to provide stable and interpretable variable rankings at a reasonable computational cost.

Scenario Discovery with HHCART(D)

Following the identification of the most influential uncertainties via Global Sensitivity Analysis, the HHCART(D) algorithm was applied to generate the final oblique scenarios. To explore the trade-off between scenario complexity and performance, an iterative approach was adopted. Using the same custom implementation of HHCART(D) as detailed in Section 3.2.3, separate models were trained on increasingly larger subsets of the top-ranked features. This process was conducted for the top two, top three, top four, and finally the top five most influential uncertainties. The choice for only going up to five features was made as more then five features in splits really deteriorates intepretability of created splits. For each of these iterative runs, the algorithm was regularised with a *minimum purity threshold* of 95%, a *minimum mass constraint* of 5%, and a *maximum depth* of 8. This iterative application allows for a direct comparison of the resulting scenarios not only against the established methods but also across different levels of feature complexity.

3.3.3 Evaluation Criteria

The final comparison between the established methods and the HHCART(D) approach is again based on the three primary objectives of scenario discovery used throughout this research: coverage, density, and interpretability. The quantitative metrics of coverage and density are used to assess the performance of the scenarios generated by each algorithm. Interpretability is again evaluated using two complementary diagnostics: the number of distinct subspaces classified as class 1 (also called regions), and the algebraic complexity of the split expressions used to define them. This dual perspective allows direct comparison between, for instance, a single, axis-aligned box produced by PRIM and a multi-region, oblique-rule-based structure generated by HHCART(D). The overall analysis of trade-offs across these three criteria serves to assess whether the geometric flexibility of oblique trees offers a meaningful advantage for policy-relevant scenario discovery.

4 Phase 1: Benchmarking Oblique Decision Tree Algorithms

This chapter presents the results of Phase 1, which aimed to identify the most suitable oblique decision tree algorithm for scenario discovery from a predefined set of candidates. The selection process was structured in two stages: the first benchmarked the algorithms on stylised synthetic problems to assess their ability to isolate decision-relevant regions under different forms of geometric complexity, while the second examined the top performers under added dimensional noise and increased sample size. The combined insights provided a robust basis for selecting a single algorithm for further comparison in Phases 2 and 3.

4.1 Stage 1: Benchmarking Across Shapes and Boundary Noise

Stage 1 systematically evaluated six oblique decision tree algorithms on synthetic benchmark problems designed to expose the limitations of axis-aligned methods. CART was included as a baseline, and RandCART served as a benchmark oblique model using random projections. Each algorithm was tested across eight geometric shapes and four levels of boundary noise. Performance was measured using coverage and density over tree depth, with training time recorded to assess tractability for scenario discovery. These results informed the selection of candidates for deeper analysis in Stage 2.

4.1.1 Performance Across Tree Depths

This section presents how algorithmic performance changes with increasing tree depth. The results focus on whether strong coverage and density are achieved at shallow depths, which is critical for interpretability. Coverage and density metrics are aggregated across all noise levels for both 2- and 3-dimensional benchmark shapes. Training time is also evaluated to compare the computational cost of each method as tree depth increases.

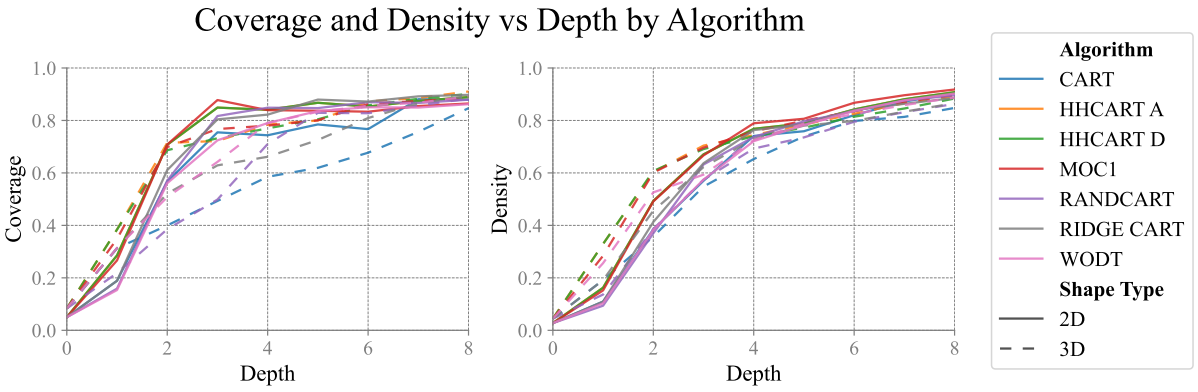
Coverage performance on the 2-dimensional benchmark shapes, aggregated over all boundary noise levels, shows that MOC1, HHCART(D), and HHCART(A) achieve similarly high coverage at lower tree depths, while outperforming the other algorithms (Figure 4.1a). By tree depth 3, these algorithms already exceed 90% coverage, clearly outperforming the rest. RandCART, RidgeCART, and CART follow at approximately 80%, while WODT remains below 70%. At depth 4, coverage levels increase across all algorithms, with the top three dipping slightly and the others approaching similar values near 85%. Beyond this point, coverage remains largely stable for each method, with only little further improvement as tree depth increases. CART, however, lags behind – remaining near 80% until depth 6 – demonstrating a slower and more gradual convergence. These trends suggest that MOC1, HHCART(D), and Householder Classification and Regression Tree using all eigenvectors of each class (HHCART(A)) identify decision-relevant regions more efficiently, reaching stronger coverage with shallower trees – an important advantage for scenario discovery as lower tree depths are expected to be easier to interpret.

Coverage performance on the 3-dimensional benchmark shapes, aggregated across all boundary noise levels, shows a general decline in coverage at low tree depths across all algorithms compared to the 2-dimensional case – reflecting the added challenge of higher-dimensional input spaces (Figure 4.1a). Despite this drop, MOC1, HHCART(D), and HHCART(A) continue to perform similarly to each other and retain their lead, achieving the highest coverage at low tree depths. At depth 3, MOC1, HHCART(D), and HHCART(A) already exceed 75% coverage, while RidgeCART and WODT follow at around 60%, and RandCART and CART lag further behind at approximately 55%. By depth 4, WODT catches up to the leading group, reaching around 80% coverage, whereas RidgeCART and RandCART only approach these levels between depths 5 and 6. CART remains the weakest performer throughout, plateauing near 80% by depth 8, while the top algorithms continue to improve toward 90%. Together, these results shows that the top-performing methods retain their advantage when moving from two to three dimensions.

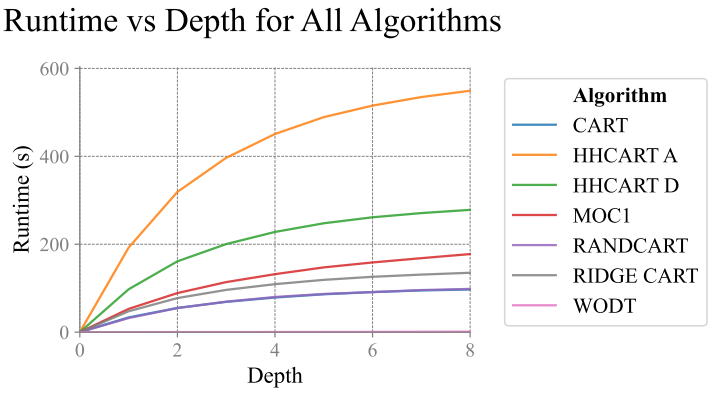
Following the analysis of coverage, attention now shifts to density. Performance on the 2- and 3-dimensional benchmark shapes, aggregated across all boundary noise levels, shows that MOC1, HHCART(D), and HHCART(A) consistently achieve higher density at shallow depths than the other algorithms (Figure 4.1a). Between depths 2 and 4, these three methods maintain a clear performance advantage, with density values approximately 10% higher than those of RandCART, RidgeCART, WODT, and CART. After depth 4, the gap between methods narrows as all algorithms continue refining their subspaces, though small differences persist. Taken together, these results confirm that MOC1, HHCART(D), and HHCART(A) consistently outperform the other methods in both coverage and density at low tree depths across 2- and 3-dimensional benchmark problems.

When looking at training time across tree depths substantial differences in computational efficiency between algorithms become apparent (Figure 4.1b). WODT and HHCART(A) stand out as runtime outliers in opposite directions: WODT maintains runtimes close to zero at all depths, while HHCART(A) exceeds 400 seconds at depth 4. The remaining methods – MOC1, HHCART(D), RidgeCART, and RandCART – all fall within a moderate range of 90 to 160 seconds at depth 4. All of these scale sublinearly with tree depth, reflecting the reduced computational cost of splits as node subsets shrink at deeper tree levels.

Figure 4.1
Performance trends across tree depth for all evaluated algorithms. Panel (a) shows coverage and density by tree depth (1–8), averaged across all benchmark shapes and boundary noise levels. Solid lines represent 2D shapes; dotted lines represent 3D shapes. Panel (b) presents the corresponding average training time by depth, aggregated across all shape types and boundary noise levels.



(a)



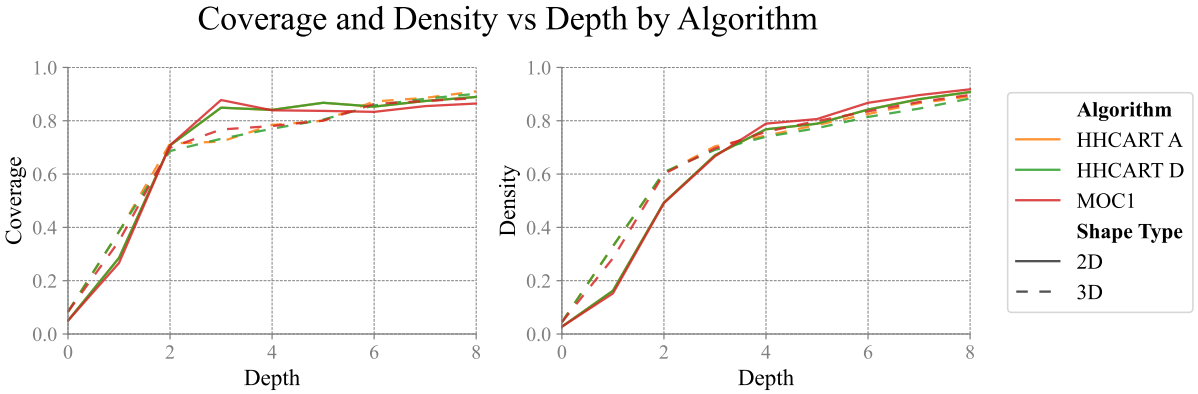
(b)

4.1.2 Candidate Selection

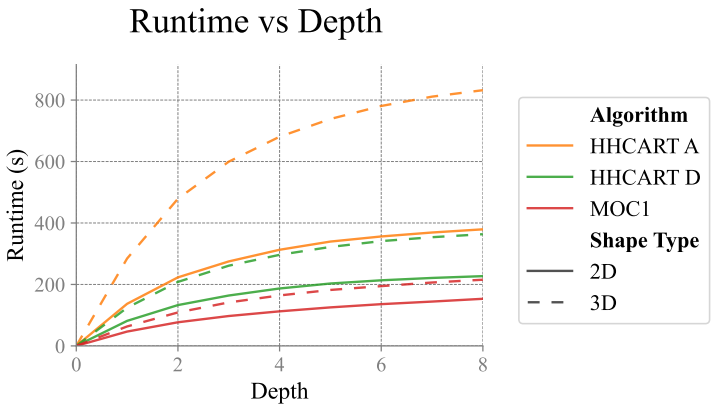
Based on the preceding results, four algorithms were excluded from further consideration. All four underperformed relative to MOC1, HHCART(D), and HHCART(A) when looking at coverage and density at low tree depths. RandCART and RidgeCART consistently required deeper trees to achieve comparable coverage and density, particularly on 3-dimensional benchmark shapes, limiting their interpretability. WODT, while computationally efficient, lagged in both coverage and density across depths. CART, included as a baseline, performed worst overall – especially on 3-dimensional shapes – highlighting the limitations of axis-aligned methods on these benchmark shapes.

To further compare the remaining candidates, MOC1, HHCART(D), and HHCART(A) were displayed separately from the previously excluded algorithms in Figures 4.2a and 4.2b. As the coverage and density curves in Figure 4.2a indicate, all three performed almost identical across both 2- and 3-dimensional benchmark shapes, providing no clear grounds for further exclusion based on these metrics. However, runtime analysis revealed a substantial difference in performance: as visible in Figure 4.2b HHCART(A) required roughly twice the training time of MOC1 and HHCART(D) on 2-dimensional shapes, and more than three times the training time on 3-dimensional shapes. Given the absence of performance gains and its significantly higher computational cost, HHCART(A) was excluded. MOC1 and HHCART(D) were therefore selected as the final candidates for continued evaluation under dimensional noise and samples size variation.

Figure 4.2
Performance comparison across tree depths (1–8) for MOC1, HHCART(D), and HHCART(A). Panel (a) shows coverage and density by depth, averaged across all benchmark shapes and boundary noise levels. Solid lines represent 2D shapes; dotted lines represent 3D shapes. Panel (b) presents the corresponding average training time by depth, aggregated across shape types and noise levels.



(a)



(b)

4.2 Stage 2: Robustness to Dimensional Noise and Sample Size

This section presents Stage 2 results, which evaluate the robustness of MOC1 and HHCART(D) to structural complexity. Building on their strong performance in Stage 1, both algorithms are tested under more realistic conditions by varying the number of dimensional noise features and the total sample size. Experiments were conducted on the 3D Radial Segment shape with fixed boundary noise ($\lambda = 0.03$).

4.2.1 Robustness to Increasing Dimensional Noise

To evaluate robustness to dimensional noise, the number of features was varied from 3 to 30 while holding all other conditions constant.

HHCART(D) maintained stable classification performance across the full range of tested dimensionalities, while MOC1 deteriorated significantly. As shown in Figure 4.3a, HHCART(D) sustained similar levels of both coverage and density from 3 to 30 input dimensions, with no consistent downward trend and only minor fluctuations across feature counts. This indicates that the algorithm's ability to identify decision-relevant subspaces was largely unaffected by the introduction of irrelevant variables. MOC1, by contrast, exhibited a sharp performance drop as dimensionality increased. Figure 4.3b reveals that coverage began to fall at 10 features and continued to decline steeply thereafter, while density followed a less pronounced, but still decrease in perfomance over all tree depths. These patterns suggest that MOC1s ability to isolate meaningful regions in the input space is increasingly disrupted by feature noise, even though the underlying classification task remains unchanged.

This difference in robustness reflects the inherent trade-offs in algorithm design. HHCART(D) applies a deterministic analytical approach at each node, using per-class covariance directions and axis-aligned thresholding in reflected and orignal spaces. As a result, its split construction is unaffected by the dimensionality of the feature space beyond a linear runtime cost. In contrast, MOC1 relies on an iterative perturbation-based search that evaluates many candidate hyperplanes through restarts, perturbations, and bias shifts. As the number of irrelevant features increases, the search space expands and becomes harder to explore effectively within a fixed computational budget. The observed drop in MOC1s performance is therefore not due to misclassification from noise per se, but rather due to the limited number of restarts, perturbations, and bias steps used during training, settings that may be insufficient to find good splits in high-dimensional spaces.

Runtime increased with the number of noise features for both algorithms, but the rate of growth diverged significantly. When plotted on a log–log scale, both MOC1 and HHCART(D) exhibited approximately linear trends, indicating power-law scaling (Figure 4.4). MOC1 showed sublinear growth with a scaling exponent of 0.80, meaning that doubling the number of features increased runtime by less than a factor of two. HHCART(D), by contrast, scaled almost linearly with an exponent of 0.98, suggesting a near one-to-one relationship between feature count and runtime. These differences are also reflected in the raw runtime values (Figure D.3, Appendix D): while runtimes are comparable at 3 and 5 dimensions, HHCART(D) becomes increasingly slower, requiring around 60% more training time than MOC1 at 30 dimensions. This divergence likely stems from implementation differences. MOC1 performs a fixed number of random restarts, perturbations, and bias shifts regardless of dimensionality, whereas HHCART(D) recomputes class-specific covariance matrices and reflections at each node, which becomes increasingly expensive in high-dimensional spaces.

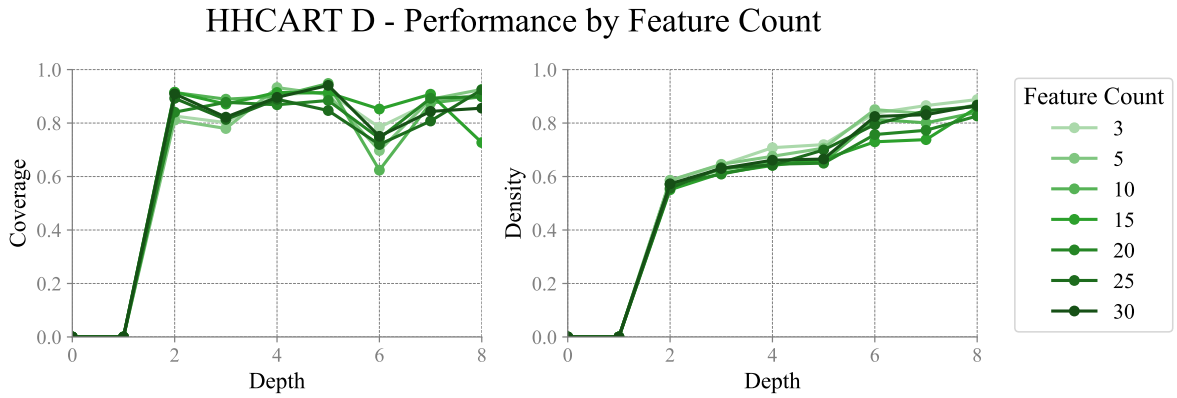
Neither algorithm systematically enforced sparsity, but their feature use did differ. MOC1 consistently produced dense decision rules, with each split involving all input features across all depths. HHCART(D) also generated predominantly dense rules, though occasional axis-aligned splits led to slightly lower average feature counts at higher depths (Figure D.4, Appendix D).

4.2.2 Robustness to Sample Size

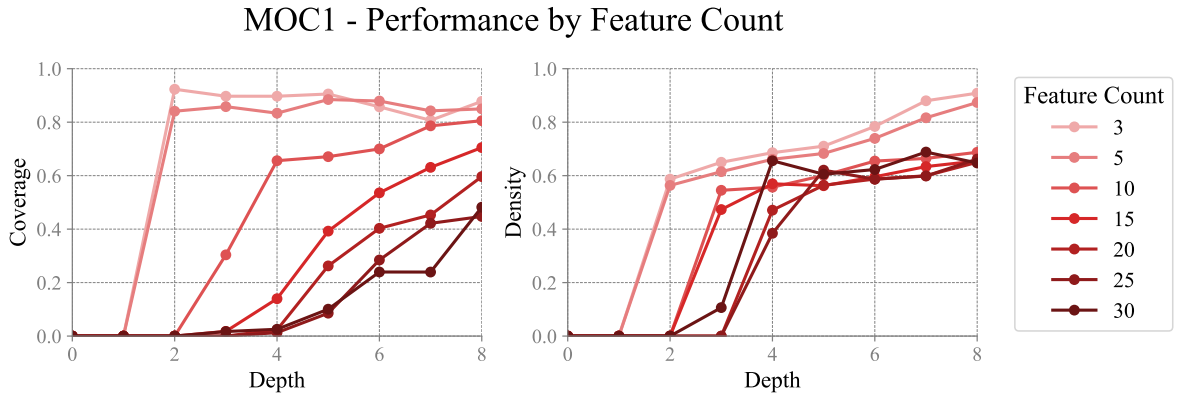
Following the evaluation of dimensional noise, the analysis next assessed how performance responded to changes in sample size. The number of observations was reduced from 10 000 to 1000, while the number of input features was held constant.

Coverage and density remained largely unaffected by sample size for both MOC1 and HHCART(D), suggesting that the subspace structures identified by the algorithms generalised well across data volume 1000 to 10 000 samples. As shown in Figure D.1 (Appendix D), both classifiers achieved consistent

Figure 4.3
Coverage and density per tree depth (1–8) on the 3D Radial Segment under increasing dimensional feature noise. Panel (a) shows results for HHCART(D), indicating consistent performance across dimensional noise levels. Panel (b) shows results for MOC1, with declining performance as dimensional noise increases.

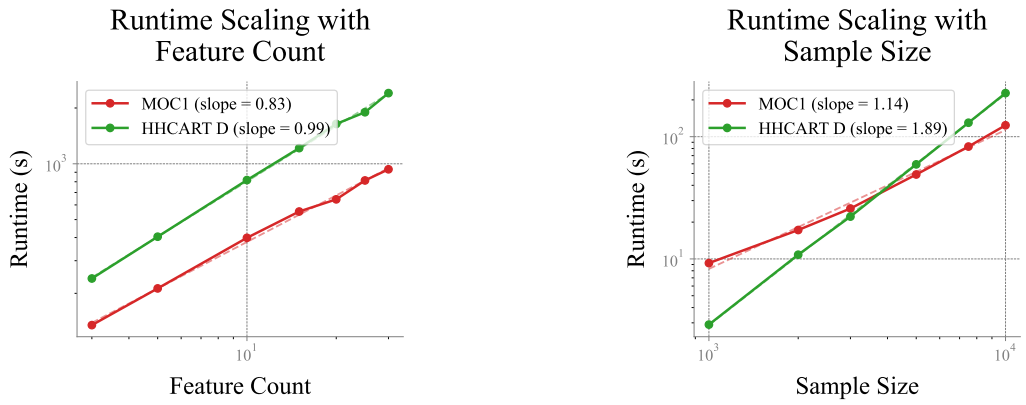


(a)



(b)

Figure 4.4
Log-log plots showing runtime scaling of MOC1 and HHCART(D). Dashed lines indicate fitted power-law trends. Panel (a) varies dimensional noise and shows sublinear scaling for both methods, with MOC1 (0.80) more efficient than HHCART(D) (0.98). Panel (b) varies sample size and shows superlinear scaling for both MOC1 (1.14) and HHCART(D) (1.88).



(a) **(b)**

coverage and density across depths, regardless of whether the dataset was small or large. This stability indicates not only that both methods are robust to sample size within the tested range for a 3-dimensional problem, but also that Latin Hypercube Sampling provided sufficient resolution to represent the relevant dimensions, even at 1000 samples.

Runtime increased with sample size for both algorithms, but the rate of growth diverged noticeably. On a log–log scale, both followed a power-law trend (Figure 4.4). MOC1 exhibited slightly superlinear scaling with an exponent of 1.14, while HHCART(D) scaled more steeply with an exponent of 1.88. This difference is also evident in the raw runtimes (Figure D.2, Appendix D): although HHCART(D) was slightly faster at smaller sample sizes, MOC1 became more efficient as data volume increased. At 10 000 samples, HHCART(D) required approximately 25% more training time than MOC1. This discrepancy can be attributed to structural differences between the two algorithms. HHCART(D) performs multiple threshold scans per node, one in the original space and others in reflected spaces, each of which involves sorting feature values and becomes increasingly expensive with more data. MOC1, by contrast, performs a fixed number of random perturbations and bias shifts at each node, and evaluates splits directly without sorting, which helps to limit its runtime growth as sample size increases.

4.2.3 Final Choice

The stress tests in Stage 2 forced a decisive trade-off between the two leading candidates, pitting the superior runtime scalability of MOC1 against the robustness of HHCART(D). While both algorithms demonstrated high performance on the initial geometric benchmarks, their equivalence ended there. MOC1, despite its efficiency, proved to be brittle; its performance on coverage and density deteriorated significantly when confronted with irrelevant features, revealing a critical weakness for any high-dimensional application.

HHCART(D), in stark contrast, proved to be exceptionally robust, maintaining high performance under the exact conditions where MOC1 faltered. Its ability to consistently identify high-quality scenarios remained stable even as the dimensionality of the input space increased tenfold. This resilience to feature noise is not merely a desirable trait but a fundamental requirement for a tool intended for scenario discovery, where models are often characterized by a large number variables of wich many may be weakly informative or redundant.

This stark divergence in performance stems from the core design philosophy of each algorithm. MOC1s iterative search relies on a fixed number of random perturbations to explore the solution space. As the feature space expands, this fixed search effort becomes increasingly insufficient, making it less likely to find an optimal split. HHCART(D)s deterministic, analytical approach, however, which uses the internal covariance structure of the data to guide its search for a split, is inherently more resilient to being diluted by noisy, high-dimensional inputs.

Therefore, the final selection is a direct and rigorous application of the two-part criterion established in the experimental design: high initial performance combined with demonstrated robustness. HHCART(D) was the only candidate to meet both conditions. It was a top performer in Stage 1 and proved its robustness by maintaining that performance throughout the stress tests of Stage 2. Given the primary goal of this phase, to find a method suitable for high-dimensional, structurally complex policy problems, this demonstrated robustness is non-negotiable. HHCART(D) was therefore selected for the comparative analyses in Phase 2 and 3.

5 Phase 2: Comparison on Benchmark Shapes

Building on the selection of HHCART(D) in Chapter 4, this chapter evaluates its performance against established scenario discovery methods on a subset of benchmark problems. To enable a meaningful comparison with the outputs of established methods, an initial analysis is conducted to explore the effect of stuctural regularisation on HHCART(D)s tendency for over-segmentation. The main comparative evaluation then assesses the algorithm against PRIM, PCA-PRIM, and CART on the rotated rectangle and barbell benchmarks. The focus lies on the trade-offs between coverage, density, and the interpretability of the resulting scenario partitions.

5.1 Regularisation of HHCART(D)

This section examines the impact of structural regularisation on the partitioning behaviour of HHCART(D). First, it illustrates the problem of over-segmentation that arises when the algorithm is unconstrained. Then it evaluates how parameters such as the *minimum purity* and *minimum mass* thresholds can be used to simplify the tree structure, and how this affects performance.

5.1.1 Over-Segmentation and the Effect of Boundary Noise

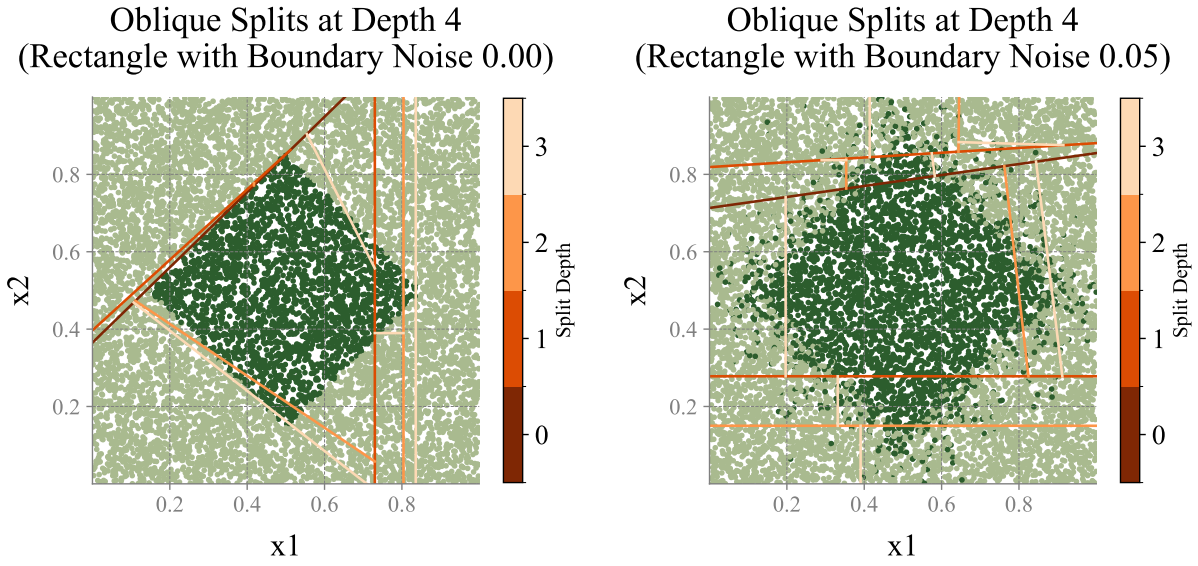
When trained without regularisation, HHCART(D) generates decision boundaries that are excessively fragmented and lack interpretability. This issue is starkly illustrated by the rotated rectangle benchmark. Figure 5.1a presents all splits up to depth 4 in the absence of boundary noise. Although the initial split at depth 0 aligns closely with one side of the rectangle, subsequent splits fail to respect the rectangle's edges in a similar manner. Instead of capturing the four intuitive partitions defining the rectangle, the model creates ten oblique splits within this shallow depth range. The result is a segmented space that does not align intuitively with the underlying geometric shape.

This fragmentation is also evident when looking at the number of subspaces classified as 1. Figure 5.2a plots coverage on the x-axis and density on the y-axis, with each point representing a tree depth. Adjacent to each point is the count of positively classified subspaces. At shallow depths (0 to 2), no subspaces are assigned to the target class. At depth 3, coverage leaps to roughly 90% and density approaches 80%, as the first subspace gets classified as 1. Despite continued improvements in these metrics with increasing depth, the number of subspaces escalates rapidly. By depth 8, the model represents the rectangle using sixteen distinct subspaces, a level of fragmentation that severely compromises interpretability.

This progressive over-segmentation exposes a fundamental limitation of HHCART(D)s greedy splitting strategy. At each node, the algorithm chooses the oblique hyperplane that locally minimizes impurity, considering only the data within that node. Lacking global coordination, these locally optimal splits collectively form a globally fragmented partitioning of the input space. Consequently, although the model achieves near-perfect coverage and density at depth 8, it does so by dividing a simple geometric region into many small, disjoint subspaces, undermining the goal of interpretability in scenario discovery.

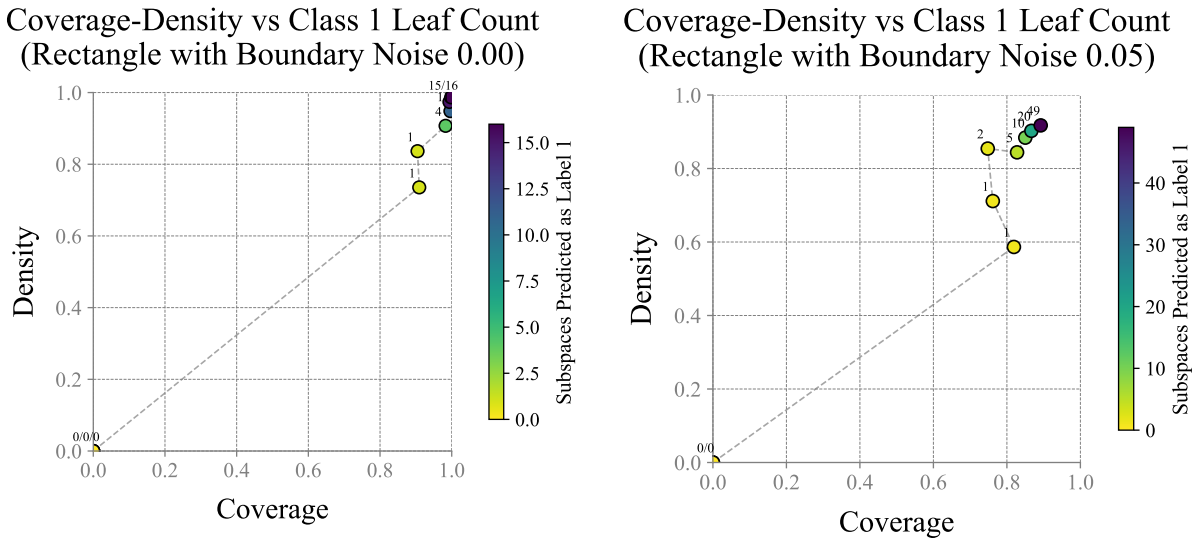
When boundary noise is applied the fragmentation of decision regions produced by HHCART(D) worsens even further and clear overfitting becomes apparent. As shown in Figure 5.1b, none of the splits align with the rectangle's edges; instead, splits are irregularly fractured near the boundary, breaking the shape into many small incoherent oblique regions. Figure 5.2b further illustrates that coverage and density increase more slowly and reach lower maxima under noise, reflecting the challenge of separating overlapping classes. Crucially, the number of positively classified subspaces grows even faster than without noise, reaching forty-nine at depth 8, which is over three times the noiseless count. This accelerated growth highlights how noise intensifies over-segmentation, as the model aims to achieve purity in mixed regions.

Figure 5.1
Effect of boundary noise on split structure for HHCART(D) on the 2-dimensional rectangle dataset, showing all oblique splits up to depth 4 under $\lambda = 0.00$ and $\lambda = 0.05$. Each line represents a decision boundary, coloured by split depth. Both models were trained using minimum purity *= 1.0 and* minimum mass *= 2 points. Panel (a) shows the noiseless case, while panel (b) applies moderate boundary noise.*



(a) **(b)**

Figure 5.2
Effect of boundary noise on coverage-density trade-off for HHCART(D) on the 2-dimensional rectangle dataset, evaluated from depth 0 to 8 under $\lambda = 0.00$ and $\lambda = 0.05$. The x-axis indicates scenario coverage, the y-axis shows scenario density, and colour encodes the number of positively classified subspaces. Both models were trained using minimum purity *= 1.0 and* minimum mass *= 2 points. Panel (a) shows the noiseless case, while panel (b) applies moderate boundary noise.*



(a) **(b)**

5.1.2 Refining Tree Structure via Minimum Purity and Minimum Mass Thresholds

To improve interpretability and reduce over-segmentation, this section examines the impact of two pre-split constraints used in HHCART(D): the *minimum purity* threshold and the *minimum mass* threshold. These parameters do not affect the quality of candidate splits directly but determine whether a node is eligible for splitting. If a node's class distribution already exceeds the purity threshold, or if it contains too few data points, it is immediately converted into a leaf without further evaluation.

Effect of Minimum Purity Threshold

The *minimum purity threshold* sets a requirement on class homogeneity before further splits are considered. Nodes that already exceed the specified proportion of samples from a single class are terminated early, preventing unnecessary splits near boundary regions affected by noise. Figure 5.3a and 5.3b show the effect of adjusting this on the 2-dimensional barbell dataset with moderate boundary noise $\lambda = 0.05$ and a *minimum mass threshold* equivalent to two samples. In Figure 5.3a, where the *minimum purity threshold* is set to 1.0, the tree generates many oblique splits by depth 4, reacting strongly to small boundary irregularities. By contrast, Figure 5.3b shows that relaxing the threshold to 0.9 suppresses a substantial portion of these, yielding a more compact structure with only 5 partitions remaining.

This difference is also reflected in the coverage-density trade-offs. Figure 5.4a and Figure 5.4b show the resulting trajectories, coloured by the number of positively classified subspaces. While the relaxed model (Figure 5.4b) uses far fewer subspaces than the strict model (Figure 5.4a), the difference in coverage and density is very small, with only a very small decrease on coverage visible. This suggests that relaxation of the *minimum purity threshold* can reduce structural complexity while retaining nearly all of the relevant scenario performance.

Effect of Minimum Mass Threshold

The *minimum mass threshold* sets a lower bound on node size, expressed as a fraction of the full dataset. Nodes containing fewer than this fraction of data points are blocked from further splitting. Figure 5.5a and Figure 5.5b show the effect of adjusting this parameter on the 2-dimensional star dataset with moderate boundary noise $\lambda = 0.05$ and a fixed *minimum purity threshold* of 1.0. In this example, *minimum mass thresholds* equivalent to 1% and 5% are compared. In Figure 5.5a, the tree produces many small oblique partitions at depth 8, Figure 5.5b, by contrast, shows that increasing the threshold eliminates many of these partitions, particularly along the shape's noisy boundary. Depth 8 was chosen because the impact of the *minimum mass threshold* is more subtle at lower depths compared to the *minimum purity threshold*, as node sizes do not yet approach the threshold set.

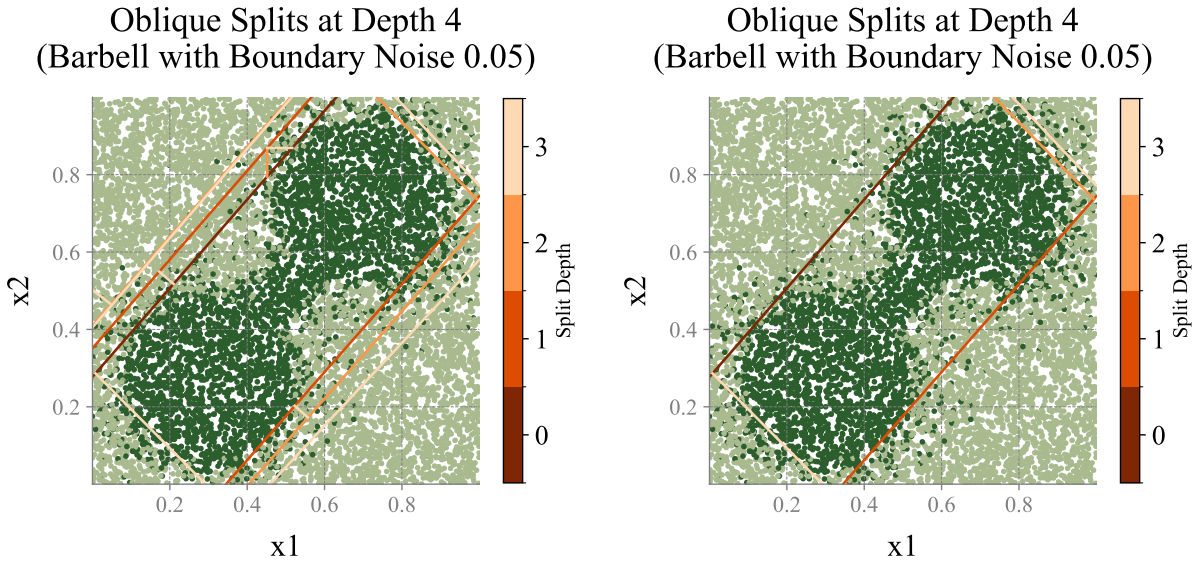
The described reduction in splitting is reflected in the scenario-level outputs. Figure 5.6a and 5.6b present the coverage–density trajectories for both settings, coloured by the number of positively classified subspaces. Although the curves are nearly identical at lower depths a slight decrease of achieved denisty is visible beyond depth 5, this is caused by the decreased oversegmentation of the shape. However, the number of positively classified subspaces at depth 8 has decreased significantly as well, from 17 to just 5. This indicates that the model becomes more selective, yielding fewer and broader scenario regions. Thus, the *minimum mass* threshold offers a simple yet effective mechanism to control partition granularity without having a big impact on coverage and density.

Regulating Tree Growth to Enhance Interpretability

The results in this section demonstrate that scenario interpretability in oblique decision trees does not arise automatically but must be deliberately engineered through structural constraints. The *minimum purity* and *minimum mass* thresholds provide two complementary mechanisms for controlling tree complexity. By stopping splits in homogeneous or data-sparse regions, these parameters reduce the number of subspaces and mitigate overfitting to noisy boundaries. This leads to simpler, more compact trees whose structure is easier to interpret and better aligned with policy-relevant distinctions.

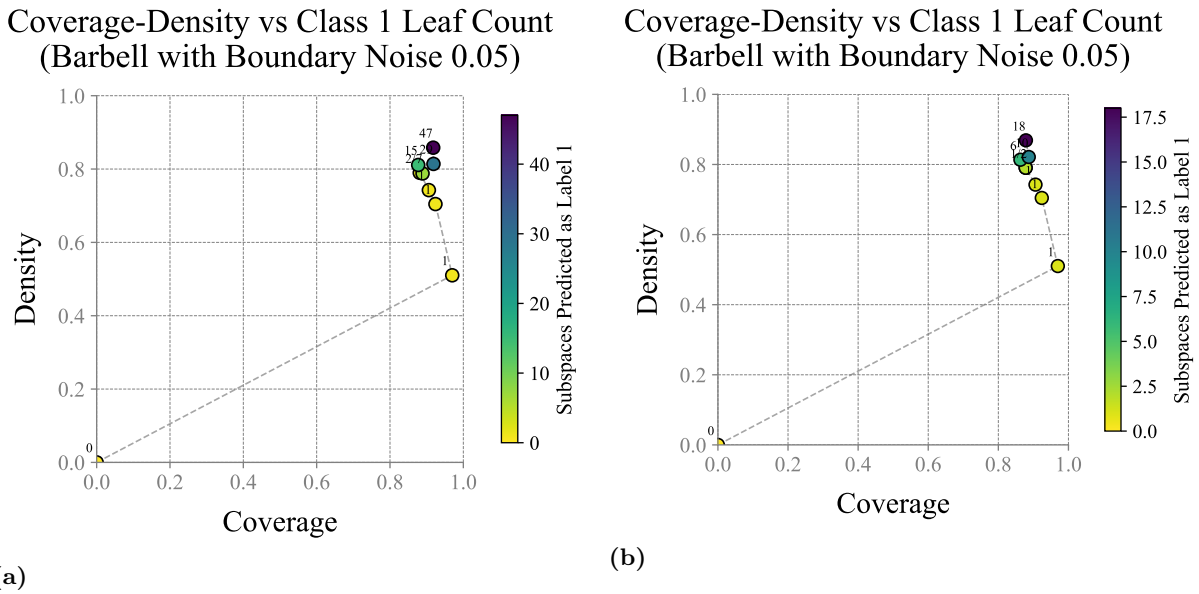
However, even with these structural constraints in place, many decision boundaries remain fragmented and challenging to interpret from a policy perspective. To investigate this limitation further, the next section applies a regularised version of HHCART(D) to the 2-dimensional rectangle and barbell benchmark shapes and compares the resulting partitions to those produced by established methods.

Figure 5.3
Effect of purity threshold on split structure for HHCART(D) on the 2-dimensional barbell dataset, showing all oblique splits up to depth 4 under moderate boundary noise $\lambda = 0.05$. Each line represents a decision boundary, coloured by split depth. Both models were trained using minimum mass *= 2. Panel (a) uses* minimum purity *= 1.0, while panel (b) uses* minimum purity *= 0.9.*



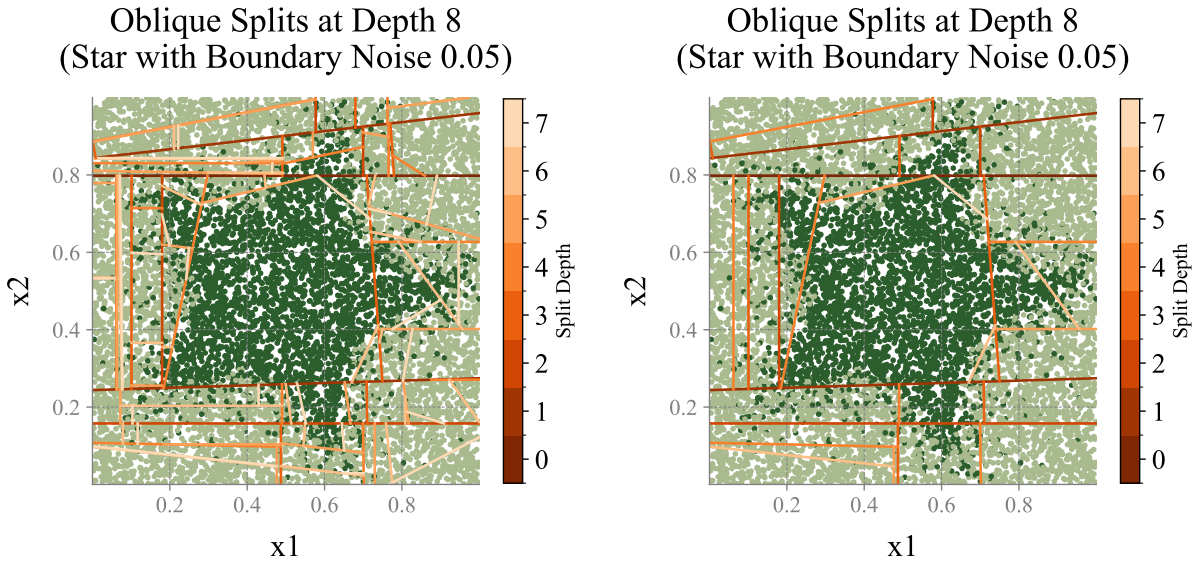
(a) **(b)**

Figure 5.4
Effect of purity threshold on coverage-density trade-off for HHCART(D) on the 2-dimensional barbell dataset, evaluated from depth 0 to 8 under noiseless conditions $\lambda = 0.00$. The x-axis indicates scenario coverage, the y-axis shows scenario density, and colour encodes the number of positively classified subspaces. Both models were trained using minimum mass *= 2. Panel (a) uses* minimum purity *= 1.0, while panel (b) uses* minimum purity *= 0.9.*



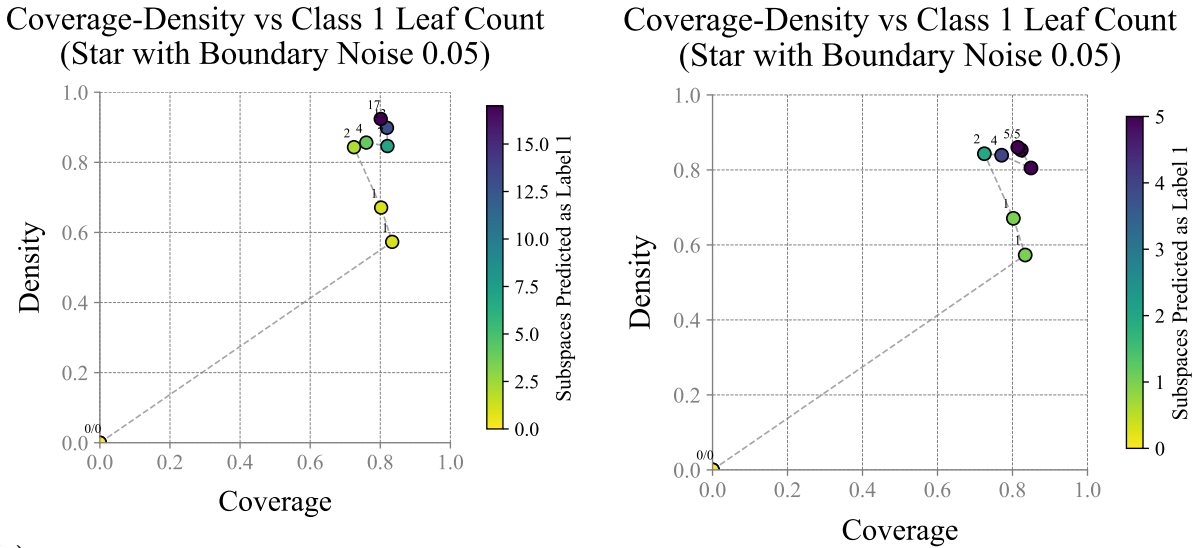
(a) **(b)**

Figure 5.5
Effect of minimum sample size threshold on split structure for HHCART(D) on the 2-dimensional star dataset, showing all oblique splits up to depth 8 under moderate boundary noise $\lambda = 0.05$. Each line represents a decision boundary, coloured by split depth. Both models were trained using minimum purity *= 1.0. Panel (a) uses* minimum mass *= 1%, while panel (b) uses* minimum mass *= 5%.*



(a) **(b)**

Figure 5.6
Effect of minimum sample size threshold on coverage-density trade-off for HHCART(D) on the 2-dimensional star dataset, evaluated from depth 0 to 8 under moderate boundary noise $\lambda = 0.05$. The x-axis indicates scenario coverage, the y-axis shows scenario density, and colour encodes the number of positively classified subspaces. Both models were trained using minimum purity *= 1.0. Panel (a) uses* minimum mass *= 1%, while panel (b) uses* minimum mass *= 5%.*



(a) **(b)**

5.2 Comparative Evaluation on Benchmark Problems

This section presents the comparative evaluation of HHCART(D) against the established methods on two two-dimensional benchmark problems. The analysis is performed on the rotated rectangle and barbell benchmarks, which were selected to present distinct geometric challenges. For each shape, HHCART(D) is applied with regularisation parameters, leveraging the insights gained from the previous section. The resulting partitions are then evaluated based on the trade-offs between coverage, density, and interpretability.

5.2.1 2D Rectangle Benchmark

The rotated rectangle benchmark provides a fundamental test of partitioning efficiency using a structurally simple, yet geometrically misaligned, region. Its 45-degree rotation creates a shape that can be perfectly defined by a minimal set of four oblique decision boundaries. Consequently, the benchmark evaluates not just whether an algorithm can achieve high coverage and density on non-axis-aligned data, but also whether it can identify the geometrically optimal solution with little splits. The analysis is conducted on data with a boundary noise setting of $\lambda = 0.05$.

PRIM produced a partition that serves as a baseline for an interpretable, single-box solution. As shown in Figure 5.7a, this solution consists of a single, axis-aligned box defined by simple restrictions on the two input variables: $x_1 \in [0.17, 0.77]$ and $x_2 \in [0.20, 0.80]$. This selected box achieved a coverage of 79% and a density of 80%. The performance is inherently constrained by the method's geometry; its axis-aligned orientation cannot match the 45-degree rotation of the target region, leading to a mismatch at the corners and edges. This specific box represents one point on the coverage-density peeling curve, which contains multiple candidate solutions illustrating the trade-off between coverage and density.

Consistent with the observation by Lempert et al. (2008) that the CART algorithm "may proliferate the number of boxes", its solution for this benchmark was structurally more complex than the single-box partition from PRIM, requiring three separate boxes to describe the region (Figure 5.7b). While these boxes can be manually aggregated into a single interpretable partition, this requires additional post-processing effort from the analyst, a step not required by PRIM, as highlighted by Lempert et al. (2008). The combination of the original three boxes resulted in a direct performance trade-off against the selected PRIM box: a higher density of 84% was achieved at the cost of a lower coverage of 73%. This specific trade-off is not necessarily unique to CART, as a box with a similar performance profile could have been selected by the modeller from PRIMs peeling trajectory. Crucially, CARTs automated, greedy procedure offers no such choice, presenting a single, fixed outcome. Therefore, given its higher initial complexity, its lack of flexibility for the modeller, and its similar coverage-density performance, the CART solution provides no compelling advantage over PRIM for this benchmark.

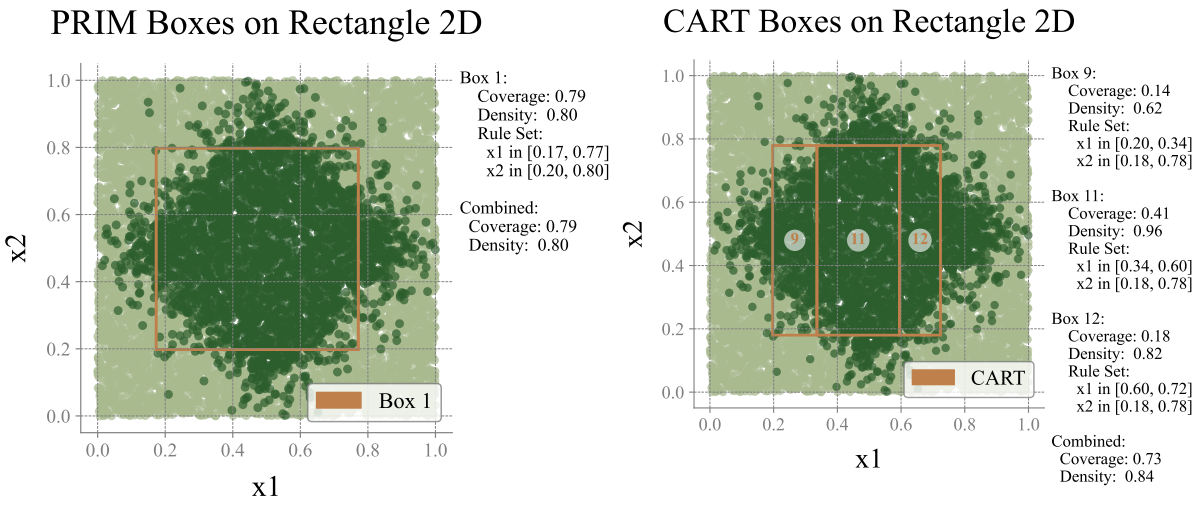
PCA-PRIM achieved the closest geometric match, aligning all four sides of the rotated rectangle with a single rotated box (Figure 5.7c). The box covered 87% of the target region with a density of 88%, representing the best result among all methods. The box followed the true shape perfectly; the fact that coverage and density did not reach 100% was entirely due to boundary noise along the edges of the rectangle, which made it impossible to include all true class 1 points without also admitting noisy points. The partition was defined by four linear inequalities in the rotated principal component space:

$$
\begin{aligned}
-2.11 \cdot x_1 - 2.75 \cdot x_2 &\geq -3.48 \\
-2.11 \cdot x_1 - 2.75 \cdot x_2 &\leq -1.39 \\
-2.75 \cdot x_1 + 2.11 \cdot x_2 &\geq -1.35 \\
-2.75 \cdot x_1 + 2.11 \cdot x_2 &\leq \quad 0.70
\end{aligned}
$$

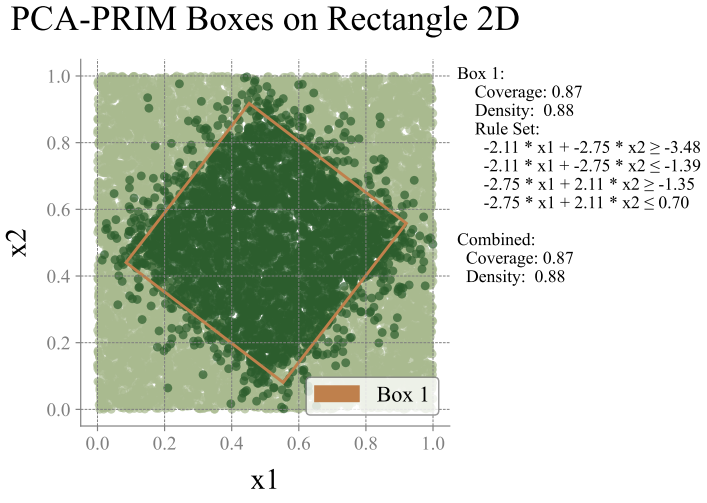
While the algorithm recovered the target shape very well, the resulting linear combinations are not as easy to interpret as simple restrictions on the original input variables.

HHCART(D) produced a partition that failed to capitalize on its geometric flexibility, creating a result that did not effectively match the rectangle's simple structure (Figure 5.8a). Using a regularized tree with *minimum mass* = 0.05 and *minimum purity* = 0.85, the model identified two distinct subspaces as belonging to the target class by depth five. The coverage-density curve shows a standard refinement path, with the model achieving a final coverage of 80% and a density of 84% at this depth (Figure 5.8b). This

Figure 5.7
*Results of PRIM, CART, and PCA-PRIM on the rotated 2-dimensional rectangle benchmark under moderate boundary noise ($\lambda = 0.05$). Panel (a) shows the axis-aligned box selected using PRIM (*minimum mass *= 0.05,* density threshold *= 0.8). Panel (b) shows the CART result (*minimum mass *= 0.05). Panel (c) shows the rotated box selected using PCA-PRIM (*minimum mass *= 0.05, density threshold = 0.8). PRIM and PCA-PRIM boxes were selected from the peeling trajectory based on the coverage-density trade-off.*

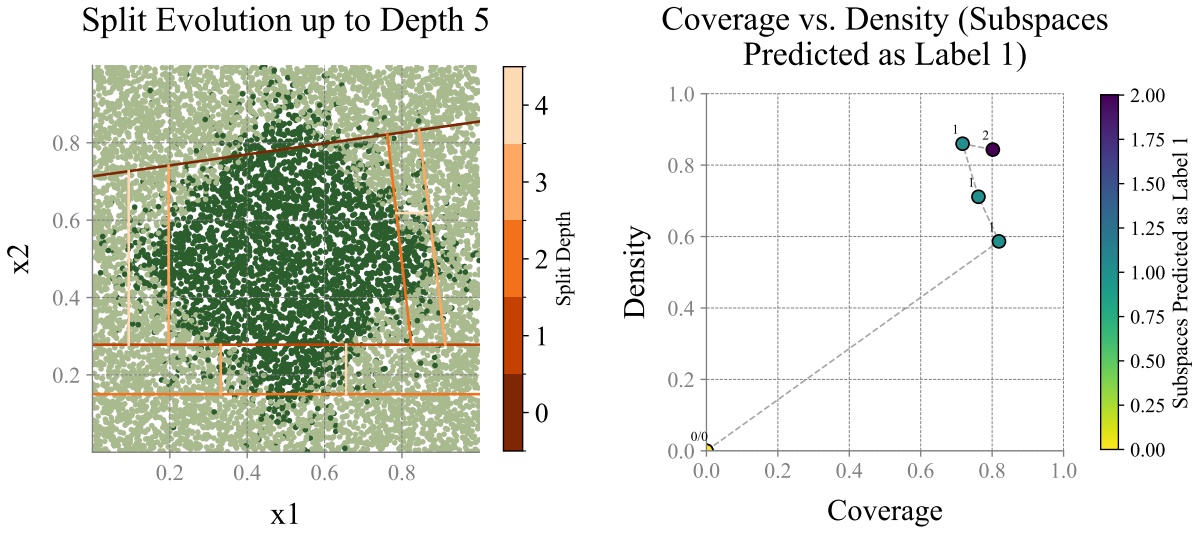


(a) **(b)**

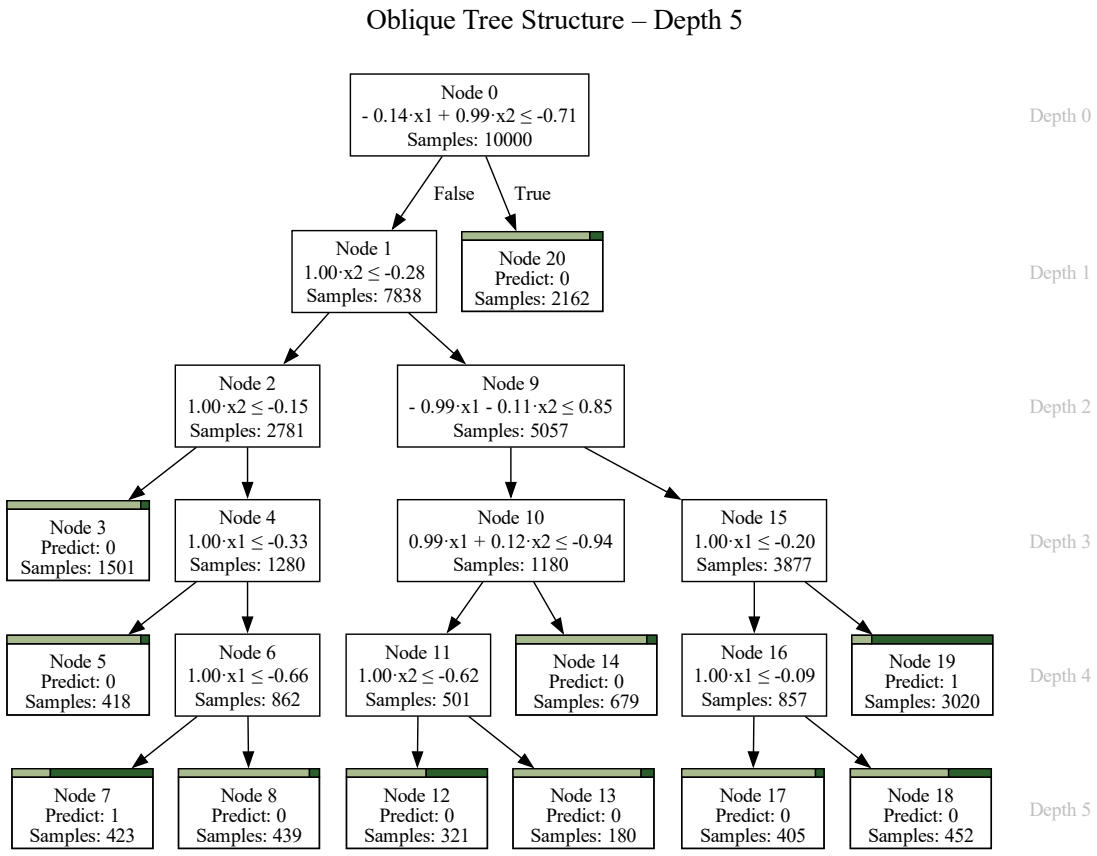


(c)

Figure 5.8
HHCART(D) results on the rotated 2-dimensional rectangle benchmark under moderate boundary noise ($\lambda = 0.05$). All panels show the model trained with minimum purity *= 0.85 and* minimum mass *= 0.05. Panel (a) overlays the oblique decision boundaries up to depth 5. Panel (b) shows the coverage-density trajectory across tree depths and number of subspaces classified as 1. Panel (c) visualises the corresponding tree structure up to depth 5.*



(a) **(b)**



(c)

performance, however, was inferior to that of PCA-PRIM and offered only a marginal improvement in the coverage-density trade-off compared to the simpler results of PRIM and CART.

Beyond its mediocre performance, the partition generated by HHCART(D) was also the least interpretable of all methods tested. An inspection of the tree structure (Figure 5.8c) reveals that the identified scenario was fragmented into two separate regions. The main subspace was defined by a path of four splits, a mix of oblique and axis-aligned rules, that did not correspond meaningfully to the rectangle's boundaries. The algorithm's heuristic thus failed to find the four simple oblique lines that would have defined the target shape, instead producing a complex and counter-intuitive description. This outcome is a direct consequence of the algorithm's constrained split-search strategy, a limitation that is explored in detail in the final synthesis of this chapter.

The results on the 2-dimensional rotated rectangle highlight a clear hierarchy of performance. PCA-PRIM provided the best geometric match and statistical performance, while PRIM offered the most interpretable solution. HHCART(D), by contrast, was neither the best performing nor the most interpretable on this benchmark.

5.2.2 2D Barbell Benchmark

The rotated barbell benchmark presents a structurally fragmented classification problem, consisting of two circular decision-relevant regions connected by a narrow linear piece. It tests each method's ability to recover multiple disjoint regions of interest, capturing the two circles and, where beneficial, portions of the connecting piece, while balancing coverage and density.

PRIM provided a simple and highly interpretable partition of the barbell shape, identifying two axis-aligned boxes centred on the two circular regions (Figure 5.9a). The selected boxes covered 70% of the target region with a high density of 96%. No additional box was selected for the connecting piece, as no candidate in the peeling trajectory satisfied the minimum mass and density thresholds. The final result thus offered a compact and transparent description of the two main regions of interest, balancing high density with moderate coverage.

CART achieved higher coverage than PRIM on this benchmark, but at the cost of increased fragmentation and reduced density (Figure 5.9b). It used four axis-aligned boxes to approximate the two circular regions and portions of the connecting piece. The combined coverage reached 76%, with a density of 86%. CART included a larger portion of the connecting piece and the surrounding mixed region, which improved coverage but resulted in a lower density. The added fragmentation of boxes increased model complexity without providing a clear performance advantage over PRIM.

PCA-PRIM yielded lower coverage and density than PRIM on this benchmark, while also producing a result that was harder to interpret (Figure 5.9c). The two rotated boxes achieved a combined coverage of 69% and a density of 95%, representing a 1% decrease in both metrics relative to PRIM. Although the rotated boxes visually captured the two circular regions, the use of eight linear combinations of the original variables made the resulting partition substantially less interpretable in terms of simple rule extraction. On this shape, therefore, PCA-PRIM provided no practical advantage over the simpler and more transparent PRIM result.

HHCART(D) initially captured the barbell benchmark through a large oblique space covering the full target region, and progressively refined this space to improve density. The resulting coverage-density trajectory (Figure 5.10b) started at high coverage from depth 1 and gradually declined as density was improved through further splitting. The model was trained using a *minimum mass* of 0.05 and a *minimum purity* of 0.90. Depths 4 and 6 are analysed here in detail, as they represent two informative trade-off points: depth 4 corresponds to the last stage with a single connected space, while depth 6 shows a balanced trade-off with coverage and density both equal to 0.83.

At depth 4, the model captured the full barbell in a single large oblique space (Figure 5.10a), achieving a coverage of 88% and a density of 79%. The space encompassed both circular regions and the connecting piece, necessarily including points near the connector that limited density. It was defined by four oblique inequalities, already introducing interpretability challenges compared to the axis-aligned used by the other methods. The use of a single space at this depth, however, preserved some degree of simplicity. The corresponding tree structure (Figure 5.10c) reflected this, with four empty leaf nodes surrounding the barbell and a single class-1 region at node 5.

At depth 6, the model refined the partition further, reducing coverage to 83% and increasing density to 83%. This refinement selectively removed portions of the initial space to better align with the target shape, but introduced substantial fragmentation and increased interpretability complexity. The space classified as 1 was now divided into three separate subspaces, collectively defined by seven oblique inequalities (Figure 5.11a). This increased tree complexity can also be observed in the tree visusalisation in Figure 5.11b.

The barbell benchmark revealed different strategic trade-offs among the methods. PRIM delivered the most interpretable, high-density result by focusing only on the two main regions at the cost of coverage. While CART achieved higher coverage by capturing more of the connecting piece, it did so with greater fragmentation and lower density. PCA-PRIM underperformed PRIM across the board. HHCART(D), in contrast, prioritized initial coverage over all other metrics, an approach that ultimately led to a highly complex and fragmented partition with poor interpretability.

5.2.3 Comparative Synthesis of Results

The results on the rotated rectangle and barbell benchmark shapes underline important differences between the evaluated methods in terms of coverage, density, and interpretability.

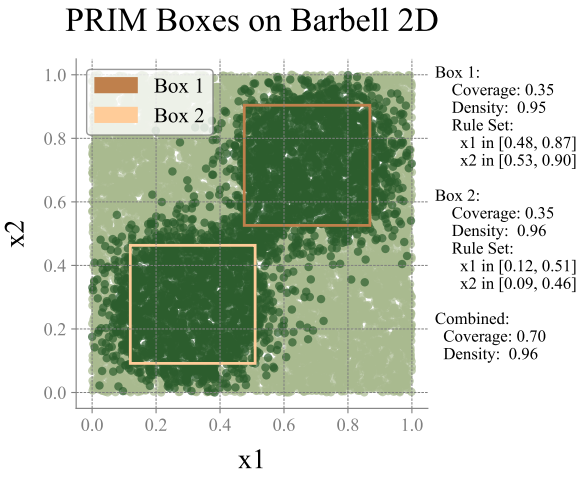
PRIM consistently provided the most transparent subspaces, achieving high density through simple axis-aligned boxes. It sometimes sacrificed coverage when no further regions could be added without violating density or mass constraints, but its outputs remained compact and easy to interpret. CART achieved higher coverage by including more of the target space and mixed regions, but this came at the cost of fragmentation and lower density, offering no interpretability advantage over PRIM. PCA-PRIM performed well on the rotated rectangle, where the geometry favoured a single rotated box, but performed worse than PRIM on the barbell benchmark and generated partitions that were harder to interpret.

HHCART(D) produced the least interpretable results across both benchmarks. On the rotated rectangle, a shape that looks easy to approach, HHCART(D) created incoherent splits and was outperformed by PCA-PRIM in both coverage and density, while requiring two oblique subspaces. On the barbell, HHCART(D) performed somewhat better: it offered the potential for higher coverage than PRIM, PCA-PRIM, or CART, but this gain came at the cost of lower density and more complex partitions. Across both benchmarks, interpretability remained a major weakness: even at conservative purity and mass settings, the resulting oblique partitions were complex, fragmented, and difficult to communicate, as evident in the split and tree visualisations.

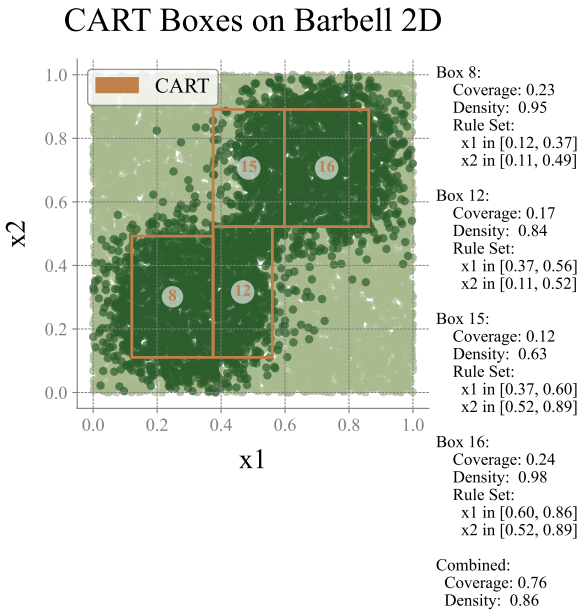
The consistent underperformance and fragmentation of HHCART(D) on these benchmarks can be explained by its core split-construction heuristic. As detailed in Appendix E, the algorithm's search for an optimal split is not exhaustive. At each node, it evaluates a limited set of candidate orientations. For a problem with d features and C classes, it tests a total of $(C+1) \times d$ directions: the d original axes, plus the d axes within each of the C class-based reflected spaces. If the true decision boundary does not happen to align with one of these, in this case six candidate directions, HHCART(D) is structurally incapable of finding it and the split minimsing impurity that the algorithm selects will not align with the intuitive border. This explains why the algorithm can produces counter-intuitive and fragmented partitions.

The findings from these two-dimensional benchmarks establish a clear pattern: HHCART(D) forces an unfavorable trade-off between performance and interpretability. While it can sometimes offer higher coverage than established methods, this gain is always accompanied by a significant increase in the complexity and fragmentation of the resulting scenarios. However, these stylized geometric tests cannot definitively answer how the algorithm would behave in a high-dimensional policy setting. The theoretical promise of oblique trees is that their true strength may lie not in partitioning simple shapes, but in navigating the complex interactions between many variables characteristic of real-world problems. The final phase of this research is therefore designed as a critical test of this remaining proposition, moving beyond low-dimensional benchmarks to evaluate whether the theoretical flexibility of HHCART(D) can translate into tangible advantages in a realistic policy analysis.

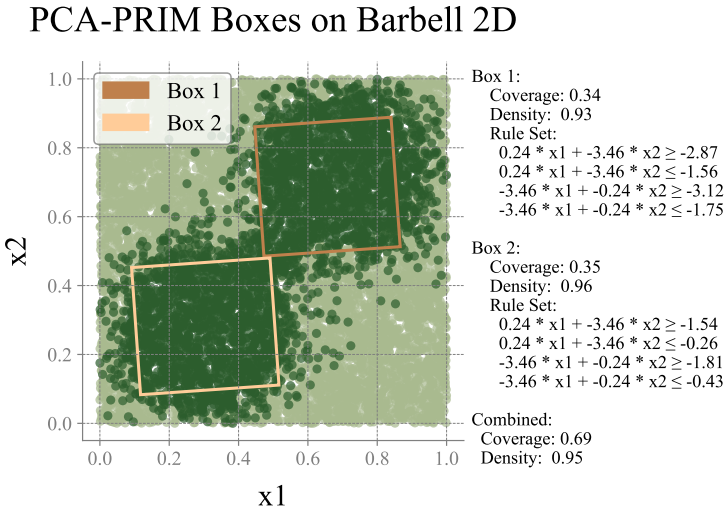
Figure 5.9
PRIM, CART, and PCA-PRIM on the rotated 2-dimensional barbell benchmark under moderate boundary noise ($\lambda = 0.05$) and minimum mass *= 0.05. Panel (a) shows the two boxes selected using PRIM (*density threshold *= 0.8). Panel (b) shows the CART result. Panel (c) shows the two rotated box selected using PCA-PRIM (*density threshold *= 0.8). PRIM and PCA-PRIM boxes were selected from the peeling trajectory based on the coverage-density trade-off.*



(a)

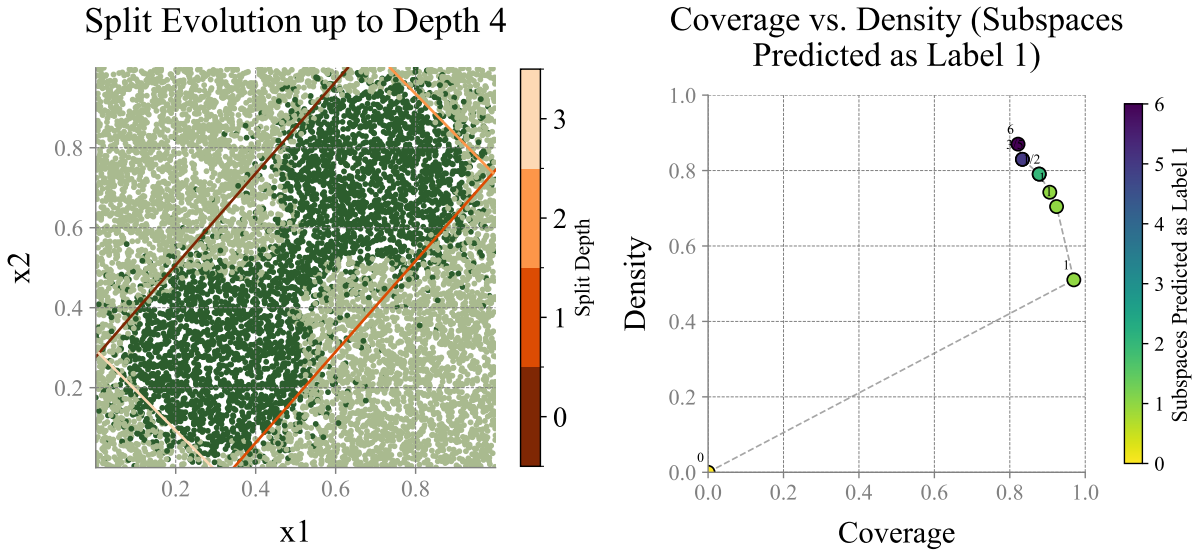


(b)

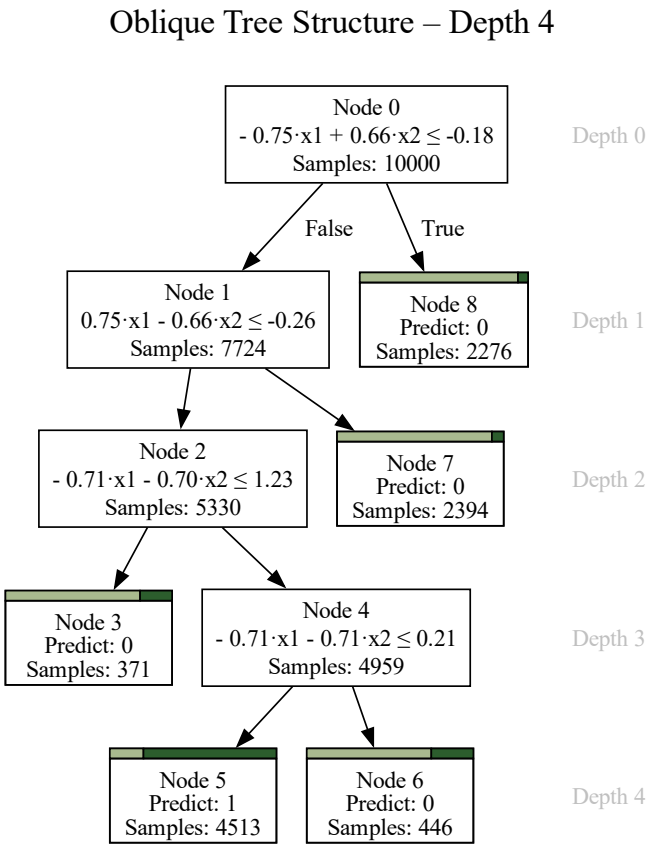


(c)

Figure 5.10
HHCART(D) on the rotated 2-dimensional barbell benchmark under moderate boundary noise ($\lambda = 0.05$). All panels show the model trained with minimum purity $= 0.90$ *and* minimum mass $= 0.05$. *Panel (a) overlays the oblique decision boundaries up to depth 4. Panel (b) shows the coverage-density trajectory across tree depths and number of subspaces classified as 1, up to depth 8. Panel (c) visualises the corresponding tree structure up to depth 4.*

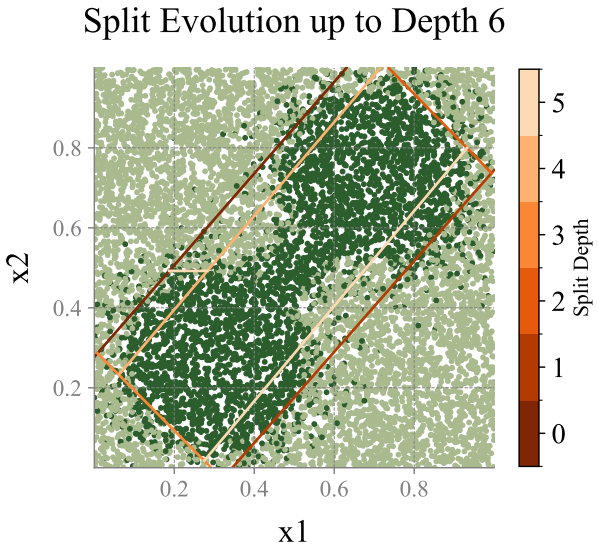


(a) **(b)**

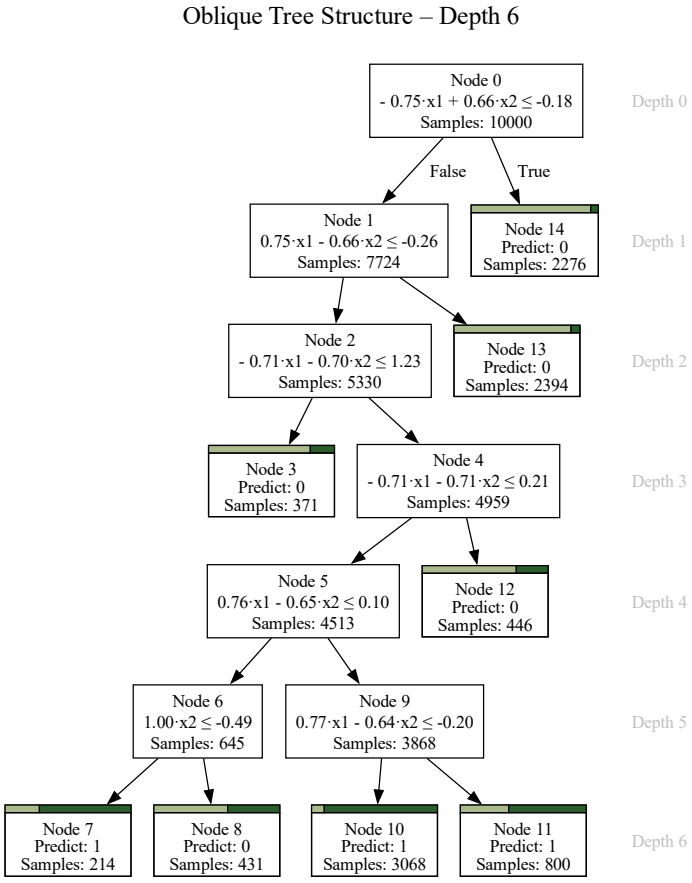


(c)

Figure 5.11
HHCART(D) at depth 6 on the rotated 2-dimensional barbell benchmark under moderate boundary noise ($\lambda = 0.05$). All panels show the model trained with minimum purity *= 0.90 and* minimum mass *= 0.05. Panel (a) overlays the oblique decision boundaries up to depth 6. Panel (b) visualises the corresponding tree structure up to depth 6.*



(a)



(b)

6 Phase 3: Comparison on Policy Model

This final phase evaluates the practical utility of oblique decision trees in a complex, high-dimensional policy context. Building on the benchmarking results from synthetic problems, HHCART(D) is now tested against established scenario discovery methods, PRIM, PCA-PRIM, and CART, to assess its performance under realistic modelling conditions. The analysis uses the EU energy system model introduced in Section 3.3.1, which simulates long-term developments in energy supply and emissions under deep uncertainty. The chapter first outlines the experimental setup, then presents scenarios identified by the established methods, followed by a feature-selected application of HHCART(D). It concludes with a comparative synthesis of interpretability and scenario quality across all methods.

6.1 Experimental Setup

To enable a robust scenario discovery analysis, it was first necessary to generate a simulation ensemble that captured a wide variety of plausible outcomes across the model's high-dimensional uncertainty space. To this end, the EU energy system model was simulated 10 000 times, with each run covering the period from 2010 to 2050 under the Emissions Trading Scheme policy, following precedent set by Hamarat et al. (2014). Latin Hypercube Sampling was used to structure the experimental design, ensuring that the full range of each of the 46 uncertain input parameters was systematically explored.

Once the dataset was generated, binary classification was applied to define the outcome of interest for scenario discovery. Following the precedent set by Hamarat et al. (2014) in their search for policy opportunities, a case is defined as *of interest* if the share of renewables in the final year, 2050, exceeds 40%. As illustrated in Figure 6.1, this threshold divides the ensemble into two parts: 4,637 cases (46.4%) are classified as of interest, while the remaining 5,363 (53.6%) are not. This setup provides a consistent basis for comparing each algorithm's ability to identify the conditions associated with high-renewable futures.

6.2 Scenarios from Established Methods

The application of the established methods to the generated ouputs revealed a key structural difference between the scenarios they produced. The PRIM analysis resulted in two distinct single-box scenarios. In contrast, the CART analysis generated a single multi-box scenario, where the conditions for success were described by a collection of three smaller, separate boxes. Although both methods provided interpretable results, neither could identify a scenario that combined high coverage and high density.

The PRIM analysis of the peeling trajectory identified two scenarios representing the best available trade-offs between coverage and density. As shown in Figure 6.2a, these two *best* boxes present a clear choice: Box 4 achieves a coverage of 62% with a density of 64%, whereas Box 5 achieves a higher density of 72% at the cost of a lower coverage of 51%. However, it is important to note that the performance of both scenario boxes is modest. The 64% density of Box 4 offers only a marginal improvement over the dataset's base rate of 46.4%, limiting its predictive power. Despite Box 5's density being higher, its coverage of 51% means it fails to explain nearly half of the desirable futures. When looking at interpretability of the scenarios, however, both scenarios are simple and defined by restrictions on only two of the same categorical variables: *Switch electrification rate* and *Switch economic growth* (see Figure 6.2b for the specific rules of Box 5). Thus, the primary outcome of the PRIM analysis is a choice between these two single-box scenarios, as attempts to identify additional non-overlapping scenarios were unsuccessful because of poor performance and excessive overlap (see Appendix F).

The CART analysis provides a complementary perspective on the scenario space. Instead of yielding a single scenario, the CART tree (Figure 6.3) produced a set of three leaf nodes that together formed a

multi-box scenario. The combined coverage and density of this set were 66% and 65%, respectively. In comparison to the PRIM results, the CART tree extended coverage beyond that of either individual PRIM box but achieved a density only marginally better than Box 4 and substantially lower than Box 5. Thus, the familiar trade-off between coverage and density remained: CART did not uniformly outperform the PRIM solutions despite its use of multiple boxes, but instead offered a different balance between coverage and density. The CART scenario set can be summarised as follows:

- **Scenario 1**: Coverage of 27.8%, density of 67.3%; defined solely by an early *Time Of Nuclear Power Plant Ban* (before 2030), with no additional constraints.
- **Scenario 2**: Coverage of 28.9%, density of 66.8%; characterised by a late *Time Of Nuclear Power Plant Ban* (post-2030), combined with restrictions on *Switch Economic Growth* (categories 3, 4, 6) and *Switch Physical Limits* (category 2).
- **Scenario 9**: Coverage of 9.2%, density of 52.3%; refining Scenario 2 by further restricting *Starting Construction Time* (early start) and *Uncertainty Initial Gross Fuel Costs* (lower than the maximum). While having the same restrictions on *Switch Economic Growth* (categories 3, 4, and 6) and *Switch Physical Limits* (category 2).

Overall, the CART tree did not achieve a substantially better performance than PRIM, despite using three boxes to define combinations of uncertain inputs leading to outcomes of interest. This multi-box structure reduced interpretability, as the resulting scenarios were more fragmented and required more complex combinations of conditions to explain only a slightly number of desirable outcomes. Moreover, producing a coherent scenario required post-processing to merge similar leaf nodes, adding to the analyst's workload, as also noted by Lempert et al. (2008).

PCA-PRIM was also applied to the dataset; however, it produced results identical to those of the standard PRIM implementation. This outcome occurred because, even after rotation via principal component analysis, the algorithm continued to prioritise the informative categorical variables when defining the box boundaries. As no additional insights were gained, PCA-PRIM is excluded from further analysis in this chapter. The full results are provided in Appendix F.

Figure 6.1
Time series plot showing the fraction of renewables for all 10 000 simulation runs between 2010 and 2050. The horizontal line in the kernel density estimate indicates the 40% threshold used to define the cases of interest. *Trajectories that finish above this line in 2050 are classified as* of interest *(dark green), while those below are not (light green).*

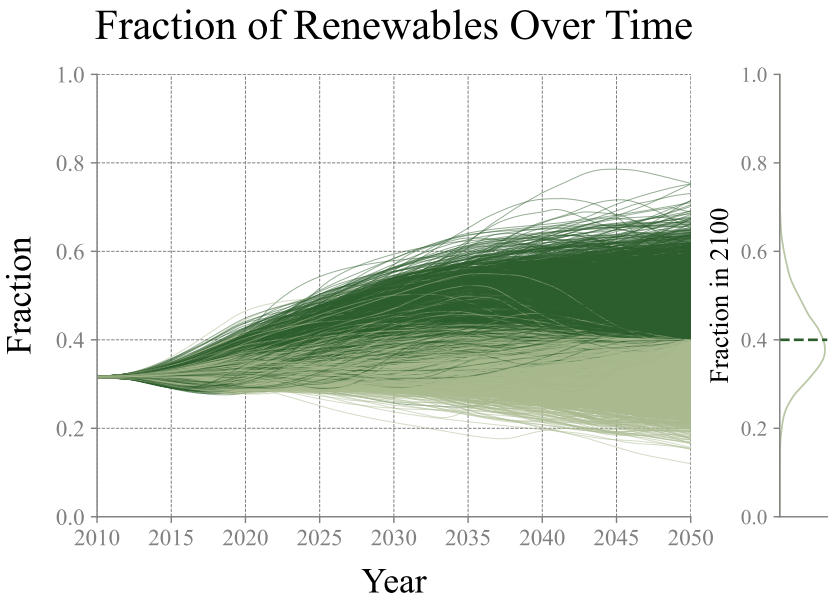


Figure 6.2
Results of the PRIM analysis applied to the EU energy model output. Panel (a) shows the coverage-density trade-off curve, with colours indicating the number of restricted dimensions. Panel (b) shows the scenario rules for the selected Box 5.

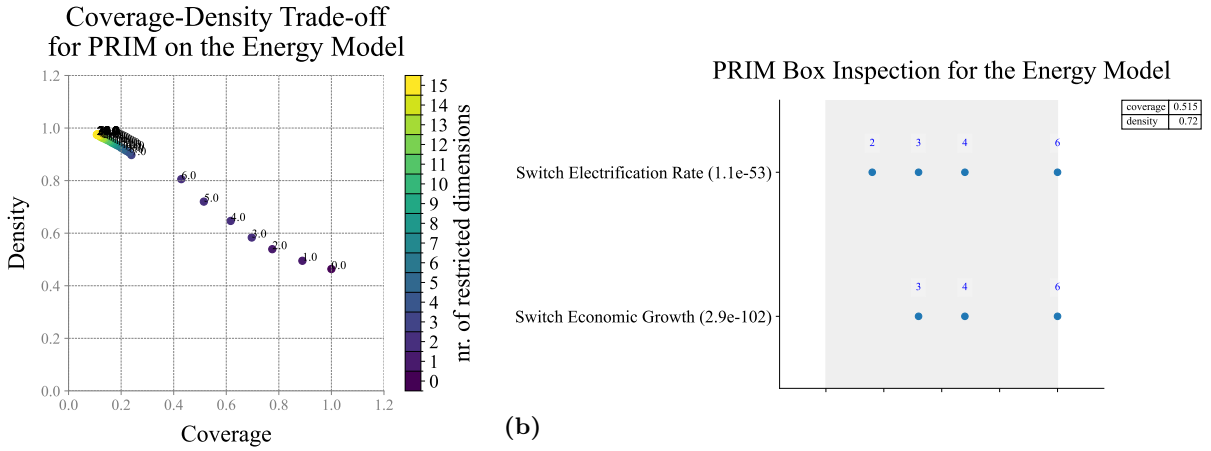
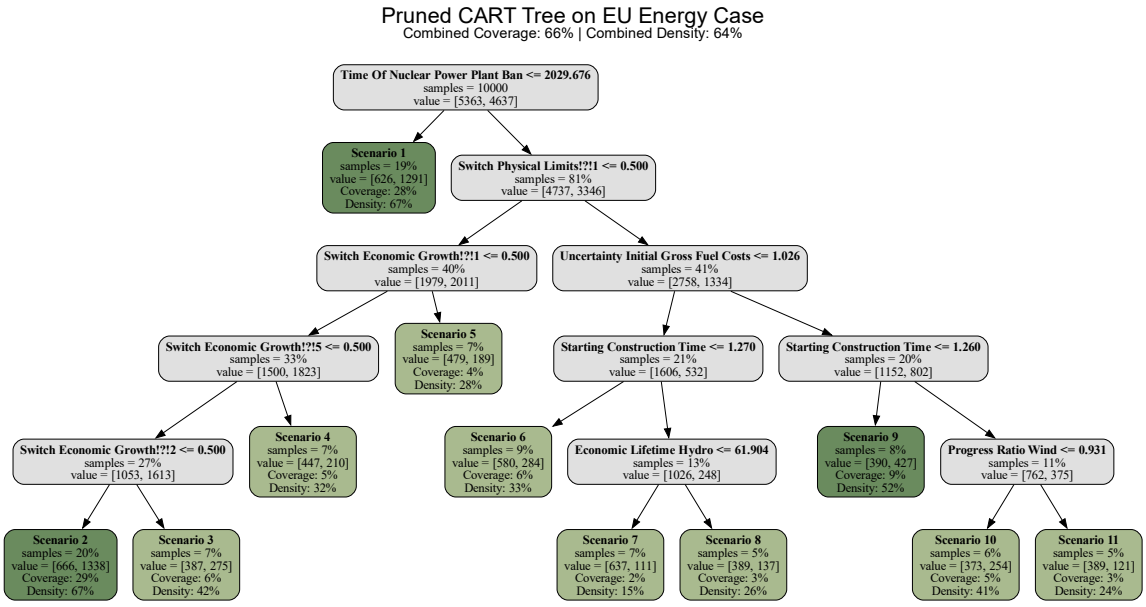


Figure 6.3
Results of the CART analysis applied to the EU energy model. The figure shows the final pruned CART tree. Each leaf node represents a scenario annotated with coverage and density. Node colours indicate classification outcomes: dark green for leaves classified as 1 (high-renewable futures), light green for leaves classified as 0, and grey for internal decision nodes.



6.3 Feature Selection for HHCART(D)

To inform the construction of interpretable oblique decision trees, an Extra-Trees classifier was employed to assess feature importance in achieving high-renewable futures. The algorithm quantifies the influence of each variable by calculating its Mean Decrease in Impurity score. This score reflects a variable's average contribution to reducing Gini impurity across the many randomised trees in the ensemble, with the resulting scores providing a relative ranking of predictive relevance. As summarised in Table 6.1, the combined contribution of the five most informative variables was modest, accounting for only 17.9% of the total impurity reduction.

This low combined importance highlights the absence of any single variable or small set of variables that strongly shapes the system's behaviour, even when accounting for both individual effects and interactions captured by the extra-trees analysis. The results reflect the inherent complexity of the policy problem: high-renewable futures in the EU energy system arise from the joint influence of many variables, with no single factor exerting a dominant first- or higher-order effect. This structural characteristic of the problem space partially explains the limited scenario quality achieved by the baseline methods: desirable futures cannot easily be captured by restricting a few key dimensions because the outcome space is shaped by a broad and distributed set of interacting influences.

Table 6.1
Top five feature importance scores derived from an Extra-Trees classifier, trained to distinguish high-renewable futures from all others in the EU energy model. The scores represent each variable's relative contribution to reducing Gini impurity across the ensemble, using the Mean Decrease in Impurity metric. The total importance sums to 1, with the top five features collectively accounting for 17.9% of the model's explanatory power.

	Importance
Time Of Nuclear Power Plant Ban	0.049
Starting Construction Time	0.036
Uncertainty Initial Gross Fuel Costs	0.034
Switch Physical Limits 2	0.031
Switch Economic Growth 6	0.030
Sum of Top 5	0.179

6.4 Scenarios from HHCART(D)

The iterative application of HHCART(D) was undertaken with two primary objectives: first, to assess whether increasing the number of input features improved the scenario quality, and second, to determine whether a high-performing scenario could be identified using only a small, highly interpretable set of variables. The analysis revealed that while adding features yielded modest performance gains, the algorithm consistently produced complex scenarios and, critically, failed to leverage its theoretical geometric flexibility in this high-dimensional policy context.

A consistent pattern emerged across all feature configurations, as shown in the performance trajectories in Figure 6.5. The initial splits (up to depth 3) caused the model performance to jump irregularly across the coverage-density space, as each new partition refined the trade-off between the two metrics. Beyond depth 4, however, these refinements became marginal, with deeper trees primarily increasing the number of subspaces substantially without meaningfully improving the overall trade-off. This clear point of diminishing returns justifies selecting the depth 4 configurations as the most efficient balance between performance and structural complexity for the final comparison. The full depth-wise results used for this selection are provided in Appendix F.

The final performance of the selected depth 4 models is summarised in Figure 6.4. The plot illustrates the trade-off between coverage, density, and complexity (annotated by the number of subspaces) for the

models trained on two to five features. When considering these three objectives, the analysis reveals a Pareto front consisting of two non-dominated configurations. The first is the four-feature model, which achieves a density of 68% at a coverage of 52% using six subspaces. The second is the five-feature model, which offers a more balanced trade-off, achieving a higher coverage of 62% at a lower density of 64% using eight subspaces. In contrast, the models trained on two and three features are clearly sub-optimal; despite using more subspaces (nine each), they deliver inferior performance on both coverage and density and are dominated by the five-feature configuration. The final choice for an analyst is therefore between the two solutions on the Pareto front, a decision that involves balancing not only coverage and density but also the added complexity of using more input features. Detailed numerical results for this comparison, including tree depth, are provided in Table F.5 in Appendix F.

The five-feature model was selected for detailed visualisation as it represents the most balanced trade-off between coverage and density among the non-dominated solutions. The full decision tree is shown in Figure 6.6. For the sake of visual clarity in this figure, only features with non-zero weights are displayed in the split equations for each node.

A detailed analysis of this tree's structure reveals a critical insight into the algorithm's behaviour in this high-dimensional application. While only two of the fourteen splits are strictly axis-aligned, the remaining oblique splits are extremely weak. These splits are consistently dominated by a single feature with a weight very close to 1.0, while the weights on all other features are orders of magnitude smaller; for instance, the largest secondary weight has a magnitude of only 0.0138, and others are as small as 0.0062.

This *weakly oblique* structure has two profound implications for the utility of the method in scenario discovery. First, it makes the resulting scenarios highly complex to interpret. Instead of a simple threshold on a single variable, the decision boundary is a 5-dimensional hyperplane where the marginal contributions of the features are almost negligible, undermining the goal of clear, communicable rules. Second, this behaviour demonstrates that the theoretical geometric flexibility of HHCART(D) was not effectively leveraged. The algorithm did not identify strong, meaningful interactions between variables but instead made small adjustments to what are, for practical purposes, axis-aligned splits.

Figure 6.4
Coverage versus density of HHCART(D) models using 2 to 5 features on the EU energy model. Marker size and annotation denotes the number of class 1-labelled regions (interpretability proxy), and colour encodes the number of input features used. The dashed black line indicates the Pareto front of non-dominated configurations based on coverage and density.

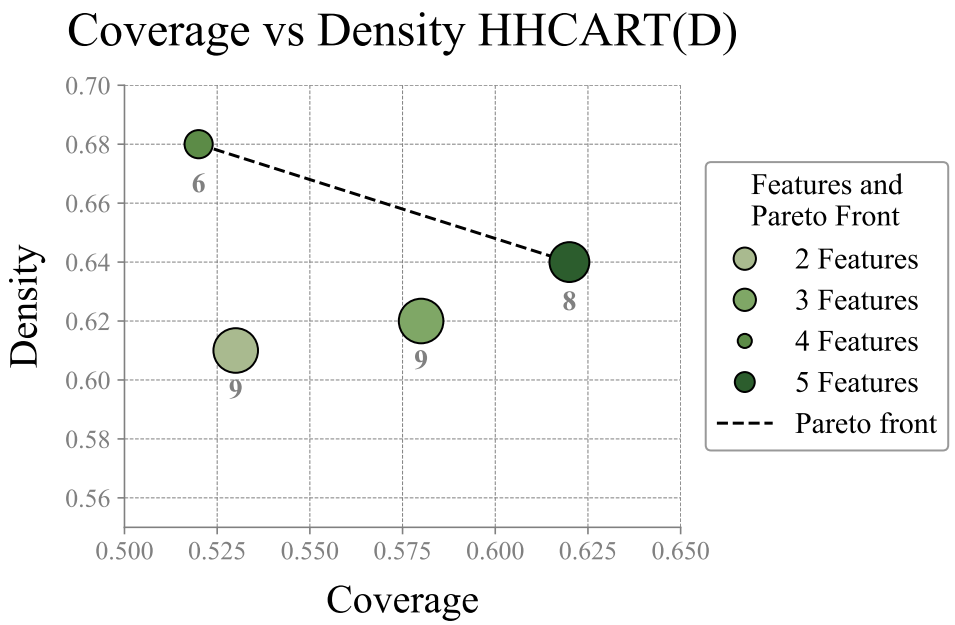
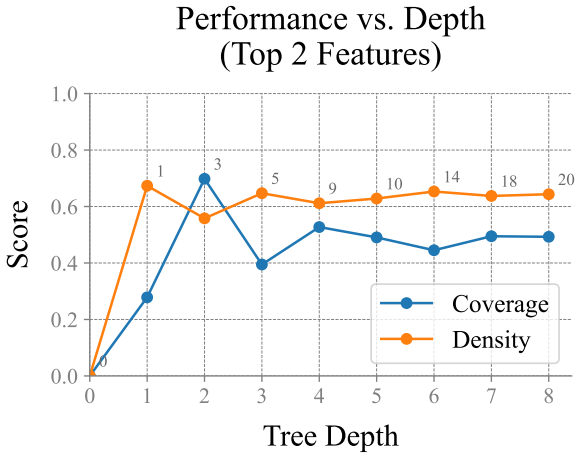
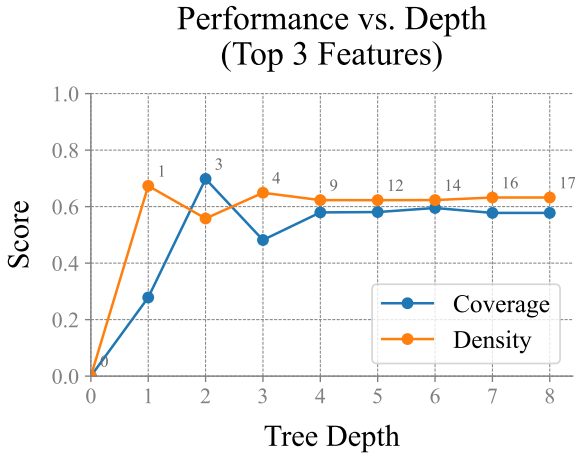


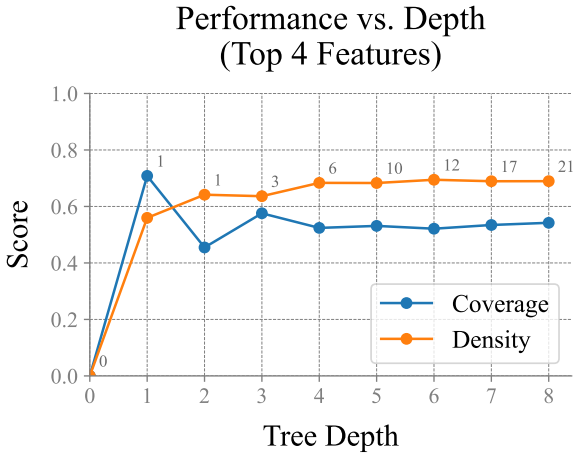
Figure 6.5
Coverage-density trade-off for HHCART(D) on the EU energy model, evaluated from depth 0 to 8 with increasing number of features used. The x-axis indicates tree depth, the y-the evolution of coverage end density over tree depth. Points are annotated with the number of subspaces classfied as 1. All models were trained using minimum purity = *0.95 and* minimum mass = *0.05. The features used in each sub-panel are as follows: Panel (a) uses 2 features (*Time Of Nuclear Power Plant Ban, Starting Construction Time*); Panel (b) uses 3 features, adding* Uncertainty Initial Gross Fuel Costs*; Panel (c) uses 4 features, adding* Switch Physical Limits 2*; Panel (d) uses 5 features, adding* Switch Economic Growth 6.



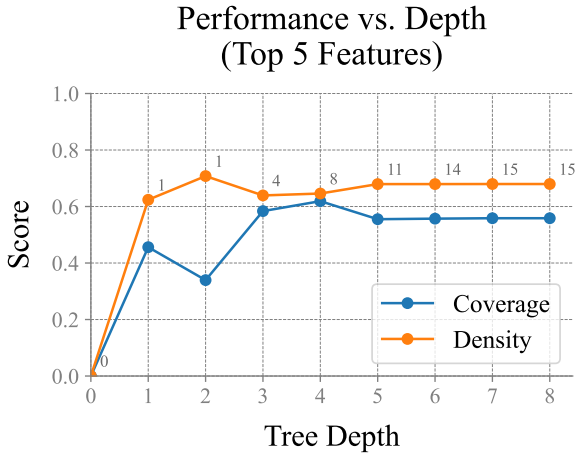
(a)



(b)



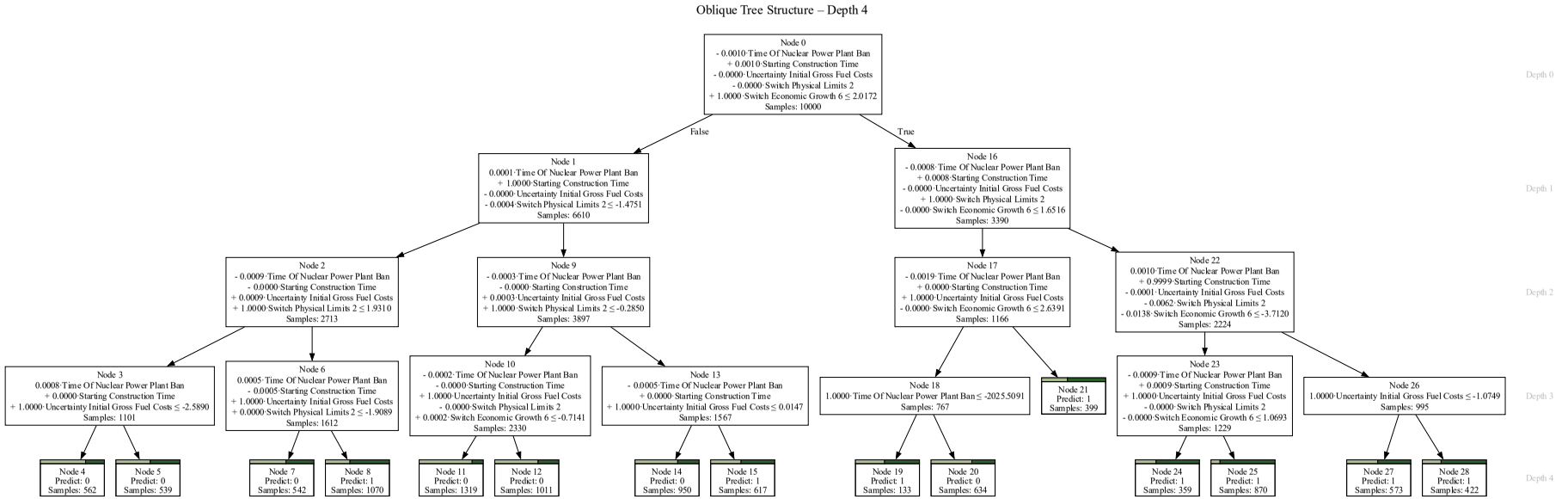
(c)



(d)

Figure 6.6

Visualisation of the trained HHCART(D) decision tree on the EU energy model at depth 4, using 5 features. The figure is rotated to fit the landscape orientation.



6.5 Comparative Synthesis of Results

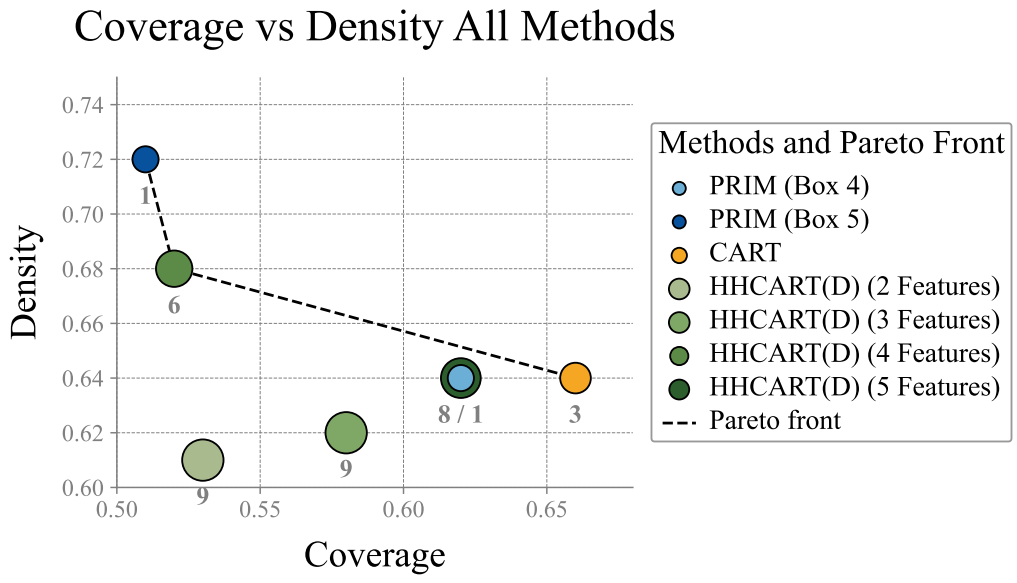
A direct comparison of the best-performing models from each method reveals that the oblique tree approach did not yield a substantial improvement in statistical performance. The results show that no HHCART(D) configuration offered a big advantage in coverage or density over the scenarios from PRIM and CART. Instead, as illustrated in Figure 6.7, while HHCART(D) produced solutions on the Pareto front, this performance was consistently associated with a higher degree of structural complexity. Full numerical details for all comparisons are available in Table F.5 in Appendix F.

The trade-offs between statistical performance and model complexity are evident when comparing the scenarios from each method. A comparison of the high-density scenarios shows that Box 5 from the PRIM trajectory offers a density of 72%. The four-feature HHCART(D) model, in contrast, results in a significantly lower density, although it achieves a marginally higher coverage. This alternative performance profile, however, is achieved with a substantial increase in complexity: the HHCART(D) scenario is composed of six distinct subspaces defined by four features, whereas the PRIM scenario is a single, interpretable box defined by only two.

A similar pattern emerges with the more balanced scenarios. The five-feature HHCART(D) model produces a result with a coverage of 62% and a density of 64%, statistically identical to that of Box 4 from PRIM. However, the HHCART(D) model requires eight distinct subspaces and five features to reach this performance, while the PRIM scenario is again a single box defined by two features. In comparison, the CART model offers a more favourable trade-off, achieving a higher coverage of 66% for a similar density. Although this requires three subspaces, making it slightly harder to understand then a PRIM scenario, these are simple axis-aligned boxes that are substantially easier to interpret than the eight weakly oblique subspaces of the HHCART(D) model.

Furthermore, the interpretability challenge is compounded by the nature of the splits themselves. An analysis of the five-feature HHCART(D) tree reveals that its decision boundaries, while technically oblique, are consistently dominated by a single variable with a weight near 1.0. The other features in the splits have weights of a much smaller magnitude, meaning the resulting hyperplanes are only minor deviations from a purely axis-aligned split. This behaviour creates a significant drawback: the scenarios are defined by complex, multi-variable linear equations that are harder to interpret than simple thresholds, yet they do not leverage the core strength of oblique splits by meaningfully rotating the decision boundary to capture non-orthogonal relationships in the data.

Figure 6.7
Coverage versus density of established methods and HHCART(D) models on the EU energy model. Marker size and annotation denotes the number of class 1-labelled regions (interpretability proxy), and colour encodes the model type and number of input features used. The dashed black line indicates the Pareto front of non-dominated configurations based on coverage and density.



7 Discussion

This chapter synthesises the main findings of this thesis and considers their implications for scenario discovery and policy analysis. It begins by identifying key insights that emerged from the comparative evaluation of oblique and established scenario discovery methods (Section 7.1). It then outlines the societal and scientific contributions of the research (Section 7.2), followed by a critical appraisal of its limitations and the validity of the findings (Section 7.3). The chapter concludes by identifying directions for future research (Section 7.4).

7.1 Key Findings and Insights

The central finding of this thesis is that the theoretical advantages of oblique decision tree algorithms, as represented by HHCART(D), did not translate into practical benefits for scenario discovery. Despite its design to support more flexible and geometrically expressive splits, HHCART(D) failed to outperform established axis-aligned and globally rotated methods such as PRIM, CART, and PCA-PRIM. Where statistical performance was comparable, it came at the cost of greater model complexity, including deeper trees, fragmented partitions, and diminished interpretability.

This finding is particularly notable given that HHCART(D) was selected as the best-performing oblique algorithm. It demonstrated robustness to irrelevant features and maintained competitive performance in high-dimensional settings, traits that made it a strong candidate for more complex scenario discovery tasks. However, across both stylised benchmarks and the EU energy model, its partitions were consistently more complex while failing to yield improvements in coverage or density. Despite its capacity for flexible, multivariate splits, the algorithm rarely departed from axis-aligned behaviour, and the resulting scenarios lacked the structural coherence achieved by simpler baseline methods.

The remainder of this section outlines the key explanatory factors behind this outcome.

Oblique Splits Rarely Add Expressiveness

A central finding from the Phase 3 experiment is the stark disconnect between the theoretical flexibility of HHCART(D) and its practical application. The algorithm's capacity for geometrically flexible, oblique splits went largely unused; its partitions were functionally indistinguishable from simple axis-aligned thresholds. This occurred because, while formally oblique, the splits were consistently dominated by a single variable with a weight near 1.0, rendering all other feature contributions negligible. As a result, the added algebraic flexibility of the method did not produce more expressive or structurally distinct scenarios. Instead of leveraging its advanced capabilities, the model failed to move beyond simple univariate rules, undermining the core theoretical advantage that justified its selection.

Limited Heuristic Scope Reduces Partition Quality

This failure to leverage geometric flexibility can be partly attributed to the algorithm's limited search strategy. The fixed reflection-based heuristic used by HHCART(D) constrained its ability to discover high-quality splits, as it considers only directions derived from per-class covariance matrices at each node. This design is efficient but tends to fail when the direction of greatest variance for both classes does not align, or is not orthogonal to the direction that best separates the classes. In the rotated rectangle benchmark, for instance, this mismatch led to suboptimal and fragmented partitions that PCA-PRIM resolved with a single rotated axis.

Global Feature Selection Restricts Local Adaptivity

To mitigate the lack of split-level sparsity, HHCART(D) was tested using a globally selected subset of features in Phase 3. However, impurity analysis showed that the top five features explained only 17.9% of the total impurity reduction, indicating that no global subset could represent all decision-relevant structure. As a result, the model was unable to adapt its partitions to locally informative features. These

findings suggest that locally sparse methods, capable of selecting features dynamically per split, may better accommodate high-dimensional and structurally diffuse scenarios.

Greedy Induction Limits Global Coherence

Like most decision tree algorithms, HHCART(D) uses a greedy, node-level induction strategy. While effective for impurity reduction, this approach often resulted in fragmented partitions that failed to capture global structure. In both synthetic benchmarks such as the rotated rectangle and star, and in the EU energy model, HHCART(D) produced scenario descriptions that were less coherent and harder to interpret than those generated by methods with more global perspective, such as PCA-PRIM.

Linear Split Formulations Limit Model Expressiveness

The disappointing performance of HHCART(D) in Phase 3 must also be understood in light of its reliance on linear decision boundaries. Generated splits are hyperplanes defined by linear combinations of input variables. This inherently restricts the model's ability to represent more complex nonlinear relationships and threshold effects. The EU energy model includes 33 ordinary differential equations and 499 auxiliary variables, creating a highly nonlinear system structure. In such contexts, linear splits are often too simplistic to approximate the underlying dynamics, which likely contributed to the disappointing performance on coverage and density.

Scenario Discovery Requires Different Objectives

A broader insight emerging from this research is that the design objectives of standard classification trees diverge from those of scenario discovery. Classification methods like HHCART(D) optimise local purity, while scenario discovery requires the identification of high-coverage, high-density, and interpretable regions in input space. Without changes to their core induction logic, classification-derived algorithms will remain misaligned with the structural goals of scenario discovery.

7.2 Relevance and Implications

This thesis contributes to both the practice and theory of scenario discovery and decision making under deep uncertainty by critically evaluating the utility of oblique decision tree algorithms. The findings provide cautionary guidance for the application of these methods in policy analysis and illuminate a path forward for future scientific research. The following sections detail these societal and scientific implications.

7.2.1 Societal Relevance

This thesis contributes to the broader goal of improving the analytical tools used to support robust decision-making under deep uncertainty. While it does not directly engage with specific policy domains, such as climate adaptation or energy transitions, it strengthens the methodological foundations of scenario discovery, a technique widely used in those contexts. For scenario discovery results to inform policy effectively, the underlying algorithms must produce interpretable, reliable, and communicable insights. This study contributes to that goal by evaluating whether more complex machine learning algorithms can meet these standards without sacrificing clarity.

The findings offer a significant cautionary conclusion for the policy analysis community: algorithmic flexibility does not guarantee better insight. This work demonstrates that the tested oblique decision tree algorithm, HHCART(D), produced more fragmented and less intuitive scenarios than established methods, without a consistent improvement in statistical performance. For analysts and decision-makers, this reaffirms the value of existing, transparent tools like PRIM. The simplicity and communicability of these established methods remain critical assets in navigating complex, high-stakes decisions under uncertainty.

Ultimately, this research reinforces a core principle of decision support in the public sphere: analytical tools must be chosen to clarify uncertainty, not to add another layer of complexity. The development of the open-source benchmarking framework in this thesis further supports this goal by providing a practical toolkit for the rigorous and transparent evaluation of future methods, strengthening the evidentiary basis for policy analysis.

7.2.2 Scientific Relevance

This thesis makes three distinct scientific contributions to the field of scenario discovery. The first and primary contribution is an empirical one: it presents the first systematic, benchmark-based evaluation of oblique decision tree algorithms for scenario discovery. The findings demonstrate that the anticipated advantages of geometric flexibility did not consistently yield performance improvements over established methods. In particular, the structural limitations of the tested oblique algorithm, its tendency to produce fragmented partitions and to default to effectively axis-aligned splits in high-dimensional spaces, offer critical empirical evidence that challenges the theoretical promise of these techniques.

The second contribution is methodological: the design and implementation of a reusable benchmarking framework tailored specifically to scenario discovery. Released as the *Oblique-Decision-Tree-Algorithms-for-Scenario-Discovery* project, the framework enables systematic comparison of multiple oblique decision tree algorithms and supports the integration of new methods. It includes custom synthetic shape generators to test algorithm performance under controlled geometric conditions and is the first to incorporate configurable boundary noise to simulate the ambiguity typical of real-world models. In addition to standard scenario discovery metrics (coverage, density), the framework captures structural characteristics such as tree depth, number of subspaces, and feature usage per split, and provides rich visualisation tools, including tree diagrams, decision boundaries, and coverage–density trajectories, that support both diagnostic analysis and interpretability. Together, these features establish a practical, extensible standard for future scenario discovery algorithm development and evaluation.

Finally, this research makes a conceptual contribution by deepening the theoretical understanding of why oblique decision tree algorithms underperform in scenario discovery tasks. It identifies a fundamental misalignment between the local optimisation objective of classification algorithms – typically focused on impurity reduction – and the global goals of scenario discovery, which prioritise coherence, interpretability, and policy relevance. The consistent shortcomings of greedy induction underscore the need for future methods to incorporate principles such as global optimisation, node-level sparsity, and structure-aware learning. These refinements are essential to better align geometric flexibility with the analytical demands of scenario discovery under deep uncertainty.

7.3 Limitations

While this thesis offers a systematic evaluation of oblique decision tree algorithms, its findings and their generalisability are shaped by four key limitations. These relate to the practical constraints on algorithm selection, the inherent nature of the chosen heuristics, the absence of specific features like sparsity, and the controlled design of the experimental benchmarks.

Algorithmic Scope and Implementation Accessibility

The selection of candidate algorithms was significantly constrained by the practical requirement for accessible Python implementations suitable for the unified benchmarking framework. This necessity created a selection bias, narrowing the scope of the investigation to a specific class of greedy, top-down oblique tree algorithms. Many promising methods, particularly those employing non-greedy optimisation or explicit sparsity constraints, had to be excluded because they were only available as R packages or lacked well-documented source code that could be integrated into the study. Consequently, the evaluation precludes a direct comparison with more advanced or potentially more interpretable approaches.

Reliance on Greedy Induction Heuristics

A direct consequence of the constrained selection is that all evaluated algorithms rely on greedy, node-wise induction strategies. At each node, the algorithm selects the split that locally minimises impurity, without regard for the global structure of the scenario partition. This is a fundamental limitation of the current evaluation, as this heuristic frequently led to fragmented and unintuitive partitions, even when applied to simple geometric shapes such as the rotated rectangle. Recent non-greedy approaches, such as the Tree Alternating Optimisation method of Carreira-Perpiñán and Tavallali (2018) (which will be available in the summer of 2025), perform global optimisation of tree parameters and show clear potential to overcome these structural limitations. The results of this thesis, therefore, primarily reflect the constraints of greedy oblique induction, not the full potential of more advanced, globally optimised approaches.

Absence of Split-Level Sparsity

The evaluated algorithms use dense, full-dimensional splits, which fundamentally limits their interpretability in high-dimensional settings. While a global feature selection step was applied in Phase 3 to mitigate this, this approach differs fundamentally from the more adaptive split-level sparsity available in some other methods. In complex policy models where different uncertainties may drive outcomes (in different regions of the input space), a single global subset of features is too rigid as was seen on the EU energy model. Sparse oblique trees, such as `oblique.tree` (Truong, 2013), ODRF (Y. Liu & Xia, 2025), and Tree Alternating Optimisation (Carreira-Perpiñán & Tavallali, 2018), can select a small, relevant subset of features for each individual split, allowing for more transparent and subspace-specific rules. The absence of such methods from this study means the conclusions on interpretability are necessarily confined to the non-sparse trees.

Experimental Design Constraints

Finally, the experimental design itself involved simplifying assumptions that affect generalisability. During Phase 1, algorithms were evaluated using a single, fixed parameter configuration to ensure a fair comparison across a wide range of benchmarks. However, this design choice provides a conservative, baseline comparison and may have disadvantaged algorithms that are highly sensitive to hyperparameter tuning, such as MOC1 (number of restarts and perturbations) or WODT (number of optimisation iterations). In addition, dimensional noise consisted of statistically independent, uniformly random features. This does not capture the full complexity of real-world policy models, which often exhibit correlation between input variables.

7.4 Directions for Future Research

The findings of this thesis, while cautionary, should not be interpreted as a final verdict on the potential of all oblique decision trees for scenario discovery. Instead, they clarify the specific challenges that must be overcome. The consistent failure of the tested greedy, non-sparse algorithm points toward a clear and promising direction for future research, focused on a new generation of algorithms that align better with the explicit goals of scenario discovery. The following research directions outline a potential path forward.

Embracing Non-Greedy, Globally Optimised Induction

A core limitation identified was the algorithm's reliance on greedy, node-wise induction, which led to fragmented and suboptimal partitions. Future work should therefore explore non-greedy induction strategies that optimise the tree structure globally. Promising candidates include methods like Tree Alternating Optimisation (Carreira-Perpiñán & Tavallali, 2018), which iteratively refines the entire tree's parameters and structure. By optimising for a globally coherent partition rather than local purity, such methods could potentially overcome the fragmentation issues observed in this study and produce more meaningful scenarios.

Incorporating Split-Level Sparsity

This research highlighted the critical weakness of using dense, full-dimensional splits, which severely limits interpretability in high-dimensional models. A crucial next step is to evaluate oblique tree algorithms that incorporate explicit, split-level sparsity constraints. Rather than relying on a separate, global feature selection step, these methods should use techniques like L_1 regularisation or feature selection penalties during the split-finding process. This would enable dynamic, local feature selection at each node, allowing the algorithm to identify the most influential variables for a specific region of the input space while ensuring the resulting decision rules remain simple and transparent. Algorithms like `oblique.tree` (Truong, 2013), ODRF (Y. Liu & Xia, 2025), and Tree Alternating Optimisation (Carreira-Perpiñán & Tavallali, 2018), could be useful for this.

Developing Advanced Visualisation Tools for Oblique Splits

Even a well-structured oblique tree that is globally optimised and sparsely parameterised, is likely to pose a communication challenge, as its decision rules are expressed as linear combinations of input variables rather than simple thresholds. This complexity creates a significant barrier to adoption in policy-facing applications, where interpretability and transparency are essential. Future research should therefore prioritise the development of (visualisation) techniques that make high-dimensional oblique splits

more accessible. Drawing on methods from eXplainable AI (XAI), such tools could include interactive projections of decision boundaries or sensitivity diagnostics that highlight the most influential features within each split. These techniques would help bridge the gap between algorithmic flexbility and the communicative clarity required for effective decision support.

Extending the Benchmarking Framework

The methodological framework and open-source package developed in this thesis provide a robust foundation for future comparative work. This infrastructure should be leveraged to systematically evaluate the more advanced non-greedy and sparse algorithms as they become available in accessible implementations. By testing these next-generation methods against the same benchmarks, the research community can build a cumulative understanding of which algorithmic designs are truly suited to reconciling geometric flexibility with the demands of policy-relevant analysis.

8 Conclusion

The core aim of this thesis was to assess whether oblique decision tree algorithms can enhance scenario discovery relative to established methods. This aim was operationalised through three sub-questions, each addressed by the empirical work presented across Phases 1 to 3. The following answers synthesise the main findings in relation to each sub-question, after which the main research question will be answered.

SQ1: Which oblique decision tree algorithm demonstrates the best performance on benchmark datasets?

The first phase of the research identified HHCART(D) as the most suitable oblique decision tree algorithm from a set of six candidates. This selection was the result of a two-stage benchmarking process on a novel set of synthetic geometric problems. In the first stage, HHCART(D), HHCART(A), and MOC1 were identified as top performers based on their ability to achieve high coverage and density at shallow tree depths. From this group, HHCART(A) was excluded due to its excessive runtime, which was not justified by any corresponding improvement in performance. In the second stage, which stress-tested the remaining candidates for robustness against irrelevant features and sample size, HHCART(D) proved superior. While the performance of MOC1 degraded sharply as dimensional noise was added, HHCART(D) maintained stable and high-quality results. This robustness to noise, a critical property for high-dimensional policy analysis, made it the strongest candidate for further evaluation.

SQ2: How does the selected oblique decision tree algorithm compare to established scenario discovery methods when applied to benchmark datasets?

The second phase showed that, even after applying regularisation, HHCART(D) did not offer the consistent performance or interpretability advantage over established methods on benchmark problems that was expected. On the rotated rectangle benchmark, HHCART(D) was clearly outperformed; PCA-PRIM provided the best geometric match and statistical performance, while PRIM yielded the most interpretable result. In contrast, the regularised HHCART(D) produced a fragmented and unintuitive partition. On the more complex rotated barbell benchmark, HHCART(D) achieved higher coverage than PRIM and PCA-PRIM, but only at the cost of lower density and substantially more complex partitions. Across these tests, its theoretical flexibility did not translate into strong coverage and density gains, while its outputs were consistently harder to understand than those from established methods.

SQ3: How does the selected oblique decision tree algorithm compare to established scenario discovery methods in identifying relevant and interpretable regions within a high-dimensional policy model?

In the final phase, applied to the high-dimensional EU energy system model, HHCART(D) again failed to provide a clear advantage over established methods. While it could match the statistical performance of scenarios identified by PRIM and CART, this came at the cost of severe fragmentation. For instance,HHCART(D) required eight distinct subspaces with linear boundaries to achieve the same coverage and density as a single, easily interpretable box from PRIM. Moreover, the algorithm failed to leverage its core geometric flexibility in any meaningful way. A detailed inspection of the resulting tree revealed that its splits, while technically oblique, were so heavily dominated by a single variable (with weights near 1.0) that they were functionally indistinguishable from simple axis-aligned thresholds.

Main Research Question: To what extent can oblique decision tree algorithms improve scenario discovery compared to established methods?

The findings of this thesis demonstrate that the tested oblique decision tree algorithm, HHCART(D), does not offer a clear or substantial improvement over established scenario discovery methods. The algorithm's theoretical potential for greater geometric flexibility was not realized in practice and failed to justify the significant loss of interpretability.

In both benchmark and high-dimensional applications, HHCART(D) produced more fragmented and less transparent scenario descriptions than PRIM or CART, without offering any consistent coverage and density improvement. Most critically, in the high-dimensional policy setting, it defaulted to nearly axis-aligned splits, undermining its core theoretical strength.

For policy-relevant scenario discovery, where interpretability is paramount, such trade-offs are untenable. This research suggests that future oblique decision tree approaches would require fundamentally improved split search strategies and explicit mechanisms to promote sparse and interpretable outputs. Whether such approaches can ultimately provide a practical advantage over established methods remains an open question.

Bibliography

Amer, M., Daim, T. U., & Jetter, A. (2013). A review of scenario planning. *Futures*, *46*, 23–40. https://doi.org/10.1016/j.futures.2012.10.003

Bankes, S. (1993). Exploratory modeling for policy analysis. *Operations research*, *41*(3), 435–449. https://doi.org/10.1287/opre.41.3.435

Bertsimas, D., & Dunn, J. (2017). Optimal classification trees. *Machine Learning*, *106*, 1039–1082. https://doi.org/10.1007/s10994-017-5633-9

Blanquero, R., Carrizosa, E., Molero-Río, C., & Morales, D. (2020). Sparsity in optimal randomized classification trees. *ArXiv*, *284*, 255–272. https://doi.org/10.1016/j.ejor.2019.12.002

Bradfield, R., Wright, G., Burt, G., Cairns, G., & Van Der Heijden, K. (2005). The origins and evolution of scenario techniques in long range business planning. *Futures*, *37*(8), 795–812. https://doi.org/10.1016/j.futures.2005.01.003

Breiman, L., Friedman, J. H., Olshen, R. A., & Stone, C. J. (1984). *Classification and regression trees*. Routledge. https://doi.org/10.1201/9781315139470

Breiman, L. (2001). Random forests. *Machine Learning*, *45*, 5–32. https://doi.org/10.1023/A:1010933404324

Bryant, B. P., & Lempert, R. J. (2010). Thinking inside the box: A participatory, computer-assisted approach to scenario discovery. *Technological Forecasting and Social Change*, *77*(1), 34–49. https://doi.org/10.1016/j.techfore.2009.08.002

Cañete-Sifuentes, L., Monroy, R., & Medina-Pérez, M. A. (2021). A review and experimental comparison of multivariate decision trees. *IEEE Access*, *9*, 110451–110479. https://doi.org/10.1109/ACCESS.2021.3102239

Carreira-Perpiñán, M. A., & Tavallali, P. (2018). Alternating optimization of decision trees, with application to learning sparse oblique trees. *Advances in Neural Information Processing Systems 31 (NeurIPS 2018)*, 1211–1221. https://papers.nips.cc/paper_files/paper/2018/hash/185c29dc24325934ee377cfda20e414c-Abstract.html

Constantino, S. M., & Weber, E. U. (2021). Decision-making under the deep uncertainty of climate change: The psychological and political agency of narratives. *Current opinion in psychology*, *42*, 151–159. https://doi.org/10.1016/j.copsyc.2021.11.001

Costa, V. G., & Pedreira, C. E. (2023). Recent advances in decision trees: An updated survey. *Artificial Intelligence Review*, *56*(5), 4765–4800. https://doi.org/10.1007/s10462-022-10275-5

Dalal, S., Han, B., Lempert, R., Jaycocks, A., & Hackbarth, A. (2013). Improving scenario discovery using orthogonal rotations. *Environmental Modelling & Software*, *48*, 49–64. https://doi.org/10.1016/j.envsoft.2013.05.013

Forrester, J. W. (1997). Industrial dynamics. *Journal of the Operational Research Society*, *48*(10), 1037–1041. https://doi.org/10.1057/palgrave.jors.2600946

Friedman, J. H., & Fisher, N. I. (1999). Bump hunting in high-dimensional data. *Statistics and Computing*, *9*(2), 123–143. https://doi.org/10.1023/A:1008894516817

Gerst, M. D., Wang, P., & Borsuk, M. E. (2013). Discovering plausible energy and economic futures under global change using multidimensional scenario discovery. *Environmental Modelling & Software*, *44*, 76–86. https://doi.org/10.1016/j.envsoft.2012.09.001

Geurts, P., Ernst, D., & Wehenkel, L. (2006). Extremely randomized trees. *Machine Learning*, *63*, 3–42. https://doi.org/10.1007/s10994-006-6226-1

Greeven, S., Kraan, O., Chappin, É. J., & Kwakkel, J. H. (2016). The emergence of climate change mitigation action by society: An agent-based scenario discovery study. *Journal of Artificial Societies and Social Simulation*, *19*(3). https://doi.org/10.18564/jasss.3134

Groves, D. G., & Lempert, R. J. (2007). A new analytic method for finding policy-relevant scenarios. *Global Environmental Change*, *17*(1), 73–85. https://doi.org/10.1016/j.gloenvcha.2006.11.006

Guivarch, C., Rozenberg, J., & Schweizer, V. (2016). The diversity of socio-economic pathways and CO2 emissions scenarios: Insights from the investigation of a scenarios database. *Environmental Modelling & Software*, *80*, 336–353. https://doi.org/10.1016/j.envsoft.2016.03.006

Guyon, I., & Elisseeff, A. (2003). An introduction to variable and feature selection. *Journal of machine learning research*, *3*(Mar), 1157–1182. https://doi.org/10.5555/944919.944968

Haasnoot, M., Kwakkel, J. H., Walker, W. E., & Ter Maat, J. (2013). Dynamic adaptive policy pathways: A method for crafting robust decisions for a deeply uncertain world. *Global Environmental Change*, *23*(2), 485–498. https://doi.org/10.1016/j.gloenvcha.2012.12.006

Haasnoot, M., Middelkoop, H., Van Beek, E., & Van Deursen, W. (2011). A method to develop sustainable water management strategies for an uncertain future. *Sustainable Development*, *19*(6), 369–381. https://doi.org/10.1002/sd.438

Hadjimichael, A., Schlumberger, J., & Haasnoot, M. (2024). Data visualisation for decision making under deep uncertainty: Current challenges and opportunities. *Environmental Research Letters*, *19*(11), 111011. https://doi.org/10.1088/1748-9326/ad858b

Hadjisotiriou, S., Marchau, V., Walker, W., & Rikkert, M. O. (2023). Decision making under deep uncertainty for pandemic policy planning. *Health policy*, *133*, 104831. https://doi.org/10.1016/j.healthpol.2023.104831

Halim, R. A., Kwakkel, J. H., & Tavasszy, L. A. (2016). A scenario discovery study of the impact of uncertainties in the global container transport system on european ports. *Futures*, *81*, 148–160. https://doi.org/10.1016/j.futures.2015.09.004

Hamarat, C., Kwakkel, J. H., Pruyt, E., & Loonen, E. T. (2014). An exploratory approach for adaptive policymaking by using multi-objective robust optimization. *Simulation Modelling Practice and Theory*, *46*, 25–39. https://doi.org/10.1016/j.simpat.2014.02.008

Hamarat, C., Kwakkel, J. H., & Pruyt, E. (2013). Adaptive robust design under deep uncertainty. *Technological Forecasting and Social Change*, *80*(3), 408–418. https://doi.org/10.1016/j.techfore.2012.10.004

Hamarat, C., Pruyt, E., & Loonen, E. T. (2013). A multi-pathfinder for developing adaptive robust policies in system dynamics. *The 31st international conference of the System Dynamics Society, Cambridge, Massachusetts, USA*, 1–17. https://proceedings.systemdynamics.org/2013/proceed/papers/P1367.pdf

Heath, D., Kasif, S., & Salzberg, S. (1993). Induction of oblique decision trees. *IJCAI*, *1993*, 1002–1007. https://citeseerx.ist.psu.edu/document?repid=rep1%5C&type=pdf%5C&doi=4d3f466fa7e32ab8f11873778893c38558537975

Jaxa-Rozen, M., & Kwakkel, J. H. (2018). Tree-based ensemble methods for sensitivity analysis of environmental models: A performance comparison with sobol and morris techniques. *Environmental Modelling & Software*, *107*, 245–266. https://doi.org/10.1016/j.envsoft.2018.06.011

Kasprzyk, J. R., Nataraj, S., Reed, P. M., & Lempert, R. J. (2013). Many objective robust decision making for complex environmental systems undergoing change. *Environmental Modelling & Software*, *42*, 55–71. https://doi.org/10.1016/j.envsoft.2012.12.007

Kwakkel, J. H. (2017). The exploratory modeling workbench: An open source toolkit for exploratory modeling, scenario discovery, and (multi-objective) robust decision making. *Environmental Modelling & Software*, *96*, 239–250. https://doi.org/10.1016/j.envsoft.2017.06.054

Kwakkel, J. H. (2019). A generalized many-objective optimization approach for scenario discovery. *FUTURES & FORESIGHT SCIENCE*, *1*(2), e8. https://doi.org/https://doi.org/10.1002/ffo2.8

Kwakkel, J. H., Auping, W. L., & Pruyt, E. (2013). Dynamic scenario discovery under deep uncertainty: The future of copper. *Technological Forecasting and Social Change*, *80*(4), 789–800. https://doi.org/10.1016/j.techfore.2012.09.012

Kwakkel, J. H., & Cunningham, S. C. (2016). Improving scenario discovery by bagging random boxes. *Technological Forecasting and Social Change*, *111*, 124–134. https://doi.org/10.1016/j.techfore.2016.06.014

Kwakkel, J. H., & Jaxa-Rozen, M. (2016). Improving scenario discovery for handling heterogeneous uncertainties and multinomial classified outcomes. *Environmental Modelling & Software*, *79*, 311–321. https://doi.org/10.1016/j.envsoft.2015.11.020

Kwakkel, J. H., & Pruyt, E. (2013). Exploratory modeling and analysis, an approach for model-based foresight under deep uncertainty. *Technological Forecasting and Social Change*, *80*(3), 419–431. https://doi.org/10.1016/j.techfore.2012.10.005

Lempert, R. J. (2003). *Shaping the next one hundred years: New methods for quantitative, long-term policy analysis* (tech. rep. No. MR-1626). RAND Corporation. https://www.rand.org/content/dam/rand/pubs/monograph_reports/2007/MR1626.pdf

Lempert, R. J., Bryant, B. P., & Bankes, S. C. (2008). *Comparing algorithms for scenario discovery* (tech. rep. No. WR-557-NSF). RAND Corporation. https://www.rand.org/content/dam/rand/pubs/working_papers/2008/RAND_WR557.pdf

Lempert, R. J., Groves, D. G., Popper, S. W., & Bankes, S. C. (2006). A general, analytic method for generating robust strategies and narrative scenarios. *Management Science*, *52*(4), 514–528. https://doi.org/10.1287/mnsc.1050.0472

Li, X.-B., Sweigart, J. R., Teng, J. T., Donohue, J. M., Thombs, L. A., & Wang, S. M. (2003). Multivariate decision trees using linear discriminants and tabu search. *IEEE transactions on systems, man, and cybernetics-part a: systems and humans*, *33*(2), 194–205. https://doi.org/10.1109/TSMCA.2002.806499

Liu, Q., & Homma, T. (2009). A new computational method of a moment-independent uncertainty importance measure. *Reliability Engineering & System Safety*, *94*(7), 1205–1211. https://doi.org/10.1016/j.ress.2008.10.005

Liu, Y., & Xia, Y. (2025). *ODRF: Oblique decision random forest for classification and regression* (Version 0.0.5) [Computer software]. CRAN package. https://CRAN.R-project.org/package=ODRF

Loh, W.-Y., & Shih, Y.-S. (1997). Split selection methods for classification trees. *Statistica sinica*, 815–840. http://www3.stat.sinica.edu.tw/statistica/oldpdf/A7n41.pdf

Loonen, E., Pruyt, E., & Hamarat, C. (2013). Exploring carbon futures in the EU power sector [Paper P1184]. *Proceedings of the 31st International Conference of the System Dynamics Society*. https://proceedings.systemdynamics.org/2013/proceed/papers/P1184.pdf

Louppe, G. (2014). *Understanding random forests: From theory to practice* [Doctoral dissertation]. University of Liège. https://www.proquest.com/openview/8093a549d407b0ffcfbd9dd868a06d7a/1?pq-origsite=gscholar&cbl=2026366&diss=y

Majumder, T. (2020). *Ensembles of oblique decision trees* [Master's thesis]. The University of Texas at Dallas. https://hdl.handle.net/10735.1/8818

Majumder, T. (2021). *Ensembles of oblique decision trees* [Computer software]. GitHub repository, accessed March 5, 2025. https://github.com/TorshaMajumder/Ensembles_of_Oblique_Decision_Trees

Mao, Q. (2023). *DecisionTreeBaseline: A collection of oblique decision tree algorithms for regression in Python* [Computer software]. GitHub repository, accessed April 9, 2025. https://github.com/maoqiangqiang/DecisionTreeBaseline

Mienye, I. D., & Jere, N. (2024). A survey of decision trees: Concepts, algorithms, and applications. *IEEE access*. https://doi.org/10.1109/ACCESS.2024.3416838

Mingers, J. (1989). An empirical comparison of selection measures for decision-tree induction. *Machine Learning*, *3*, 319–342. https://doi.org/10.1007/BF00116837

Murthy, S. K., Kasif, S., & Salzberg, S. (1994). A system for induction of oblique decision trees. *Journal of artificial intelligence research*, *2*, 1–32. https://doi.org/10.1613/jair.63

Paredes-Vergara, M., Palma-Behnke, R., & Haas, J. (2024). Characterizing decision making under deep uncertainty for model-based energy transitions. *Renewable and Sustainable Energy Reviews*, *192*, 114233. https://doi.org/10.1016/j.rser.2023.114233

Pruyt, E., Cunningham, S., Kwakkel, J. H., & De Bruijn, J. (2014). From data-poor to data-rich: System dynamics in the era of big data. *32nd International Conference of the System Dynamics Society-2014*, 2458–2469. https://proceedings.systemdynamics.org/2014/proceed/papers/P1390.pdf

Saltelli, A., Ratto, M., Andres, T., Campolongo, F., Cariboni, J., Gatelli, D., Saisana, M., & Tarantola, S. (2008). *Global sensitivity analysis: The primer*. John Wiley & Sons. https://www.andreasaltelli.eu/file/repository/A_Saltelli_Marco_Ratto_Terry_Andres_Francesca_Campolongo_Jessica_Cariboni_Debora_Gatelli_Michaela_Saisana_Stefano_Tarantola_Global_Sensitivity_Analysis_The_Primer_Wiley_Interscience_2008_.pdf

Schwartz, P. (1997). *Art of the long view: Planning for the future in an uncertain world*. John Wiley & Sons

Sobol, I. M. (2001). Global sensitivity indices for nonlinear mathematical models and their monte carlo estimates. *Mathematics and computers in simulation*, *55*(1-3), 271–280. https://doi.org/10.1016/S0378-4754(00)00270-6

Steinmann, P. (2018). *Behavior-based scenario discovery: Induction of decision-relevant input subspaces from nonlinear model outputs using time series clustering* [Master's thesis, Delft University of Technology]. http://resolver.tudelft.nl/uuid:cb4fee52-e23f-41d3-9c63-7ca7ec948304

Truong, A. (2013). *oblique.tree: Oblique trees for classification data* (Version 1.1.1) [Computer software]. R package. https://www.rdocumentation.org/packages/oblique.tree

van Droffelaar, I. S. (2020). *A dependent sampling approach to scenario discovery* [Master's thesis]. Delft University of Technology, Faculty of Technology, Policy, Management, Engineering, and Policy Analysis. https://repository.tudelft.nl/record/uuid:425f4402-fd79-4568-be22-08f850b06b36

Wack, P. (1985). Scenarios: Uncharted waters ahead. *Harvard business review*, *63*(5), 72–89. https://hbr.org/1985/09/scenarios-uncharted-waters-ahead

Walker, W. E., Lempert, R. J., & Kwakkel, J. H. (2013). Deep uncertainty. In S. I. Gass & M. C. Fu (Eds.), *Encyclopedia of operations research and management science* (3rd, pp. 395–402). Springer. https://www.researchgate.net/profile/Warren-Walker-3/publication/283999513_Deep_Uncertainty/links/58eccae8458515316aab8296/Deep-Uncertainty

Wickramarachchi, D. C., Robertson, B. L., Reale, M., Price, C. J., & Brown, J. (2016). Hhcart: An oblique decision tree. *Computational Statistics & Data Analysis*, *96*, 12–23. https://doi.org/10.1016/j.csda.2015.11.006

Yang, B.-B., Shen, S.-Q., & Gao, W. (2019). Weighted oblique decision trees. *Proceedings of the AAAI conference on artificial intelligence*, *33*(01), 5621–5627. https://doi.org/10.1609/aaai.v33i01.33015621

Yildiz, O. T., & Alpaydin, E. (2005). Linear discriminant trees. *International Journal of Pattern Recognition and Artificial Intelligence*, *19*(03), 323–353. https://doi.org/10.1142/S0218001405004125

Zharmagambetov, A., Hada, S. S., Gabidolla, M., & Carreira-Perpinán, M. A. (2021). Non-greedy algorithms for decision tree optimization: An experimental comparison. *2021 International Joint Conference on Neural Networks (IJCNN)*, 1–8. https://doi.org/10.1109/IJCNN52387.2021.9533597

List of Figures

List of Tables

A Code Availability and Software Environment

All computational experiments and visualisations presented in this thesis were conducted using a dedicated Python codebase developed for the benchmarking and evaluation of oblique decision tree algorithms in scenario discovery contexts. The entire codebase is publicly accessible and structured to support full replication of results.

Online Repository

All source code is available via GitHub by clicking the link below:

`Oblique Decision Tree Algorithms for Scenario Discovery`

Core Software Environment

The experiments were conducted in the following software environment:

- **Python:** Version 3.11.0
- **Vensim DSS:** Version 10.3.2
- **PyCharm IDE:** Version 2023.3.7 Professional Edition

Development and experimentation were carried out entirely within PyCharm. Vensim DSS was accessed via the EMA Workbench to run system dynamics simulations, and Python served as the core runtime environment for all algorithmic and analytical components.

Python Environment and Dependency Management

All experiments were conducted within a dedicated Python virtual environment (`.venv/`) using `uv`, a modern and high-performance package manager written in Rust. This tool was selected for its speed, simplicity, and seamless handling of dependency installation, offering a smoother experience than traditional tools such as `pip` or Anaconda.

The following packages were explicitly declared as top-level dependencies. Transitive dependencies (installed automatically) are omitted for brevity:

- `ema_workbench == 2.5.3`
- `graphviz == 0.20.3`
- `ipyparallel == 9.0.1`
- `ipython == 9.3.0`
- `ipywidgets == 8.1.7`
- `jupyter == 1.1.1`
- `matplotlib == 3.10.3`
- `numpy == 2.2.6`
- `pandas == 2.2.3`

- `pyDOE2 == 1.3.0`
- `pydot == 4.0.0`
- `scikit-learn == 1.6.1`
- `scipy == 1.15.3`
- `seaborn == 0.13.2`
- `shapely == 2.1.1`
- `statsmodels == 0.14.4`
- `tqdm == 4.67.1`

To replicate the environment, users may refer to the included `requirements.txt` and initialise a virtual environment using `uv venv` and `uv pip install -r requirements.txt`.

B Benchmark Problem Definitions

This appendix describes how the benchmark problems used in this study are generated. These problems are designed to systematically evaluate the performance of scenario discovery algorithms under controlled geometric conditions. Each shape targets specific limitations of traditional methods such as PRIM, CART, and PCA-PRIM, including their difficulty in handling rotated boundaries, multiple local orientations, or non-linear structures (see Section 3.1.1). The shapes are implemented as parameterised functions that produce binary-labelled datasets. By allowing controlled variation in rotation, curvature, dimensionality, boundary noise, and irrelevant input features, these functions support benchmarking of algorithmic performance and robustness.

All shape functions follow a consistent design logic, ensuring comparability and reproducibility:

- **Sampling:** Each function uses Latin Hypercube Sampling to generate a uniformly distributed set of input points in a unit hypercube. For 2D shapes, sampling occurs in $[0,1]^2$; for 3D shapes, in $[0,1]^3$.
- **Shape Evaluation (with optional rotation):** A shape is first defined in its default axis-aligned form. If rotation is specified, each sampled point is temporarily rotated in the opposite direction before checking whether it lies inside the shape. This combined step allows for rotated shape evaluation without altering the sampling grid. Based on the inclusion check, each point is assigned a binary label: 1 (inside) or 0 (outside).
- **Adding Boundary Noise (optional):** To simulate noise and uncertainty caused by high-dimensional conditions common in policy models boundary noise was added. Points outside the shape (label 0) may be flipped to inside (label 1) based on their proximity to the shape boundary. The probability of flipping decays exponentially with distance to the boundary:

  $$P_{\text{flip}} = \exp\left(-\frac{d}{\lambda}\right)$$

  where d is the distance to the boundary and λ is the label noise parameter. Points already inside the shape are never flipped.

The shape generation system is implemented using a Python `ipywidgets` interface within a Jupyter notebook, enabling users to interactively adjust parameters. Each change triggers a re-execution of the full generation pipeline, including sampling, shape evaluation, and optional noise injection. When the user chooses to save the configuration, the system automatically exports all output formats: a combined `.csv` file containing coordinates and labels, separate `x.csv` and `y.csv` files for features and labels, and a `.png` image visualising the shape and its classifications.

The remainder of this appendix presents the formulation of each stylised shape used in this study. Each section defines the geometric structure, inclusion rules, and adjustable parameters for the shape. The last part of this appendix shows the paramter settings used for all shapes.

2D Rectangle

This shape defines a rectangular decision region in two-dimensional space. The rectangle is centred at a specified location and defined by its width and height. It may optionally be rotated around its centre to produce an oblique orientation.

Parameters

- **Number of samples:** Total number of points generated in $[0,1]^2$
- **Centre position:** Coordinates of the rectangle's centre
- **Size:** Width and height of the rectangle
- **Rotation angle:** Counterclockwise angle (in degrees) used to rotate the rectangle around its centre

Mathematical Definition and Inclusion Procedure

Let $x = (x_1, x_2) \in [0,1]^2$ be a sampled point and $c = (c_1, c_2)$ the rectangle's centre. To evaluate whether the point lies inside the rotated rectangle, the following transformation is applied:

$$x' = R_{-\theta}(x - c)$$

where $R_{-\theta}$ is the inverse (clockwise) rotation matrix for angle θ, and $x' = (x'_1, x'_2)$ is the rotated point expressed relative to the centre.

The rectangle has width w and height h. A point is considered inside the shape if:

$$|x'_1| \leq \frac{w}{2} \quad \text{and} \quad |x'_2| \leq \frac{h}{2}$$

Based on this rule, the binary classification function is defined as:

$$f(x) = \begin{cases} 1 & \text{if } x' \text{ satisfies both conditions above} \\ 0 & \text{otherwise} \end{cases}$$

2D Radial Segment

This shape defines a decision region in two-dimensional space based on a circular arc segment. The region includes all points that lie between an inner and outer radius and within a specified angular span, forming a wedge-shaped section of a ring. It is designed to test whether algorithms can detect curved, non-axis-aligned regions with both angular and radial boundaries.

Parameters

- **Number of samples:** Total number of points generated in $[0,1]^2$
- **Centre position:** Coordinates of the arc's centre
- **Inner and outer radii:** Radial bounds defining the minimum and maximum distance from the centre
- **Arc angle:** Angular span of the segment in degrees
- **Rotation angle:** Counterclockwise angle (in degrees) defining the central direction of the arc

Mathematical Definition and Inclusion Procedure

Let $x = (x_1, x_2) \in [0,1]^2$ be a sampled point, and let $c = (c_1, c_2)$ be the arc's centre. Define the relative position vector:

$$\vec{r} = x - c$$

and let $d = \|\vec{r}\|_2$ be the Euclidean distance from the centre. The angle of the point relative to the centre is:

$$\phi = \operatorname{atan2}(r_2, r_1) \mod 2\pi$$

Let θ be the arc's central direction (in radians), obtained from the rotation angle, and let α be the arc span (in radians). The arc bounds are defined as:

$$\phi_{\min} = (\theta - \frac{\alpha}{2}) \mod 2\pi, \quad \phi_{\max} = (\theta + \frac{\alpha}{2}) \mod 2\pi$$

The angular condition is satisfied if:

$$\phi_{\min} < \phi_{\max} \Rightarrow \phi_{\min} \leq \phi \leq \phi_{\max} \qquad \phi_{\min} > \phi_{\max} \Rightarrow \phi \geq \phi_{\min} \text{ or } \phi \leq \phi_{\max}$$

A point is classified as inside the shape if both radial and angular constraints hold:

$$r_{\text{inner}} \leq d \leq r_{\text{outer}} \quad \text{and} \quad \phi \in \text{arc sector}$$

The binary classification function is:

$$f(x) = \begin{cases} 1 & \text{if both radial and angular conditions are satisfied} \\ 0 & \text{otherwise} \end{cases}$$

2D Barbell

This shape defines a decision region in two-dimensional space consisting of two circular regions connected by a horizontal rectangular bridge. The entire structure is centred at a specified point and can be rotated. It is designed to test an algorithm's ability to detect both compact and extended regions, as well as weak connections between separate components.

Parameters

- **Number of samples:** Total number of points generated in $[0,1]^2$
- **Centre position:** Coordinates of the barbell's overall centre
- **Barbell length:** Distance between the centres of the two circles
- **Circle radius:** Radius of each of the two circular end regions
- **Connector thickness:** Half the vertical thickness of the rectangular bridge
- **Rotation angle:** Counterclockwise angle (in degrees) used to rotate the entire barbell around its centre

Mathematical Definition and Inclusion Procedure

Let $x = (x_1, x_2) \in [0,1]^2$ be a sampled point and let $c = (c_1, c_2)$ be the centre of the barbell. To evaluate whether the point lies inside the shape, the following inverse transformation is applied:

$$x' = R_{-\theta}(x - c)$$

where $R_{-\theta}$ is the clockwise rotation matrix and $x' = (x'_1, x'_2)$ is the transformed point expressed in the unrotated frame.

Let L be the barbell length, r the circle radius, and t the connector thickness. Define:

- Circle A centre: $(-L/2, 0)$
- Circle B centre: $(+L/2, 0)$
- Rectangle spanning from $x'_1 \in [-L/2, +L/2]$ and $x'_2 \in [-t, +t]$

The point x is labelled as inside the shape if it satisfies any of the following conditions:

- $\|x' - (-L/2, 0)\| \leq r$
- $\|x' - (+L/2, 0)\| \leq r$
- $|x'_1| \leq \frac{L}{2}$ and $|x'_2| \leq t$

The binary classification function is:

$$f(x) = \begin{cases} 1 & \text{if the point lies in either circle or the rectangle} \\ 0 & \text{otherwise} \end{cases}$$

2D Sine Wave

This shape defines a decision region in two-dimensional space based on a bounded vertical band around a sine curve. The region is restricted to a specified horizontal interval and includes points whose vertical distance to the sine curve is within a fixed threshold. This shape tests an algorithm's ability to follow smooth, periodic, and spatially bounded patterns.

Parameters

- **Number of samples:** Total number of points generated in $[0,1]^2$
- **Horizontal range:** Interval $(x_{\min}, x_{\max})$ specifying where the sine wave is active
- **Vertical offset:** Baseline vertical position of the sine curve
- **Amplitude:** Height of the sine wave's peaks above and below the vertical offset
- **Frequency:** Number of full sine cycles over the specified horizontal interval
- **Band thickness:** Maximum vertical deviation allowed from the sine curve
- **Rotation angle:** Counterclockwise angle (in degrees) used to rotate the sine wave around its centre

Mathematical Definition and Inclusion Procedure

Let $x = (x_1, x_2) \in [0,1]^2$ be a sampled point. Let $c = \left(\frac{x_{\min}+x_{\max}}{2}, \text{offset}\right)$ be the centre of the sine wave band. If rotation is applied, the point is first transformed via:

$$x' = R_{-\theta}(x - c) + c$$

where $R_{-\theta}$ is the inverse (clockwise) rotation matrix, and $x' = (x'_1, x'_2)$.

Define the scaled sine curve:

$$f(x'_1) = \text{offset} + A \cdot \sin\left(2\pi f \cdot \frac{x'_1 - x_{\min}}{x_{\max} - x_{\min}}\right)$$

where A is the amplitude and f is the frequency.

The point x is labelled as inside the shape if:

$$x_{\min} \le x'_1 \le x_{\max} \quad \text{and} \quad \left|x'_2 - f(x'_1)\right| < \delta$$

where δ is the vertical half-thickness of the band.

The binary classification function is:

$$f(x) = \begin{cases} 1 & \text{if both the horizontal range and band thickness conditions are satisfied} \\ 0 & \text{otherwise} \end{cases}$$

2D Star

This shape defines a decision region in two-dimensional space based on a regular star polygon. The star is created by alternating between inner and outer radii to form sharp tips and concave indentations, resulting in a non-convex, symmetric geometry. The entire shape can be rotated and translated. This shape tests an algorithm's ability to resolve complex, highly non-linear, and radially structured boundaries.

Parameters

- **Number of samples:** Total number of points generated in $[0,1]^2$
- **Centre position:** Coordinates of the star's centre
- **Number of points:** Number of tips (outer vertices) of the star
- **Outer radius:** Distance from the centre to each outer vertex
- **Inner radius:** Distance from the centre to each inner vertex
- **Star size:** Scaling factor applied to both radii
- **Rotation angle:** Counterclockwise angle (in degrees) used to rotate the star polygon

Mathematical Definition and Inclusion Procedure

Let $x = (x_1, x_2) \in [0,1]^2$ be a sampled point and $c = (c_1, c_2)$ the centre of the star. The star is constructed as a polygon with $2n$ vertices, where n is the number of tips. The vertices alternate between the outer radius r_{outer} and inner radius r_{inner}, distributed evenly along the unit circle. The vertex positions in polar coordinates are:

$$\text{angle}_i = \frac{2\pi i}{2n}, \quad r_i = \begin{cases} r_{\text{outer}} & \text{if } i \text{ is even} \\ r_{\text{inner}} & \text{if } i \text{ is odd} \end{cases}$$

Each vertex is then scaled by the star size parameter and rotated by angle θ (converted to radians), using inverse rotation:

$$v_i = R_{-\theta}(r_i \cdot (\cos(\text{angle}_i), \sin(\text{angle}_i))) + c$$

These rotated and shifted vertices define the star polygon $\mathcal{P}$. A point is labelled as inside if it lies within the polygon:

$$f(x) = \begin{cases} 1 & \text{if } x \in \mathcal{P} \\ 0 & \text{otherwise} \end{cases}$$

3D Radial Segment

This shape defines a decision region in three-dimensional space based on a toroidal arc segment. The region consists of a curved tube (torus cross-section) with a specified radial thickness and angular span. It is designed to evaluate whether algorithms can capture curved, tubular regions with both spatial curvature and directional constraints.

Parameters

- **Number of samples:** Total number of points generated in $[0,1]^3$
- **Centre position:** Coordinates of the torus centre
- **Outer radius:** Distance from the centre to the centreline of the toroidal tube
- **Inner radius:** Controls the tube thickness as *outer_radius* minus *inner_radius*
- **Arc angle:** Angular span (in degrees) limiting the segment to a portion of the full torus
- **Rotation angles:** Three Euler angles (in degrees) to rotate the torus around the x_1, x_2, and x_3 axes

Mathematical Definition and Inclusion Procedure

Let $x = (x_1, x_2, x_3) \in [0,1]^3$ be a sampled point and $c = (c_1, c_2, c_3)$ the centre of the torus. The point is transformed using inverse Euler rotations:

$$x' = R_{-\theta}(x - c)$$

where $R_{-\theta}$ is the composite 3D rotation matrix using negative values of the input angles for each axis. Denote the rotated point as $x' = (x'_1, x'_2, x'_3)$.

Let R be the outer radius and $r = R - r_{\text{inner}}$ the tube radius. Define:

$$\rho = \sqrt{{x'_1}^2 + {x'_2}^2}, \quad \phi = \text{atan2}(x'_2, x'_1)$$

$$d_{\text{torus}} = \sqrt{(\rho - R)^2 + {x'_3}^2} - r$$

The region of interest includes points satisfying: - $d_{\text{torus}} \leq 0$ (i.e., inside the toroidal tube) - $\phi \in [\pi - \alpha/2, \pi + \alpha/2]$ where α is the arc span in radians (if *arc_span_degrees* is less than 360)

Points outside the angular bounds are excluded by assigning infinite distance.

The classification function is:

$$f(x) = \begin{cases} 1 & \text{if } d_{\text{torus}} \leq 0 \text{ and } \phi \text{ within arc bounds} \\ 0 & \text{otherwise} \end{cases}$$

3D Barbell

This shape defines a decision region in three-dimensional space consisting of two spherical regions connected by a cylindrical bridge. The shape is symmetric, centred in space, and can be rotated around all three axes. It is used to evaluate whether algorithms can resolve compound regions composed of both curved and linear boundaries.

Parameters

- **Number of samples:** Total number of points generated in $[0,1]^3$
- **Centre position:** Coordinates of the barbell's overall centre
- **Barbell length:** Distance between the centres of the two spheres
- **Sphere radius:** Radius of each of the spherical end regions
- **Connector thickness:** Radius of the cylindrical connector (in the x_2-x_3 plane)
- **Rotation angles:** Euler angles (in degrees) applied about the x_1, x_2, and x_3 axes

Mathematical Definition and Inclusion Procedure

Let $x = (x_1, x_2, x_3) \in [0,1]^3$ be a sampled point and $c = (c_1, c_2, c_3)$ the centre of the barbell. Apply inverse rotation to evaluate the point in the canonical (unrotated) frame:

$$x' = R_{-\theta}(x - c)$$

where $R_{-\theta}$ is the inverse 3D rotation matrix and $x' = (x'_1, x'_2, x'_3)$.

Let the two sphere centres in this frame be:

$$s_A = (-L/2, 0, 0), \quad s_B = (L/2, 0, 0)$$

where L is the barbell length.

Let r be the sphere radius, and t the connector thickness. Then: - Distance to each sphere's boundary is:

$$d_A = \|x' - s_A\| - r, \quad d_B = \|x' - s_B\| - r$$

- Distance to the cylinder is approximated as:

$$d_{\text{cyl}} = \max\left(|x_1'| - \frac{L}{2}, \sqrt{{x_2'}^2 + {x_3'}^2} - t\right)$$

The signed distance to the barbell is:

$$d(x) = \min(d_A, d_B, d_{\text{cyl}})$$

A point is labelled as inside the shape if $d(x) \leq 0$.

The binary classification function is:

$$f(x) = \begin{cases} 1 & \text{if the point lies in either sphere or the connecting cylinder} \\ 0 & \text{otherwise} \end{cases}$$

3D Saddle

This shape defines a decision region in three-dimensional space based on a quadratic saddle surface. The region includes points that lie within a specified vertical thickness around a warped, hyperbolic surface. It is designed to evaluate whether algorithms can accurately capture non-planar, curved geometries with opposing curvature along orthogonal axes.

Parameters

- **Number of samples:** Total number of points generated in $[0,1]^3$
- **Centre position:** Coordinates of the saddle surface midpoint
- **Saddle height:** Total vertical height over which the saddle surface is normalised
- **Curvature along x_1:** Controls upward curvature in the x_1 direction
- **Curvature along x_2:** Controls downward curvature in the x_2 direction
- **Surface thickness:** Maximum vertical distance allowed from the saddle surface
- **Rotation angles:** Euler angles (in degrees) applied around the x_1, x_2, and x_3 axes

Mathematical Definition and Inclusion Procedure

Let $x = (x_1, x_2, x_3) \in [0,1]^3$ be a sampled point and $c = (c_1, c_2, c_3)$ the centre of the saddle. To evaluate inclusion, apply the inverse of the specified Euler rotations:

$$x' = R_{-\theta}(x - c) + c$$

where $R_{-\theta}$ is the inverse composite rotation matrix, and $x' = (x_1', x_2', x_3')$ is the transformed point.

The unnormalised saddle surface is defined as:

$$z_{\text{saddle, raw}} = a(x_1' - c_1)^2 - b(x_2' - c_2)^2$$

where a and b are curvature coefficients in the x_1 and x_2 directions, respectively.

This raw surface is linearly scaled to span the target vertical range $[c_3 - h/2,\ c_3 + h/2]$, where h is the saddle height. Let $z_{\min}$ and $z_{\max}$ be the minimum and maximum values of the raw surface. The normalised saddle height is:

$$z_{\text{saddle}} = \left(\frac{z_{\text{saddle, raw}} - z_{\min}}{z_{\max} - z_{\min}}\right) \cdot h + \left(c_3 - \frac{h}{2}\right)$$

A point is classified as inside the region if its distance to the saddle surface in the x_3 dimension is less than or equal to the allowed thickness t:

$$f(x) = \begin{cases} 1 & \text{if } |x_3' - z_{\text{saddle}}| \leq t \\ 0 & \text{otherwise} \end{cases}$$

Parameter Settings

The parameter settings for all of the shapes are shown in the Table B.1.

Table B.1
Parameter Settings Used for Benchmark Shape Generation

Shape	Parameter Settings
2D Rectangle	*num_samples* = 10,000; *center* = (0.5, 0.5); *ribs* = (0.5, 0.5); *rotation* = 45; *width* = 0.5; *length* = 0.5°
2D Radial Segment	*num_samples* = 10,000; *center* = (0.5, 0.5); *outer_radius* = 0.4; *inner_radius* = 0.2; *arc_span_degrees* = 300°; *rotation* = 90°
2D Barbell	*num_samples* = 10,000; *center* = (0.5, 0.5); *barbell_length* = 0.6; *sphere_radius* = 0.2; *connector_thickness* = 0.04; *rotation* = 50°
2D Sine Wave	*num_samples* = 10,000; *x_range* = (0.1, 0.9); *vertical_offset* = 0.5; *amplitude* = 0.2; *frequency* = 0.5; *thickness* = 0.10; *rotation* = 0°
2D Star	*num_samples* = 10,000; *center* = (0.5, 0.5); *num_points* = 5; *outer_radius* = 0.4; *inner_radius* = 0.2; *star_size* = 1.0; *rotation* = 0°
3D Saddle Surface	*num_samples* = 10,000; *center* = (0.5, 0.5, 0.5); *saddle_height* = 0.5; *curve_sharpness_x1* = 1.0; *curve_sharpness_x2* = 1.0; *surface_thickness* = 0.2; *rotation_x1* = 0°; *rotation_x2* = 0°; *rotation_x3* = 0°
3D Radial Segment	*num_samples* = 10,000; *center* = (0.5, 0.5, 0.5); *outer_radius* = 0.4; *inner_radius* = 0.2; *arc_span_degrees* = 300°; *rotation_x1* = 35°; *rotation_x2* = 0°; *rotation_x3* = 60°
3D Barbell	*num_samples* = 10,000; *center* = (0.5, 0.5, 0.5); *barbell_length* = 0.8; *sphere_radius* = 0.25; *connector_thickness* = 0.1; *rotation_x1* = 50°; *rotation_x2* = 50°; *rotation_x3* = 0°

C Algorithms and Implementation

This appendix documents the algorithmic changes made during the course of this study and the parameter settings. Two categories of changes were made to the decision tree models evaluated. The first aimed to restore theoretical alignment with the original formulations presented in the literature. The second focused on structural adjustments required to ensure compatibility with the unified evaluation framework developed for benchmarking and visualisation. This appendix focuses exclusively on the first category, substantial algorithmic changes affecting the functional logic of the models. Changes related solely to random seed initialisation, internal formatting, or data structure alignment (e.g., conversion to the DecisionTree class) are not discussed here, as they had no impact on model behaviour.

Classification and Regression Tree Segmentor

Initial inspection of the codebase by Majumder (2021), from which several oblique tree algorithms were adapted, revealed that many implementations relied on a simplified thresholding strategy using a MeanSegmentor (i.e., splitting each feature at its mean value). However, the original publications for these methods typically describe the use of a classification and regression tree-style thresholding approach that evaluates all midpoints between the sorted feature values. To address this discrepancy, and to allow more thorough split evaluation between models, a reusable classification and regression tree segmentor was integrated into oblique models to replace the less rigorous mean-based logic.

Modified Oblique Classifier 1

The Oblique Classifier 1 (OC1) model from Majumder (2021) required significant revision for this thesis. The source implementation used a simplified one-shot random vector perturbation, which was inconsistent with the local coordinate search strategy introduced in the original OC1 paper by Murthy et al. (1994). Initially, the algorithm was re-implemented to be fully faithful to the original paper, incorporating its coordinate-wise search and probabilistic acceptance strategies. However, this theoretically correct implementation proved computationally intractable for scenario discovery, suffering from excessively long runtimes on small datasets with only three input dimensions.

To address this, a hybrid version, named Modified Oblique Classifier 1 (MOC1), was developed. MOC1 is designed to balance attributes of OC1 with runtime tractability. It retains the faster full-vector perturbation approach found in the Majumder (2021) implementation but re-introduces the robust multiple restarts from the original Murthy et al. (1994) OC1 algorithm. Furthermore explicit bias-term search was added for each of the full-vector perturbations. This approach creates an algorithm that is more thorough than the simplified source implementation but significantly faster than the theoretically aligned version, making it suitable for scenario discovery. A detailed comparison of the original, adapted, and final MOC1 algorithms is provided in Table C.1.

Householder Classification and Regression Tree

The Householder Classification and Regression Tree (HHCART) was fundamentally re-implemented for this thesis, as the single variant provided by Majumder (2021) diverged from the algorithmic design described by Wickramarachchi et al. (2016). The following major corrections were made. First, the method for generating reflections was overhauled. Instead of performing a Principal Component Analysis (PCA) on the entire dataset at a node, the new implementation computes separate covariance matrices for each class. The eigenvectors from these matrices are then used to construct Householder reflections, which better capture class-specific geometric structures. Second, the split selection process was changed to compare all proposed splits at each node, comparing the best possible oblique split against the best possible axis-aligned split and selecting the superior one. These corrections made it possible to offer two distinct, correctly implemented variants where previously only one (slightly flawed) version existed. The new implementation now provides both HHCART(A), which exhaustively searches over all class-specific eigenvectors, and the more computationally efficient HHCART(D), which uses only the dominant eigenvector.

Table C.1
Comparison of OC1 Variants: Original, Adopted, and Modified

Aspect	OC1 Murthy et al. (1994)	OC1 Majumder (2021)	Modified OC1 (MOC1)
Weight term update	Coordinate-wise perturbation of weights	One-shot random vector perturbation	Full-vector perturbation along random directions
Bias term update	Explicit bias term shifting	Bias fixed during split search	Grid-based bias shifting along hyperplane normal
Acceptance strategy	Probabilistic acceptance of worse splits	Greedy acceptance only	Greedy acceptance only
Split selection	Local impurity minimisation with restarts	Single perturbation chosen	Global impurity minimisation across all trials
Random restarts	Multiple restarts from random hyperplanes	No restarts used	Multiple restarts from random directions
Search space	Oblique hyperplanes in original feature space	Oblique hyperplanes in original space	Oblique hyperplanes in original feature space
Impurity metric	Gini or Twoing	Gini or Twoing	Gini or Twoing
Stochastic elements	Restarts and probabilistic tie-breaking	Single perturbation per node	Random directions, restarts, and bias steps

Randomised Classification and Regression Tree

The original implementation of the Randomised Classification and Regression Tree (RandCART) by Majumder (2021) generated a single randomly rotated hyperplane at each node and applied a simplified thresholding rule using a MeanSegmentor. This approach restricted split selection to a single threshold – the feature mean in the rotated space – and limited the algorithm's ability to identify optimal partitions. To address these shortcomings, two substantive changes were introduced. First, the number of random rotations was made configurable, allowing the model to sample multiple candidate directions and thereby improve search diversity. Second, the thresholding mechanism was upgraded by replacing the MeanSegmentor with a CARTSegmentor, which evaluates all midpoints between sorted feature values. These revisions transformed RandCART into a more rigorous stochastic baseline, capable of more competitive splits.

Ridge Classification and Regression Tree

The Ridge Classification and Regression Tree (RidgeCART) model was adapted from the DecisionTree-Baseline repository by Mao (2023). The original implementation was designed for regression, where at each node, a Ridge model would learn to predict a continuous target value. To make the algorithm suitable for the binary classification needs of scenario discovery, a crucial adaptation was made to this process. Although the Ridge regressor, which minimizes a squared-error loss, is retained, it is now fitted on the binary class labels (0 or 1). By regressing on a binary target, the model effectively learns a linear probability model, and the resulting coefficients are optimised to find a hyperplane that best separates the two classes, rather than to track a continuous outcome. Following this, a second adaptation defines the split selection as a competition: the linear projection derived from these classification-focused coefficients is treated as a new candidate feature. An exhaustive search is then conducted across all original, axis-aligned features and this new projection to find the single best split. This hybrid approach ensures that the model can leverage a learned oblique projection if it proves more effective than any standard axis-aligned split, while retaining the simplicity of axis-aligned splits otherwise. This adaptation transformed the original regression tool into a flexible classifier suitable for this study.

Weighted Oblique Decision Tree

Both the Weighted Oblique Decision Tree (WODT) model was structurally compatible with this study's framework and required no conceptual changes. The WODT model, adapted from Majumder (2021) and originally formulated by Yang et al. (2019), uses an L-BFGS-based optimiser to minimise weighted

impurity and was adopted without modification to its core optimisation logic. Only slight change was applied to ensure consistency with the unified tree structure.

Parameter Settings

All decision tree algorithms were configured to allow unconstrained growth by default, with *min_samples_split* = 2 for all methods, unless otherwise specified. This common setting ensured comparability of split behaviour across models. The following list outlines the remaining algorithm-specific parameters and their configured values:

1. **HHCART Algorithms**: The HHCART algorithms were configured with $tau = 0.05$. This parameter sets a minimum difference between the dominant direction of class variation and the nearest axis-aligned direction. A value of 0.05 means that a direction must deviate by at least 5% to be considered sufficiently oblique. This avoids applying reflections that are nearly axis-aligned, which offer limited benefit and may cause numerical issues.
2. **RandCART**: This algorithm introduces stochasticity by applying random orthogonal rotations to the feature space before conducting axis-aligned split evaluation. The number of such rotations was kept at $n\_rotations = 1$ to maintain RandCART as a benchmark.
3. **WODT**: The Weighted Oblique Decision Tree was configured to consider *all* available features at each node (*max_features* = `'all'`), with a large optimisation budget of $max\_iter = 1{,}000{,}000$ iterations per split to encourage convergence of its weight-learning procedure.
4. **MOC1**: The Modified Oblique Classifier 1 was configured with $n\_restarts = 20$, enabling multiple random initialisations during the weight search phase. The threshold search along the decision boundary used $bias\_steps = 20$ to evaluate the bias axis and identify optimal cut points.
5. **CART**: The baseline CART was used with its default settings.
6. **RidgeCART**: RidgeCART was used with its default settings.

D Scalability and Robustness Analyses

This appendix presents supplementary figures from Phase 1 that support the findings discussed in Chapter 4. These include additional analyses of performance under varying sample sizes and feature dimensionalities applied to the 3D Radial Segment.

Robustness to Sample Size

Figure D.1 displays coverage and density across tree depths for MOC1 and HHCART(D) under varying sample sizes. As visible, the overall performance over tree depth remains similar for both coverage and density across the tested sample sizes. This indicates that varying the number of samples between 1,000 and 10,000 does not substantially alter the algorithms' coverage or density behaviour, suggesting robustness to dataset size.

Runtime Scaling with Sample Size

Figure D.2 compares the runtime behaviour of MOC1 and HHCART(D) as sample size increases. Both algorithms display sublinear scaling with depth, but the profiles differ slightly. HHCART(D) increases more rapidly at shallow depths and levels off beyond depth 6, while MOC1 scales more gradually and continues rising. HHCART(D) is slightly faster at lower sample sizes, whereas MOC1 becomes more efficient at larger scales.

Runtime Scaling with Feature Count

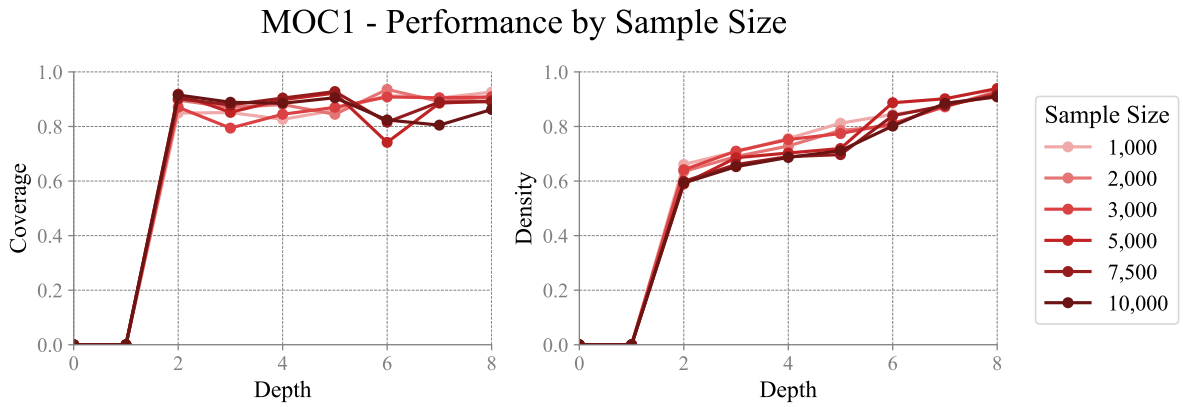
Figure D.3 shows how runtime scales with increasing feature dimensionality for MOC1 and HHCART(D). Both methods exhibit sublinear growth across tree depths, but MOC1 scales more efficiently. At moderate depths such as depth 4, MOC1 maintains lower runtimes, while HHCART(D) becomes progressively slower with increasing dimensionality – eventually requiring roughly 50% more time. These results suggest that MOC1 offers better computational scalability in high-dimensional settings.

Average Feature Count per Split Scaling with Feature Count

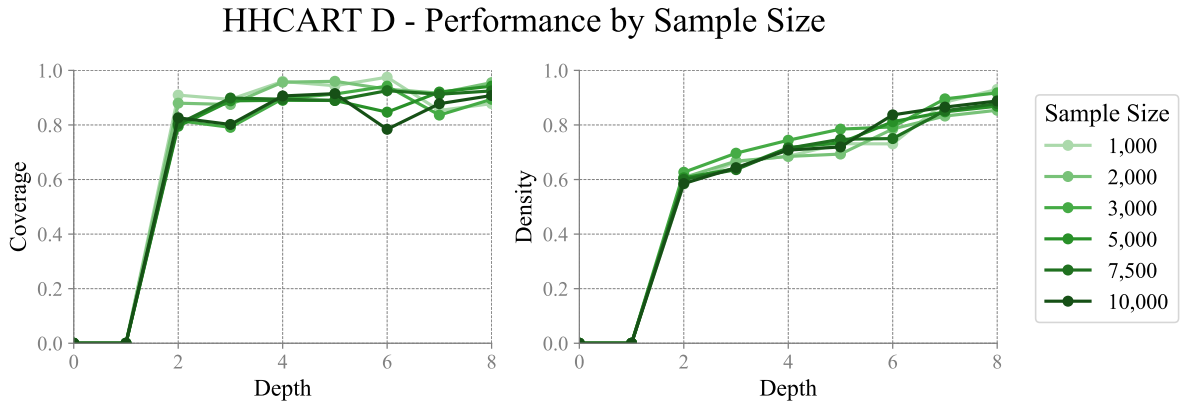
Figure D.4 presents the average number of active features per decision node across tree depths 1 through 8, evaluated under increasing input dimensionalities. Panel (a) displays results for MOC1 and Panel (b) for HHCART(D). For MOC1, feature usage immediately reaches the full dimensionality at depth 1 and remains constant thereafter. This confirms that every split in every tree node consistently involves all input variables, including any added noise features. The result is a fully dense structure in which no sparsity or feature exclusion occurs at any level of the tree.

By contrast, HHCART(D) exhibits a slightly different pattern. As shown in Panel (b), the average number of active features increases sharply at shallow depths, then decreases a bit and stabilises at a quite high fraction of features used. The modest reductions observed at likely arise from the algorithm occasionally selecting axis-aligned splits in the original space, when they yield lower impurity than the Householder-based splits. Such splits involve only one dominant feature and reduce the average feature count across nodes.

Figure D.1
Coverage and density per tree depth (1–8) across increasing sample sizes (1,000–10,000). Panel (a) shows MOC1 and Panel (b) shows HHCART(D). Both methods demonstrate stable performance across sample size, with no systematic changes in coverage or density as dataset size increases.

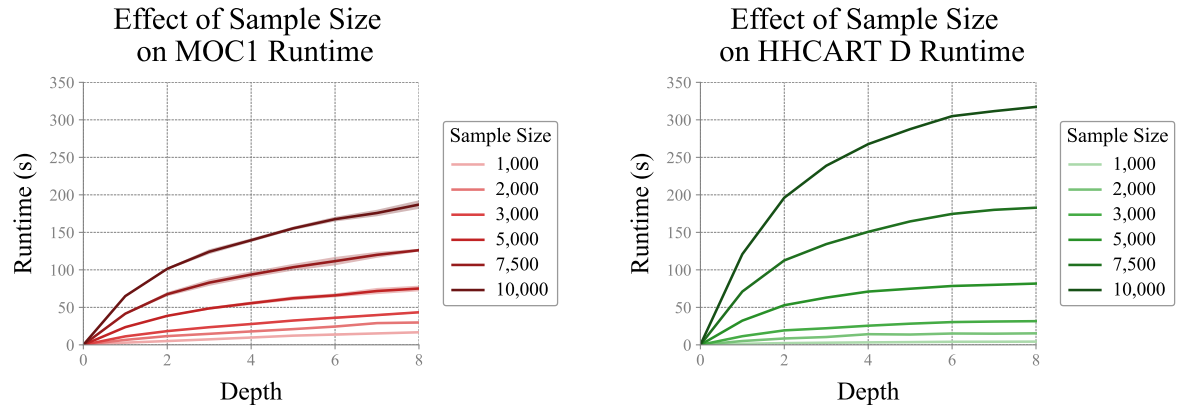


(a)



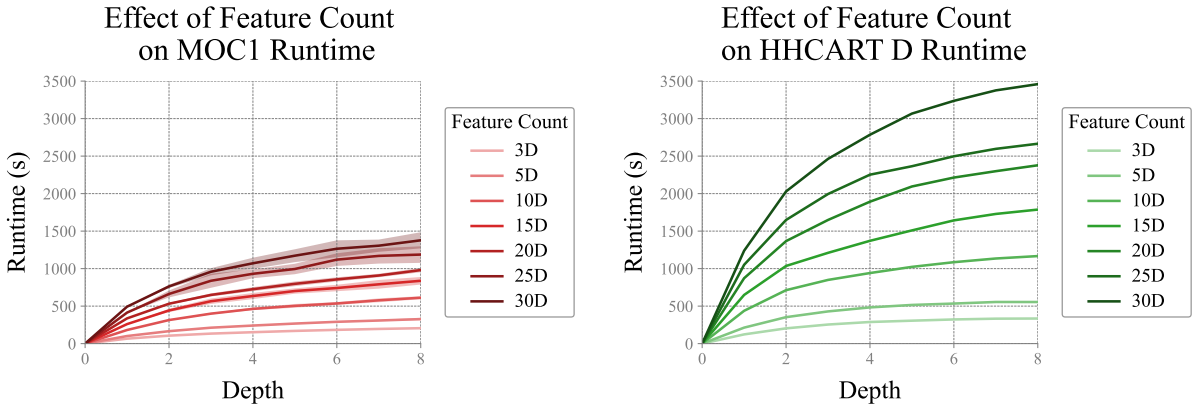
(b)

Figure D.2
Runtime per tree depth (1–8) across increasing sample sizes. Panel (a) shows MOC1 and Panel (b) shows HHCART(D).



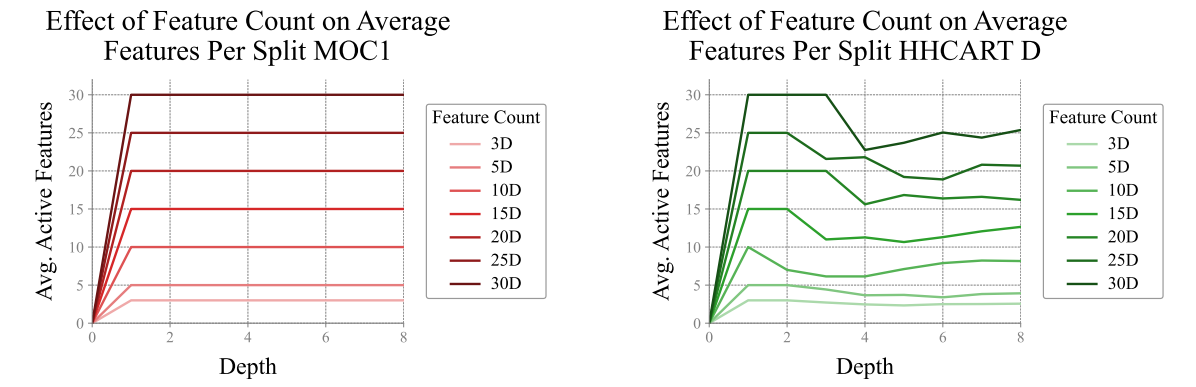
(a) **(b)**

Figure D.3
Runtime per tree depth (1-8) across increasing feature counts. Panel (a) shows MOC1 and Panel (b) shows HHCART(D).



(a) **(b)**

Figure D.4
Average active feature count per decision node per tree depth (1–8) across increasing feature counts. Panel (a) shows MOC1 and Panel (b) shows HHCART(D).



(a) **(b)**

E Greedy Split Construction of HHCART(D)

This appendix presents additional visualisations from Phase 2 that clarify how and why HHCART(D) constructs certain decision boundaries during tree induction. The primary aim is to understand the origin of split misalignment observed in some fitted trees, specifically, why partitions sometimes appear geometrically suboptimal or counterintuitive. These visualisations serve to diagnose how the algorithm's structural heuristics give rise to such outcomes.

The figures are based on the 2D rotated rectangle benchmark with label noise ($lambda = 0.05$). At each internal node, HHCART(D) evaluates several candidate reflections, one for each class, based on the dominant eigenvector of the class-specific covariance matrix, representing that class's primary direction of variance. Each reflection rotates the data so that its dominant direction aligns with the standard basis axis that is farthest from it. In each rotated space, including the original unrotated space (the identity transformation), the algorithm performs an axis-aligned threshold search to minimise the Gini impurity. The reflection–split pair that yields the lowest impurity among all candidates is selected and used to define the decision boundary at that node.

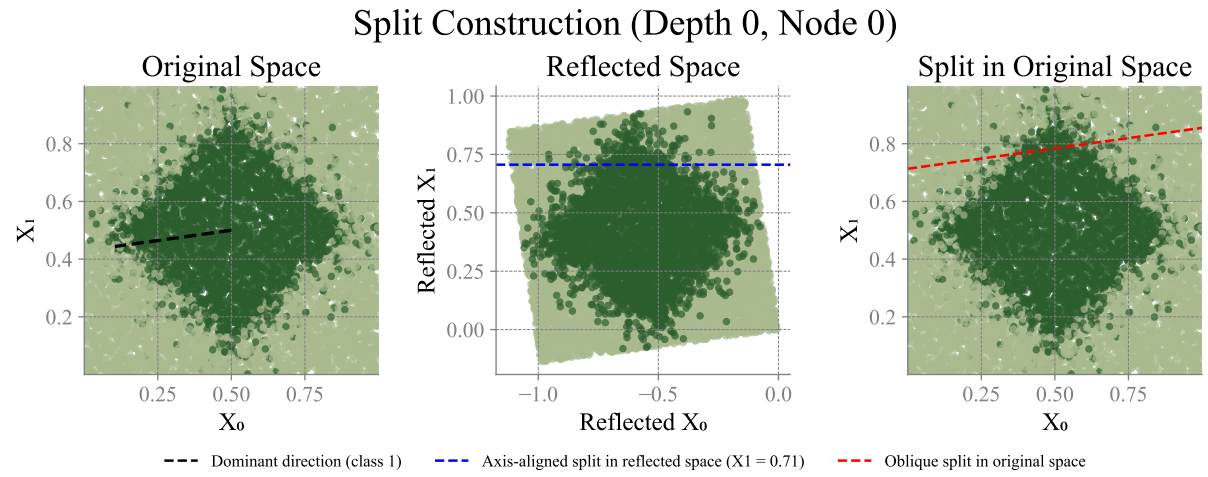
Figure E.1a illustrates this process at the root node. The left panel shows the original feature space, with the dominant direction of the selected class indicated by a black dashed arrow. This vector corresponds to the reflection that produced the lowest impurity. The centre panel shows the data after rotation, with the dominant direction now aligned with one of the basis axes; the axis-aligned split chosen in this space is shown as a blue dashed line. The right panel shows the final decision boundary in the original space, now appearing oblique (in red).

Although the split shown in Figure E.1a is optimal within HHCART(D)'s internal evaluation, it does not align with the orientation of the rotated rectangle and appears unintuitive when visualised. Instead of following one of the rectangle's edges, the boundary cuts off one of its corners. This outcome is caused by the algorithm's heuristic, which limits its search to a small set of rotated coordinate frames. At each node, HHCART(D) evaluates one reflection per class, based on that class's dominant variance direction, along with the identity transformation. In total, only three orientations are considered in the binary case. Within each frame, the algorithm performs a simple axis-aligned threshold search and selects the option that yields the lowest Gini impurity. If the true decision boundaries does not align with the dominant directions of class variance or with the original coordinate axes, HHCART(D) will struggle to identify it.

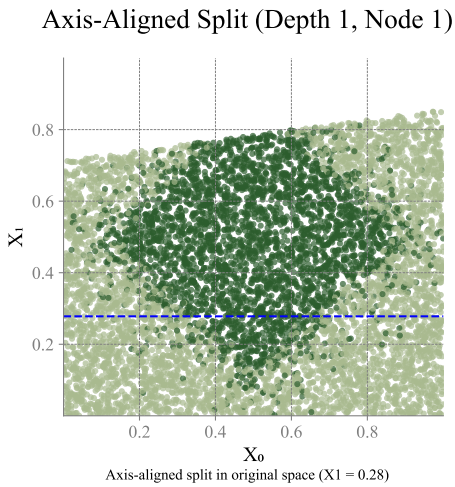
Figures E.1b and E.1c show the two splits that follow the root node in the tree. Both are axis-aligned and selected using the identity reflection. The first split further partitions one of the regions created by the root, and the second splits a subregion of that again. Neither aligns with the borders of the rotated rectangle, and both appear visually suboptimal. However, they were chosen because they achieved the lowest impurity among the tested options at their respective nodes.

Together, these visualisations demonstrate that the splits produced by HHCART(D) are shaped more by local variance structure and heuristic constraints than by global geometric alignment. While the method is capable of producing oblique boundaries, its reliance on a few (number of classes + 1) rotated frames and axis-aligned thresholding means that it may miss more intuitive partitions.

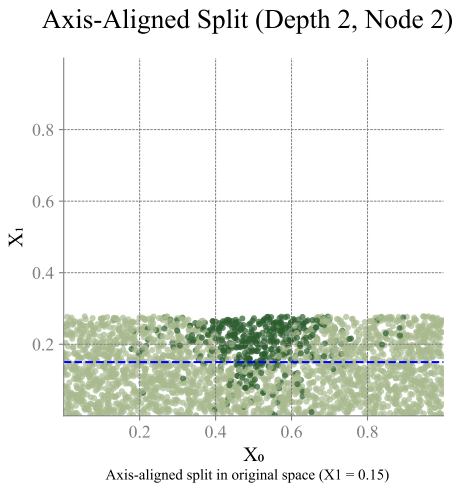
Figure E.1
Split construction visualisations for HHCART(D) applied to the 2D rotated rectangle benchmark with label noise ($\lambda = 0.05$). Each panel corresponds to a split made at a different node during recursive tree construction, visualising how the algorithm builds decision boundaries using Householder reflections and axis-aligned thresholds. Panel (a) shows the initial split at the root node (depth $= 0$), where an oblique decision boundary is constructed based on the dominant direction of class-specific variance. This split partitions the data into two regions. Panel (b) shows the split at depth 1, node 1, applied within one of the subspaces created by the root node. Here, no reflection is applied, and the split is purely axis-aligned, dividing the region horizontally. Panel (c) shows the split at depth 2, node 2, which operates within one of the subregions introduced by the previous split in Panel (b). This final split is also horizontal and further partitions that subspace, resulting in a finer rectangular tiling. All three panels reflect the recursive, locally adaptive nature of the HHCART(D) procedure.



(a)



(b)



(c)

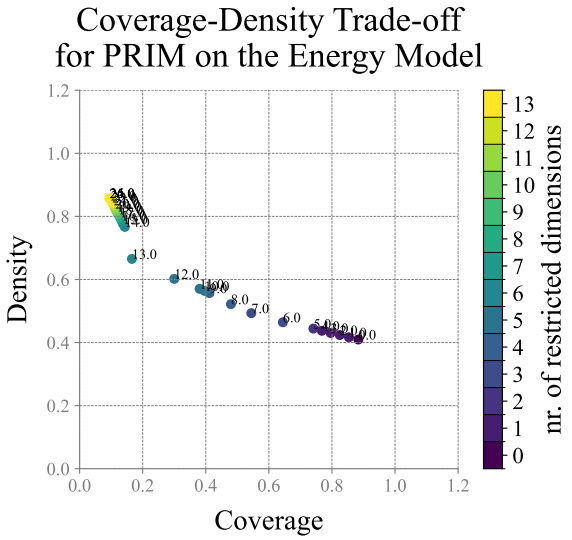
F Outputs for the Policy Model

This appendix provides supplementary materials that support the findings discussed in Chapter 6. It is structured by method, presenting additional outputs and results for each of the scenario discovery techniques applied to the EU energy model. The sections detail the full results for the established methods - PRIM, PCA-PRIM, and CART - and provide the complete depth-wise performance trajectories for the HHCART(D) models, concluding with a consolidated table that compares the final scenarios from all methods.

PRIM

This section presents the second scenario box identified by the PRIM analysis. As shown in Figure F.1, this box performed poorly on both coverage and density. The density of the full dataset is 46%, and this box does not exceed that baseline meaningfully. Furthermore, the second box overlapped substantially with the first box already reported in Chapter 6, and did not contribute additional interpretable scenarios. Based on these considerations, it was excluded from the main comparative synthesis.

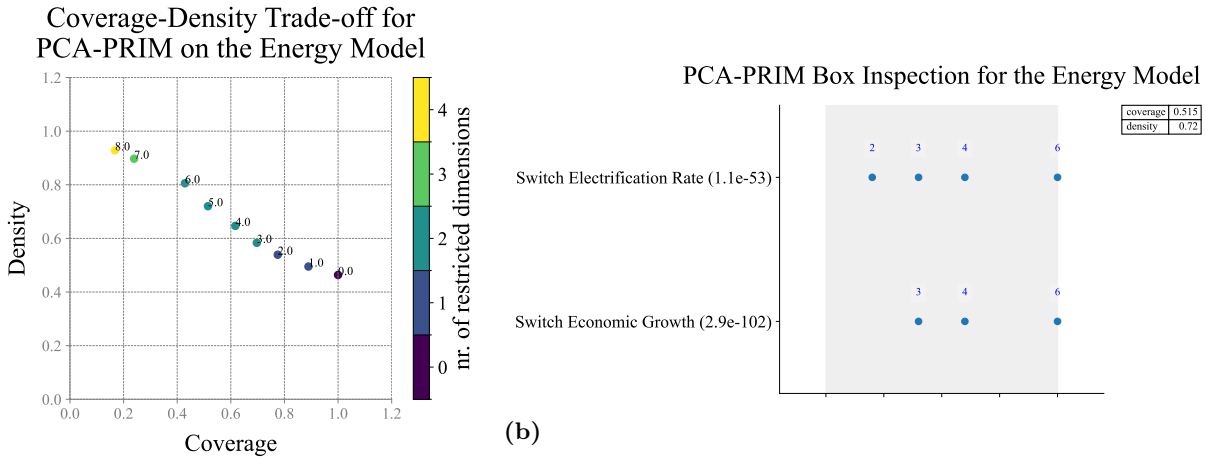
Figure F.1
Coverage-density trade-off curve for PRIM analysis applied to find a second box for the EU energy model output. Colours indicate the number of restricted dimensions.



PCA-PRIM

The PCA-PRIM analysis yielded results that were nearly identical to those of the standard PRIM analysis. At low coverage values, some minor differences were observed, but for the boxes of interest, the coverage-density trade-off curve matched that of the standard PRIM exactly. As shown in Figure F.2, Box 5 in the PCA-PRIM analysis exhibited the same box restrictions as in the original PRIM run (see Figure 6.2b in the main text). The principal components were effectively ignored, as the algorithm prioritised the highly informative categorical variables. Therefore, PCA-PRIM did not offer additional insight and was excluded from the final comparative synthesis in Chapter 6.

Figure F.2
Results of the PCA-PRIM analysis applied to the EU energy model output. Panel (a) shows the coverage-density trade-off curve. curve, with colours indicating the number of restricted dimensions. Panel (b) shows the scenario rules for the selected Box 5.



(a) **(b)**

CART

Figure F.3 shows the original unpruned CART tree trained on the EU energy model. In several cases, branches produced multiple adjacent leaf nodes classified as 1, prompting the application of post-pruning to merge such leaves into their respective parent nodes. The resulting pruned tree is presented in Chapter 6 (Figure 6.3), and did not alter the combined coverage and density of the scenario boxes.

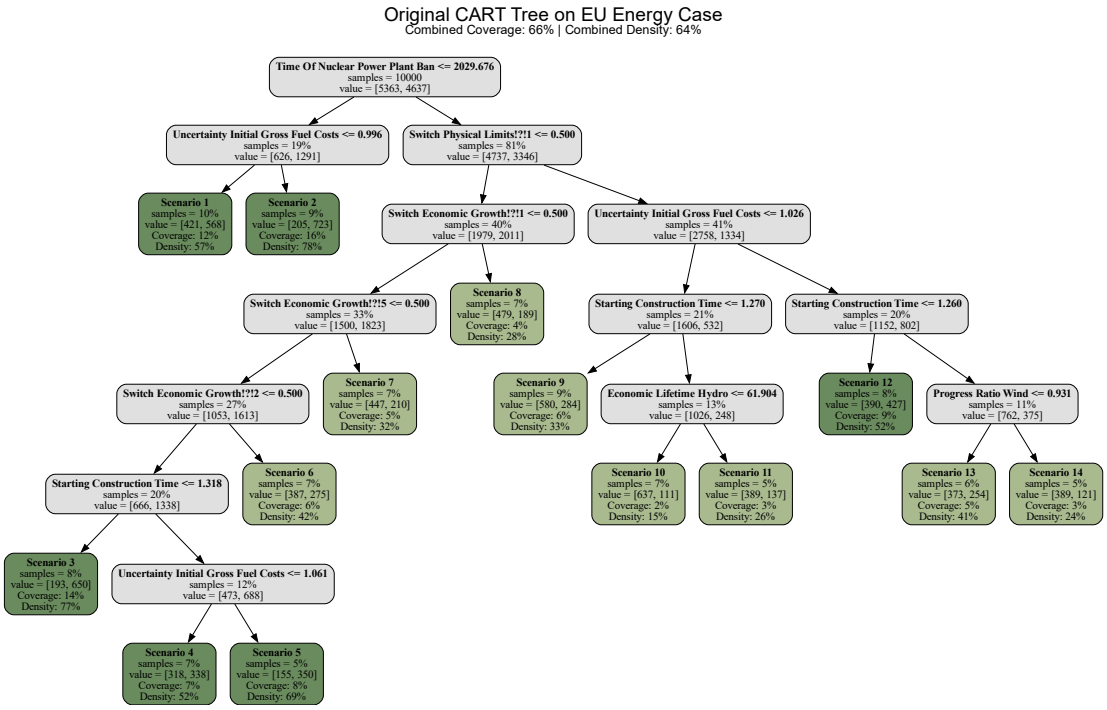
For interpretability, the final scenario boxes from the pruned tree were also extracted and summarised textually below in the Pruned CART Scenario Box. The box summary lists the specific conditions (thresholds and categories) defining each of the scenarios classified as high-renewable futures. These correspond to the scenarios discussed in Section 6.2.

HHCART(D)

This section provides the full trade-off trajectories of the HHCART(D) models evaluated in Phase 3. Tables F.1 to F.4 report coverage, density, and the number of positively classified subspaces for tree depths 0 to 8, for configurations using the top two to five features identified through global sensitivity analysis.

These results complement the comparative analysis in Section 6.4, where selected configurations at depth 4 were presented. The full trajectories shown here clarify the trade-offs across depths that informed the selection of these configurations. As discussed in the main text, no consistent performance improvements were observed beyond depth 4, and deeper models tended to increase fragmentation without meaningful gains in coverage or density.

Figure F.3
Visualisation of the final original CART tree identifying the scenarios for a high-renewable future on the EU energy model. Each leaf node represents a final scenario, showing its coverage and density.



Pruned CART Scenario Summary

```
# CART Scenarios Summary (class 1, Density > 0.5)

=== Combined Statistics (for shown scenarios) ===
Combined coverage: 0.659
Combined density:  0.645

=== Individual Scenarios (Density > 0.5) ===

Scenario 1 (Original Box ID 1):
  Coverage: 0.278
  Density:  0.673
  Bounds:
    Time Of Nuclear Power Plant Ban: [2013.002, 2029.676]

Scenario 2 (Original Box ID 8):
  Coverage: 0.289
  Density:  0.668
  Bounds:
    Switch Economic Growth: categories = [3, 4, 6]
    Switch Physical Limits: categories = [2]
    Time Of Nuclear Power Plant Ban: [2029.676, 2099.999]

Scenario 9 (Original Box ID 23):
  Coverage: 0.092
  Density:  0.523
  Bounds:
    Starting Construction Time: [0.100, 1.260]
    Switch Economic Growth: categories = [3, 4, 6]
    Switch Physical Limits: categories = [2]
    Time Of Nuclear Power Plant Ban: [2029.676, 2099.999]
    Uncertainty Initial Gross Fuel Costs: [1.026, 1.500]
```

Table F.1
HHCART(D) trade-off summary with 2 features (Time Of Nuclear Power Plant Ban, Starting Construction Time) on the EU energy model.

Depth	Coverage (%)	Density (%)	Class 1 Leaf Count
0	0.000	0.000	0
1	27.841	67.345	1
2	69.808	55.801	3
3	39.465	64.710	5
4	52.728	61.156	9
5	49.062	62.828	10
6	44.512	65.316	14
7	49.472	63.740	18
8	49.278	64.348	20

Table F.2
HHCART(D) trade-off summary with 3 features (Time Of Nuclear Power Plant Ban, Starting Construction Time, Uncertainty Initial Gross Fuel Costs) on the EU energy model.

Depth	Coverage (%)	Density (%)	Class 1 Leaf Count
0	0.000	0.000	0
1	27.841	67.345	1
2	69.808	55.801	3
3	48.178	64.904	4
4	57.947	62.300	9
5	58.076	62.280	12
6	59.521	62.317	14
7	57.774	63.229	16
8	57.774	63.229	17

Table F.3
HHCART(D) trade-off summary with 4 features (Time Of Nuclear Power Plant Ban, Starting Construction Time, Uncertainty Initial Gross Fuel Costs, Switch Physical Limits 2) on the EU energy model.

Depth	Coverage (%)	Density (%)	Class 1 Leaf Count
0	0.000	0.000	0
1	70.822	55.965	1
2	45.482	64.162	1
3	57.580	63.632	3
4	52.405	68.354	6
5	53.116	68.303	10
6	52.124	69.454	12
7	53.440	68.929	17
8	54.216	68.952	21

Table F.4
HHCART(D) trade-off summary with 5 features (Time Of Nuclear Power Plant Ban, Starting Construction Time, Uncertainty Initial Gross Fuel Costs, Switch Physical Limits 2, Switch Economic Growth 6) on the EU energy model.

Depth	Coverage (%)	Density (%)	Class 1 Leaf Count
0	0.000	0.000	0
1	45.611	62.389	1
2	33.944	70.773	1
3	58.378	63.920	4
4	61.893	64.596	8
5	55.510	67.933	11
6	55.704	67.938	14
7	55.855	67.961	15
8	55.855	67.961	15

Comparison of All Methods

For a consolidated overview, Table F.5 summarises the results across all evaluated methods. This table presents the best-performing scenarios from PRIM and CART alongside the selected configurations from the HHCART(D) analyses. It is designed to enable a direct comparison of coverage, density, and complexity across all methods, supporting the comparative synthesis presented in Chapter 6.

Table F.5
Summary of scenario discovery results across all evaluated methods. For HHCART(D), the best configurations selected from the full depth-wise performance trajectories (Appendix F) are reported. The table enables direct comparison of coverage, density, number of subspaces, and model complexity across PRIM, PCA-PRIM, CART, and HHCART(D).

Method	Regions	Coverage (%)	Density (%)	Features	Tree depth
PRIM (Box 4)	1	62	64	2	-
PRIM (Box 5)	1	51	72	2	-
CART	3	66	64	1 to 5	5
HHCART(D) (2 features)	9	53	61	2	4
HHCART(D) (3 features)	9	58	62	3	4
HHCART(D) (4 features)	6	52	68	4	4
HHCART(D) (5 features)	8	62	64	5	4