Thesis for The Degree of Doctor of Philosophy

Interpretable Machine Learning for Modeling, Evaluating, and Refining Clinical Decision-Making

Anton Matsson

Department of Computer Science and Engineering
Chalmers University of Technology
Gothenburg, Sweden 2025

Interpretable Machine Learning for Modeling, Evaluating, and Refining Clinical Decision-Making
Anton Matsson

ISBN 978-91-8103-251-2

Acknowledgements, dedications, and similar personal statements in this thesis reflect the author's own views.

Doktorsavhandlingar vid Chalmers tekniska högskola
Ny serie nr 5709
ISSN 0346-718X

Department of Computer Science and Engineering
Division of Data Science and AI
Chalmers University of Technology
SE-412 96 Gothenburg
Sweden
Telephone +46 31 772 10 00

Cover illustration by Sanna Lindblom (June 2025). The image depicts a patient meeting, where the doctor is supported by data in making clinical decisions.

Chalmers Digitaltryck
Gothenburg, Sweden 2025

To mom and dad.

Abstract

Machine learning offers great promise for developing new treatment policies from observational clinical data. However, a key challenge in this offline setting is reliably assessing the performance of new policies. Meaningful evaluation requires that the proposed policy is sufficiently similar to the data-collecting policy—constraining the search for viable policies. In clinical settings, the data-collecting policy is typically unknown, necessitating probabilistic modeling for many evaluation methods. As a result, modeling, evaluating, and refining clinical decision-making are closely interconnected. This thesis explores these tasks with a focus on interpretability, essential for clinical validation and trust.

First, we examine representations of a patient's medical history that support interpretable policy modeling. As history accumulates over time, creating compact summaries that capture relevant historical aspects becomes increasingly important. Our results show that simple aggregates of past data, combined with the most recent information, allow for accurate and interpretable policy modeling across decision-making tasks. We also propose methods that leverage structure in the data collection process—such as patterns in missing feature values—to further enhance interpretability.

Second, in the context of policy evaluation, we emphasize the need for assessments that go beyond estimating overall performance. Specifically, in which situations does the proposed policy differ from current practice? To address this question, we leverage case-based learning to identify a small set of prototypical cases in the observed data that reflect decision-making under current practice. We propose using these prototypes as a diagnostic tool to explain differences between policies, providing a compact and interpretable basis for validating new treatment strategies.

Third, motivated by the need for interpretable policies that are compatible with offline evaluation, we propose deriving new policies from an interpretable model of existing clinical behavior. By restricting the new policy to select from treatments most commonly observed in each patient state—as described by the model—we enable reliable evaluation. This standardization of frequent treatment patterns may reduce unwarranted practice variability and offers a promising alternative to current practice, as demonstrated in real-world examples from rheumatoid arthritis and sepsis care.

Keywords: interpretability, observational data, off-policy evaluation, policy modeling, reinforcement learning, sequential decision-making

Acknowledgment

This journey would not have been possible without the continuous support and guidance of my supervisor, Fredrik Johansson. Thank you, Fredrik, for always being encouraging, patient, and deeply involved in my research. I am also grateful to my co-supervisor, Morteza Haghir Chehreghani, and my examiner, Dag Wedelin, for their valuable feedback and support throughout the process. I would also like to thank my master's thesis supervisor, Adam Andersson, who encouraged me to apply for this position in the first place.

It has been a pleasure to be part of the steadily growing DSAI division during these years. I have had the chance to get to know many wonderful people—students, postdocs, faculty, and administrative staff. I am especially grateful to have been part of the Healthy AI Lab. A special thank you goes to the other founding members—Lena, Newton, and Adam—who started this journey with me back in 2020. You made these years truly special!

During my PhD, I have had the fortune to be involved in an ongoing collaboration with researchers in the United States. I would like to thank Heather Litman and her colleagues at CorEvitas/Thermo Fisher Scientific (former and current) for the opportunity to use the CorEvitas rheumatoid arthritis (RA) registry data. I am also grateful to rheumatologist Dan Solomon for the insightful discussions on the treatment of RA. The paper on treatment patterns in RA would not have been possible without his input.

Last but not least, I want to thank my family. Thank you, Gunnel and Erik—my parents—and Julia, my sister, for your invaluable love and support during this journey. And thank you, Sara, my wonderful partner, for always being my greatest supporter. I would also like to thank my good friends—Andreas, Alfred, Filip, and Adrian—who generously offered me a place to stay in Gothenburg after I moved to Stockholm. A final thank you goes to Sanna Lindblom, who created the amazing illustration that adorns the cover of this thesis.

This work was partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation. The computations and data handling were enabled by resources provided by the National Academic Infrastructure for Supercomputing in Sweden (NAISS), partially funded by the Swedish Research Council through grant agreement no. 2022-06725.

List of Publications

This thesis is based on the following manuscripts produced during the author's PhD studies.

[**Paper I**] Matsson, A., Stempfle, L., Rao, Y., Margolin, Z. R., Litman, H. J., & Johansson, F. D. (2024). How Should We Represent History in Interpretable Models of Clinical Policies? *Proceedings of the 4rd Machine Learning for Health Symposium*, *PMLR 259*, 714–734.

[**Paper II**] Stempfle, L., Matsson, A., Mwai, N., & Johansson, F. D. (2025). Prediction Models That Learn to Avoid Missing Values. To appear in *Proceedings of the 42nd International Conference on Machine Learning*, *PMLR 267*.

[**Paper III**] Matsson, A., & Johansson, F. D. (2022). Case-Based Off-Policy Evaluation Using Prototype Learning. *Proceedings of the 38th Conference on Uncertainty in Artificial Intelligence*, *PMLR 180*, 1339–1349.

[**Paper IV**] Matsson, A., Solomon, D. H., Crabtree, M. M., Harrison, R. W., Litman, H. J., & Johansson, F. D. (2024). Patterns in the Sequential Treatment of Patients With Rheumatoid Arthritis Starting a Biologic or Targeted Synthetic Disease-Modifying Antirheumatic Drug: 10-Year Experience From a US-Based Registry. *ACR Open Rheumatology*, *6*(1), 5–13.

[**Paper V**] Matsson, A., Rao, Y., Litman, H. J., & Johansson, F. D. (2025). Pragmatic Policy Development via Interpretable Behavior Cloning. To be submitted to *Machine Learning for Health 2025*.

The following manuscripts were also produced during the author's PhD studies, but they are not included in this thesis.

[**Paper VI**] Breitholtz, A., Matsson, A., & Johansson, F. D. (2024). Unsupervised Domain Adaptation by Learning Using Privileged Information. *Transactions on Machine Learning Research*.

[**Paper VII**] Litman, H. J., Matsson, A., Rao, Y., Wei, J., Margolin, Z. R., Harrold, L. R., & Johansson, F. D. (2025). Interpretable Machine Learning Models: Describing Treatment Patterns of Healthcare Providers Selecting Rheumatoid Arthritis Therapies to Evaluate Treatment Policies.

To be presented at the *41st Annual Meeting of the International Society for Pharmacoepidemiology.*

[**Paper VIII**] Matsson, A., & Johansson, F. D. (2022). Evaluating Policies for Sepsis Management: Decomposing Value Estimates Using Prototypes. *AAAI 2022 Workshop on Trustworthy AI for Healthcare.*

[**Paper IX**] Breitholtz, A., Matsson, A., & Johansson, F. D. (2023). Provable Domain Adaptation Using Privileged Information. *ICML 2023 Workshop on Spurious Correlations, Invariance and Stability.*

Contribution Summary

The author's contributions to the publications included in this thesis are detailed below.

[**Paper I**] Co-designed the study, performed most of the empirical work, performed most of the data analysis, and wrote most of the manuscript.

[**Paper II**] Co-designed the study, performed most of the empirical work, contributed to the data analysis, and co-wrote the manuscript.

[**Paper III**] Co-designed the study, performed the empirical work, contributed to the data analysis, and co-wrote the manuscript.

[**Paper IV**] Co-designed the study, performed the empirical work, contributed to the data analysis, and wrote most of the manuscript.

[**Paper V**] Co-designed the study, performed the empirical work, performed most of the data analysis, and wrote the manuscript.

Contents

Part I

Overview

Chapter 1

Introduction

Informed decision-making is central to effective patient care. Good clinical decisions rely on the integration of individual clinician expertise with the best available external evidence, while also considering the unique needs and preferences of each patient (Sackett et al., 1996). While clinical judgment remains essential, the growing emphasis on personalized medicine highlights the need for adaptive treatment strategies—or policies—to support consistent and high-quality care (Chakraborty & Moodie, 2013, Chapter 1.2). Personalizing medicine through evidence-based policies not only benefits individual patients but also holds promise for standardizing care and reducing overall healthcare costs (Chakraborty & Moodie, 2013, Chapter 1.1).

Clinical decision-making generates vast amounts of observational data, including laboratory test results, radiology images, and clinical notes. Much of this data is stored in electronic health records and clinical registries, providing a strong foundation for applying machine learning techniques to both develop and evaluate new policies aimed at improving clinical decision-making (Rajkomar et al., 2018; Shortreed et al., 2011). Compared to conducting randomized clinical trials—which are often prohibitively expensive and time-consuming—leveraging observational data offers a more accessible and cost-effective alternative for refining patient care. Eventually, it may help address a central question in medical practice: Which intervention is most appropriate for a given patient at a given time?

Using machine learning to develop new policies for clinical decision-making involves several interconnected steps. A key challenge is evaluating the performance of a proposed policy: Is it likely to outperform current practice? Due to the difficulties associated with running clinical trials, such evaluations must typically rely on historical data collected under existing clinical practices, often through a model of clinician behavior (Precup et al., 2000). For a new policy to be evaluable in this setting, it must be sufficiently similar to the observed behavior (Gottesman et al., 2018), which ultimately constrains how the policy can be derived. As a result, modeling, evaluating, and refining clinical decision-making become tightly coupled processes.

This thesis—based on four published papers and one preprint—explores vari-

ous aspects of each of these components. As will become clear, interpretability serves as a unifying theme throughout, motivated by the need for transparency in high-stakes clinical decision-making (Rudin, 2019).

In the following sections, we introduce each area of contribution. Two types of policies will recur throughout this discussion: the *behavior policy*, which is assumed to be followed by clinicians in the observed data, and the *target policy*, which represents a new policy being developed. Formally, we define a policy as a mapping from a *state* to a set of possible *actions*. In the clinical context, the state represents a summary of the patient and their medical history, while each action corresponds to a medical intervention—for example, a choice of treatment.

1.1 Modeling Clinical Decision-Making

While a model of the behavior policy is essential for many approaches to policy evaluation, it can also serve as a tool to understand, describe, and validate clinical practice—provided it is both accurate and interpretable (Deuschel et al., 2024; Hüyük et al., 2021; Pace et al., 2022). Accurate modeling requires careful consideration of how to represent the patient's state (Gottesman et al., 2018). In particular, when used for policy evaluation, the state should account for all confounding variables—factors that influence both treatment decisions and outcomes. Although this assumption cannot be verified statistically (Rosenbaum, 2010), interpretability can aid in reasoning about whether it holds. As the amount of historical information increases over time, selecting a representation of the patient's history that maintains interpretability becomes increasingly important.

In Paper I, we compare different approaches to representing patient history for interpretable behavior policy modeling. Specifically, we contrast representations learned through sequence representation learning with carefully crafted summary features. Based on a comprehensive experimental evaluation across four clinical decision-making tasks—including the management of rheumatoid arthritis and sepsis—we find that incorporating only a few aggregated and recent aspects of the patient's history into hand-crafted representations allows for learning interpretable models that perform comparably to black-box alternatives. Our analysis also highlights challenges specific to common use cases, including policy evaluation.

In Paper II and Paper V, we develop methods that leverage structure in the data-generating process to improve the interpretability of learned models. In Paper II, we introduce missingness-avoiding (MA) machine learning, a general framework for training models that avoid relying on features with missing values. In healthcare, missingness is often structured—for example, the result of one medical test may determine whether another test is performed. In such cases, tree-based MA algorithms can partition the data according to these missingness patterns, reducing the model's reliance on missing features and enhancing interpretability. As shown in our experiments, MA models generally maintain predictive performance comparable to their unregularized

counterparts.

In Paper V, we account for a common pattern in the treatment of primarily chronic diseases: patients often remain on the same treatment across decision points. Focusing on tree-based policy representations, we propose a simple meta-estimator that decouples the prediction of whether a treatment change is necessary from the prediction of which treatment to switch to. Compared to a standard decision tree—in which many rules may be redundantly repeated across subtrees, each tied to a specific treatment—this approach may better reflect the decision logic used by clinicians. In experiments with rheumatoid arthritis data, our method improves the accuracy of behavior policy modeling—especially when accounting for differences across treatment stages.

1.2 Evaluating Clinical Decision-Making

Evaluating a target policy involves answering the following question: What would the expected outcome be if physicians treated patients according to this policy? Ideally, this quantity—known as the value of the target policy—should be higher than that of the behavior policy. While the value of the behavior policy can be estimated simply as the average outcome in the recorded data, estimating the value of the target policy is a challenging causal problem known as off-policy evaluation (Precup et al., 2000). The difficulty arises because we do not observe what would have happened if the treatment recommended by the target policy differed from the treatment actually given in the data.

Most approaches to off-policy evaluation rely fully (Precup et al., 2000) or partially (Farajtabar et al., 2018; N. Jiang & Li, 2016; Thomas & Brunskill, 2016) on importance sampling. This technique reweights the outcomes of observed trajectories of state-action pairs based on the relative likelihood of those trajectories under the target and behavior policies. A model of the behavior policy is needed to compute the weights, as the true behavior policy is generally unknown.

The standard importance sampling estimator provides an unbiased estimate of the policy value, but it is known to suffer from high variance—especially when there are significant differences between the target and behavior policies across many state-action pairs (Gottesman et al., 2018). This issue is particularly pronounced when the target policy is deterministic, since only a small subset of the observed trajectories will align with the target policy and thus contribute to the weighted average. While multiple works have sought to reduce variance in off-policy evaluation estimates (Farajtabar et al., 2018; N. Jiang & Li, 2016; Thomas & Brunskill, 2016), a fundamental question remains: Can we trust the estimated value?

We address this question in Paper III. In addition to estimating the policy value, we argue that the following questions should be addressed as part of the evaluation: "In what types of situations do the target and behavior policies differ?" and "How do these differences affect the estimated value?" In this work, we answer these questions by estimating the behavior policy using prototypical learning—an interpretable machine learning technique primarily

used for classification (Li et al., 2018; Ming et al., 2019). We use the learned prototypical cases, which correspond to key patients in the state-action space, as a diagnostic tool for off-policy evaluation. By comparing the target and behavior policies in each case, we obtain a compact summary of the differences between the policies, as demonstrated in the case of sepsis management. Moreover, we decompose the estimated value into prototype-based contributions, revealing in which situations the target policy yields higher outcomes than the behavior policy, and vice versa.

1.3 Refining Clinical Decision-Making

Reinforcement learning (RL) offers a promising framework for learning new policies to support clinical decision-making. RL is a subfield of machine learning in which an agent learns how to act within an environment by interacting with it and receiving feedback in the form of rewards or penalties (Sutton & Barto, 2018). The agent's goal is to learn an optimal policy—one that maximizes the expected cumulative reward (or equivalently, minimizes the expected cumulative penalty). While this interaction-based learning procedure is ill-suited for most clinical settings, certain RL algorithms can instead be applied to a fixed dataset of collected experiences—a setting known as offline, or batch, RL (Levine et al., 2020).

A major challenge in many approaches to offline RL is managing out-of-distribution actions, which may arise when the algorithm becomes overly optimistic about actions that are rarely observed in the training data (Fujimoto et al., 2019; Kumar et al., 2020). While this issue can be mitigated by constraining the target policy to stay close to the behavior policy during training (Fujimoto et al., 2019; Kostrikov et al., 2022; Kumar et al., 2020), evaluating deterministic policies off-policy remains difficult—particularly when using importance sampling-based methods (Gottesman et al., 2018; Voloshin et al., 2021). This challenge is further compounded by the fact that much of RL's recent success stems from its integration with deep learning (i.e., deep RL), where black-box neural networks are used to represent policies (Mnih et al., 2013). Although interpretable RL is an active research area (Ernst et al., 2005; Silva et al., 2020; Verma et al., 2019), the prevailing view is that deep RL is not yet ready for high-stakes domains such as healthcare (Glanois et al., 2024).

Because policy refinement relies on accurate offline evaluation, understanding the available data is a critical first step toward improving patient care. Fundamentally, what is not observed in the data cannot be evaluated. In Paper IV, we examine treatment patterns in rheumatoid arthritis. Focusing on therapy changes starting from the initiation of the first biologic or targeted synthetic disease-modifying anti-rheumatic drug (defined as baseline), we observe substantial variation in post-baseline treatment decisions across patients. While this practice variation enables the evaluation of a wide range of target policies, it also introduces statistical challenges due to sparse observations.

In Paper V, we propose a pragmatic approach to policy refinement aimed

at producing interpretable target policies that can be evaluated off-policy with sufficient statistical support. Specifically, we derive the target policy from the most frequently chosen treatments in each state, as estimated by a model of the behavior policy—optionally incorporating their observed outcomes. By using a tree-based model, we obtain an interpretable policy whose overlap with the behavior policy can be controlled by adjusting the number of top treatments considered. This approach can be viewed as a way to standardize common treatment patterns, possibly reducing unwarranted practice variation. Our experiments show that it provides promising alternatives to current practice, particularly in the management of rheumatoid arthritis. In contrast, policies derived from offline RL often yield value estimates with high variance, raising questions about their practical utility.

1.4 Thesis Outline

This thesis is an extended summary of Papers I–V, which are included in full in Part II. Chapter 2 introduces the interpretable machine learning methods used throughout the thesis. The following three chapters, Chapters 3–5, address the core components of using machine learning to improve clinical decision-making: policy modeling, policy evaluation, and policy refinement. The ordering of these chapters is motivated by the dependencies between the three components. Off-policy evaluation typically relies on accurate behavior policy modeling, including a sufficiently rich representation of the patient's state. In turn, the challenges associated with off-policy evaluation motivate our pragmatic approach to policy refinement. Finally, Chapter 6 summarizes the main contributions, discusses limitations, and outlines directions for future work.

1.5 Datasets and Experimental Setups

The experimental results presented in this thesis are primarily based on data from the PPD™ CorEvitas™ rheumatoid arthritis (RA) registry (Kremer, 2016) (hereafter referred to as the CorEvitas RA registry) and the Medical Information Mart for Intensive Care (MIMIC-III) database (Johnson et al., 2016). However, cohort selection and data preprocessing procedures vary slightly across the different papers. These details are generally omitted in the following chapters, but we refer the reader to each individual publication for a thorough description of the experimental setup used in each case.

Chapter 2

Interpretable Machine Learning

Interpretable machine learning is a recurring theme in this thesis. Compared to black-box models, such as unregularized deep neural networks, interpretable machine learning models are more understandable to humans (Rudin et al., 2022). When working with medical data, interpretability provides insights into a model's reasoning process, enabling troubleshooting and supporting human decision-making (Afnan et al., 2021). In this context, we focus on models that are directly interpretable—either fully, meaning their entire reasoning process can be understood by humans, or partially, meaning some parts of their internal logic are human-comprehensible—rather than on post hoc explanations of inherently uninterpretable models (Guidotti et al., 2018).

In general, an interpretable machine learning model can be obtained either by selecting an interpretable model class—such as decision trees or linear models—or by imposing constraints such as sparsity or decomposability. In this short chapter, we briefly outline different approaches to interpretable machine learning, focusing on two main types: models designed for tabular data and models that are capable of handling sequential data. The latter is an appropriate extension given the thesis's focus on sequential decision-making.

2.1 Classical Methods for Tabular Data

Two of the most classical interpretable models are decision trees and generalized linear models, including linear regression and logistic regression. These models assume a tabular representation of the input data, where each feature X_j of the input $X = [X_1, \ldots, X_d]^\top$ is a meaningful predictor of the outcome $Y \in \mathcal{Y}$.

Generalized Linear Models

Generalized linear models (GLMs) are foundational in machine learning. These models are parametric, meaning they rely on a set of parameters θ that are

learned from data. The GLM framework unifies various types of models with linear components, such as linear regression and logistic regression, enabling them to accommodate different relationships between the outcome variable y and the input variables $x = [1, x_1, \ldots, x_d]^\top$. The leading constant one simplifies notation in the following derivations. A GLM is expressed as

$$g\left(\mathbb{E}[Y \mid X = x]\right) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots + \theta_d x_d = \theta^\top x, \tag{2.1}$$

where g is the link function that connects the linear regression term $\theta^\top x$, or logit, to the conditional mean of the output.

Depending on the assumed output distribution $p(Y \mid X)$ and the choice of link function g, we obtain different models with varying properties. For example, assuming a normal distribution with the identity link $g(\mu) = \mu$ yields linear regression. In contrast, assuming a Bernoulli distribution with the logit link $g(\mu) = \log(\mu/(1-\mu))$ results in logistic regression. As the inverse of the logit link—the logistic function—maps to the interval $[0, 1]$, the output of logistic regression can be interpreted as a probability, allowing it to serve as a classifier for binary outcomes. Logistic regression can also be extended to multi-class classification problems, where the response variable takes values in $\{1, \ldots, K\}$ with $K > 2$. This can be done by computing logits $\theta_k^\top x$ for each class k and passing the resulting vector into the softmax function, a multi-class generalization of the logistic function. Essentially, logistic regression for binary and multi-class classification can be viewed as combining linear regression with the logistic and softmax function, respectively.

To learn the parameters θ of a GLM, we may use maximum likelihood estimation. Given a dataset $\{(x_i, y_i)\}_{i=1}^n$, the goal is to find the parameters $\hat{\theta}$ that maximize the likelihood of the data—or equivalently, minimize the negative log-likelihood:

$$\hat{\theta} = \arg\min_\theta \underbrace{-\frac{1}{n} \sum_{i=1}^{n} \log p(y_i \mid x_i; \theta)}_{J(\theta)}. \tag{2.2}$$

For linear regression, the optimization problem (2.2) has a closed-form solution. In more general cases, such as logistic regression, numerical optimization methods are typically required. In multi-class logistic regression, where $\theta = \{\theta_1, \ldots, \theta_K\}$, the cost function becomes $J(\theta) = -\frac{1}{n} \sum_{i=1}^n \log g_{y_i}(x_i; \theta)$, where $g_{y_i}(x_i; \theta)$ denotes the predicted probability for class y_i obtained from the softmax function.

As discussed in Rudin et al. (2022), we can interpret a GLM by inspecting the individual components of the model, $\theta_j x_j$, as functions of their corresponding input variables x_j. This allows us to understand how each feature contributes to the model's prediction. This kind of introspection is especially useful for continuous input variables; for binary or categorical variables, the effect reduces to a step function. In the case of multi-class classification—which is most relevant for this thesis—interpreting the model becomes more challenging due to the presence of K separate parameter sets, one for each class.

Decision Trees

A decision tree is a type of rule-based model that partitions the input space $\mathcal{X}$ into disjoint regions, each associated with a constant prediction value. Unlike GLMs, decision trees are non-parametric models, meaning they make no assumptions about the functional form of the underlying mapping $h : \mathcal{X} \to \mathcal{Y}$. This flexibility allows them to effectively capture nonlinear relationships.

Decision trees consist of internal nodes $v \in \mathcal{V}$ and leaf nodes $\ell \in \mathcal{L}$. Each internal node v applies a logical rule that directs an input x to one of its two children, based on the value of a specific feature x_{j_v} and a threshold τ_v. For instance, if $x_{j_v} < \tau_v$, the input follows the left branch; otherwise, it follows the right branch. When the input reaches a leaf node ℓ, the tree outputs a constant prediction value $\hat{y}_\ell$. In classification settings, this prediction is usually determined by a majority vote—that is, the most common class among the training samples that reach leaf ℓ is returned.

To learn the splitting rules that define a decision tree from training data, it is common to use a recursive approach, starting from the root node and building the tree from top to bottom. Let $\theta = (j_v, \tau_v)$ denote the parameters defining the splitting rule at node v. At each node, we solve the optimization problem

$$\hat{\theta} = \arg\min_{\theta} \underbrace{n_v^l(\theta) Q_v^l(\theta) + n_v^r(\theta) Q_v^r(\theta)}_{G(\theta)}, \tag{2.3}$$

where $n_v^l(\theta)$ and $n_v^r(\theta)$ denote the number of samples routed to the left and right children of node v, respectively, under the split defined by θ. The terms $Q_v^l(\theta)$ and $Q_v^r(\theta)$ represent the impurity (or cost) of the left and right child nodes. For classification tasks with K classes, a common choice of cost function is the Gini index, defined as

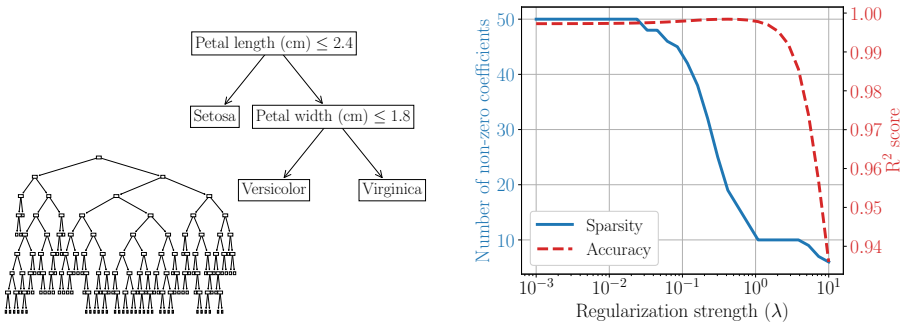
$$Q_v = \sum_{k=1}^{K} \hat{p}_{vk}(1 - \hat{p}_{vk}), \quad \text{with} \quad \hat{p}_{vk} = \frac{1}{n_v} \sum_{i : x_i \in \mathcal{S}_v} \mathbb{1}[y_i = k], \tag{2.4}$$

where $\mathcal{S}_v$ denotes the set of samples assigned to node v and $n_v = |\mathcal{S}_v|$.

Degrees of Interpretability

What makes a model interpretable? For generalized linear models and decision trees, sparsity is often a key factor (Rudin et al., 2022). A sparse GLM—where most parameters θ_j are zero—is generally easier to understand than one in which many parameters are nonzero, particularly when the number of potential input variables is large. In decision trees, sparsity is closely tied to the number of leaf nodes: the more leaf nodes a tree has, the harder it becomes to grasp the overall structure and trace individual decision paths.

For GLMs, sparsity can be encouraged through L^1-regularization, which involves adding a penalty term $\lambda \|\theta\|_1$ to the cost function $J(\theta)$, with λ controlling the strength of the regularization. In practice, L^1-regularization encourages many parameters to become exactly zero, which simplifies the model (Tibshirani, 1996). Risk scores are a specific type of sparse linear classification models,



(a) Decision trees.

(b) Sparsity vs. accuracy in a GLM.

Figure 2.1: Sparsity is often used as a proxy for interpretability in decision trees and GLMs. Panel (a): Two decision trees with differing sparsity levels. The upper tree, with only two inner nodes, accurately predicts flower species in the classic Iris dataset (Fisher, 1936) and is easily interpretable. In contrast, the lower tree, trained on a synthetic regression dataset generated with `make_regression` from scikit-learn (Pedregosa et al., 2011), has 100 leaves and is less comprehensible. For clarity, node labels are omitted in the lower tree, as they are not central to the point being illustrated. Panel (b): A simulation of the sparsity–accuracy trade-off in a regularized GLM, trained on synthetic data from `make_regression`. Increasing L^1-regularization reduces the number of features used by the model, thereby increasing sparsity at the potential cost of accuracy.

in which the coefficients are constrained to small integers (Ustun & Rudin, 2019), further enhancing interpretability.

For decision trees, sparsity is commonly improved through cost-complexity pruning, which begins with a fully grown tree and iteratively collapses leaf nodes that contribute little to predictive performance. More recently, methods have been proposed to directly learn decision trees that balance sparsity and predictive accuracy (J. Lin et al., 2020). Figure 2.1 illustrates two decision trees with different sparsity levels as well as the sparsity–accuracy trade-off in a regularized GLM.

2.2 Flexible Methods for Sequential Data

When the input data has a sequential structure of varying length, such that $X = X^1, \ldots, X^T$, with each $X^t = [X_1^t, \ldots, X_d^t]^\top$ and T a random variable, tabular methods such as decision trees and generalized linear models become difficult to apply, as they typically require inputs of fixed shape. To clarify notation, $t = 1, \ldots, T$ denotes the sequence index (e.g., time), and $j = 1, \ldots, d$ denotes the variable index. In this section, we discuss two types of interpretable models that are designed to handle sequential data: prototypical neural networks and recurrent decision trees.

Prototypical Neural Networks

Prototypical neural networks represent an approach to case-based reasoning (Aamodt & Plaza, 1994). The core idea of prototype learning is to use the training data to identify a set of representative examples—prototypes—that capture key characteristics of, for example, each class in a classification task. At test time, a new instance is compared to each prototype, and the prediction is made based on the prototypes that most closely resemble it. This approach is interpretable because the prototypes are actual training instances. For example, in a clinical context, each prototype may correspond to a real patient, enabling a domain expert to justify the model's prediction based on the test instance's similarity to one or more prototype patients. Compared to nearest-neighbor methods such as k-nearest neighbors, the most similar prototypes may better represent a particular class than the set of nearest neighbors (Rudin et al., 2022). Furthermore, since predictions only require comparisons to a fixed set of prototypes, inference is typically faster than in standard nearest-neighbor approaches.

The prototypical architecture consists of an encoding layer, a prototype layer, and a linear output layer. The choice of encoder depends on the input data. Here, we focus on sequential data and use a sequence learning model—such as a recurrent neural network—as the encoder e. The prototype layer contains m latent prototypes $\tilde{Z} = [\tilde{z}_1, \ldots, \tilde{z}_m]^\top$ that reside in the encoding space $\mathcal{Z}$, which is induced by the encoder $e : \mathcal{X} \to \mathcal{Z}$. Let

$$S(\tilde{Z}, e(x)) = [s(\tilde{z}_1, e(x)), \ldots, s(\tilde{z}_m, e(x))]^\top \tag{2.5}$$

be the similarity vector that compares the encoded input $e(x)$, where $x = x^1, \ldots, x^T$, to each prototype $\tilde{z}_k$ using a fixed similarity function $s : \mathcal{Z} \times \mathcal{Z} \to \mathbb{R}^+$. A natural choice for s is the radial basis function kernel with unit bandwidth:

$$s(\tilde{z}, e(x)) = \exp(-||\tilde{z} - e(x)||_2^2), \tag{2.6}$$

which returns a similarity score between 0 and 1, with 1 indicating complete similarity and 0 indicating no similarity.

Finally, we apply logistic regression in the space defined by the similarity vector S. Let θ_e and θ_f denote the parameters of the encoder and the logistic regression model, respectively. The full set of parameters $\theta = (\theta_e, \tilde{Z}, \theta_f)$ is learned by minimizing the negative log-likelihood of the data, as defined in Equation (2.2), using stochastic gradient descent.

The latent prototypes $\tilde{Z}$ are not directly interpretable, as they are free parameters in the latent space $\mathcal{Z}$. To obtain interpretable prototypes $\tilde{X} = [\tilde{x}_1, \ldots, \tilde{x}_m]$ in the input space, Ming et al. (2019) propose projecting the latent prototypes onto the nearest encoded training samples at regular descent steps:

$$\tilde{x}_k \leftarrow \arg\max_{x \in \mathcal{D}} s(\tilde{z}_k, e(x)) \quad \text{and} \quad \tilde{z}_k \leftarrow e(\tilde{x}_k), \tag{2.7}$$

where $\mathcal{D}$ denotes the training data.

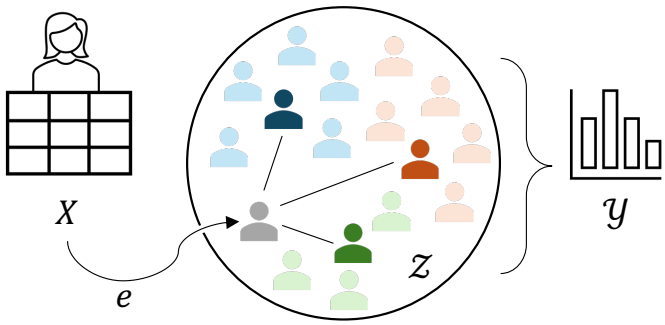


Figure 2.2: An illustration of a prototypical neural network architecture. The input data $X = X^1, \ldots, X^T$ has a two-dimensional structure, where each $X^t = [X_1^t, \ldots, X_d^t]^\top$ represents accumulated information about an individual (e.g., a patient) over time. The encoder e maps the input into an embedding space $\mathcal{Z}$, where three prototypes represent distinct clusters. For prediction, the similarity between the encoded input and each prototype is computed using a user-defined similarity metric (e.g., the radial basis function kernel). The resulting similarity vector is then used for output prediction. In this example, a classification task is assumed, and the output is a probability distribution over $\mathcal{Y} = \{1, \ldots, K\}$.

Recurrent Decision Trees

Recurrent decision trees were proposed by Pace et al. (2022) as an extension of soft decision trees (Frosst & Hinton, 2017), primarily for policy modeling applications (see Section 3.3). A soft decision tree is obtained by distilling a decision tree from a neural network. Each internal node v is represented by a gating function $p_v(x) = \sigma(x^\top \theta_v)$, where $x = [1, x_1, \ldots, x_d]^\top$ are (non-sequential) input variables, $\theta_v \in \mathbb{R}^{d+1}$ are learnable parameters, and σ denotes the logistic function. The gating function defines the probability that input x follows the right branch of the (sub)tree. Let $P_\ell(x)$ denote the probability that x reaches a given leaf node ℓ. In classification settings, each leaf defines a probability distribution over K classes using learnable parameters $\theta_\ell^y \in \mathbb{R}^K$, with $\hat{y}_\ell = \text{softmax}(\theta_\ell^y)$. To promote interpretability, the model outputs the probability distribution associated with the leaf node $\ell_{\max}$ having the highest path probability, that is, $\ell_{\max} = \arg\max_\ell P_\ell(x)$.

A soft decision tree can be optimized by minimizing the negative log-likelihood between the true label distribution and the predicted distribution at each leaf, weighted by the corresponding path probability. Interpretability can be enhanced by retaining, at each internal node, the input feature with the largest (non-bias) coefficient in θ_v (Silva et al., 2020), thereby enforcing unidimensional decision thresholds.

A recurrent decision tree extends the soft decision tree by accounting for the sequential nature of the input data $x = x^1, ..., x^T$. Each leaf of the tree updates a sequence embedding h_ℓ^t through an additional leaf parameter $\theta_\ell^h \in \mathbb{R}^m$, where m is the embedding dimensionality. The embeddings are

updated as $h_\ell^{t+1} = \tanh(\theta_\ell^h)$, and the embedding associated with the leaf $\ell_{\max}^t$ is appended to x^{t+1}, forming the full set of input variables at time step $t+1$. With $\theta_v \in \mathbb{R}^{d+m+1}$, the gating functions are now defined as

$$p_v\left(h^t, x^t\right) = \sigma\left(\theta_{v,0} + \sum_{i=1}^{m} \theta_{v,i} h_i^t + \sum_{i=1}^{d} \theta_{v,m+i} x_i^t\right). \tag{2.8}$$

When creating unidimensional thresholds at each inner node, the sum $\sum_i \theta_{v,i} h_i^t$ can be incorporated into the bias term $\theta_{v,0}$, effectively adjusting the threshold value.

Chapter 3

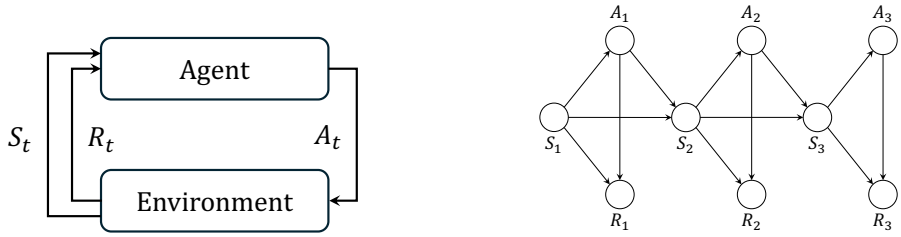
Modeling Clinical Decision-Making

In this chapter, we begin by formalizing the decision-making process, both in general and within the context of healthcare. We focus on sequential processes, as clinical decision-making is often inherently sequential: each choice influences future options and outcomes, and the overall result depends on the entire sequence of decisions. We then turn to modeling the policy that governs current decision-making behavior, as reflected in recorded data. A key aspect of policy modeling is the choice of representation for a patient's medical history. In Section 3.3, we explore different representations that support accurate and interpretable policy modeling, where interpretability enables, for example, explaining current practice and comparing alternative strategies (Pace et al., 2022). Finally, we examine two cases in which modeling can be improved by incorporating structural elements of the data-generating process—such as systematic patterns in treatment selection or missing feature values.

3.1 The Decision-Making Process

Sequential decision-making can be viewed as a sequence of interactions between an agent and an environment (see Figure 3.1(a)). At each stage $t = 1, \ldots, T$ of the process, the agent executes an action $A_t \in \mathcal{A}$ based on the current state $S_t \in \mathcal{S}$ of the environment. As a result, the environment transitions to a new state S_{t+1}, and the agent receives a scalar reward $R_t \in \mathcal{R}$, which reflects the quality of the action taken. The number of stages, T, is generally a finite random variable; however, to simplify notation, we assume T to be fixed in the following discussion. States, actions, and rewards are also random variables, and we denote their observed values using lowercase letters, s, a, r.

The interactions between the agent and the environment form a data-generating process, giving rise to a trajectory τ of states, actions, and rewards: $\tau = S_1, A_1, R_1, \ldots, S_T, A_T, R_T$. State transitions and rewards are determined by the dynamics of the environment, which can be expressed by the conditional



(a) Sequential decision-making.

(b) Markov decision process.

Figure 3.1: Sequential decision-making is commonly modeled as an interaction between a decision-maker, or agent, and an environment (panel (a)). At each time step t, the agent selects an action A_t based on the current state S_t of the environment. The environment then returns a reward R_t and transitions to a new state S_{t+1}. A standard assumption is that the decision process is Markov, meaning that the next state, action, and reward depend only on the current state-action pair. The probabilistic graphical model of a Markov decision process with horizon $T = 3$ is shown in panel (b).

distribution $p(S_{t+1}, R_t \mid S_1, A_1, R_1, \ldots, S_t, A_t)$. In a Markov decision process (MDP), the transitions and rewards depend solely on the most recent state-action pair, that is,

$$p(S_{t+1}, R_t \mid S_1, A_1, R_1, \ldots, S_t, A_t) = p(S_{t+1}, R_t \mid S_t, A_t). \tag{3.1}$$

As discussed in Sutton and Barto (2018, Chapter 3.1), the MDP formulation imposes a restriction on the state, not on the decision process itself: the state must capture all information relevant for predicting future state transitions and rewards. When this condition is met, the state is said to have the Markov property, or to be a Markov state. Figure 3.1(b) shows the probabilistic graphical model (Koller & Friedman, 2009) for an MDP with $T = 3$. An MDP is often represented as a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{R}, p)$.

The agent's decision-making strategy is defined by a policy $\psi \in \Pi$. A policy can be either deterministic or stochastic. On the one hand, a deterministic policy $\psi : \mathcal{S} \to \mathcal{A}$ maps each state $s \in \mathcal{S}$ to a specific action $a \in \mathcal{A}$. On the other hand, a stochastic policy $\psi : \mathcal{S} \to \Delta_{\mathcal{A}}$ maps each state $s \in \mathcal{S}$ to a probability distribution over the action space. For a deterministic policy, we use $\psi(s)$ to denote the selected action. For a stochastic policy, the probability of taking action a in state s is denoted by $p_\psi(A = a \mid S = s)$.

The policy ψ induces a distribution over trajectories, denoted by $p_\psi(\tau)$. For an MDP, such as the one shown in Figure 3.1(b), this distribution factorizes as

$$p(S_1)p_\psi(A_1 \mid S_1) \prod_{t=1}^{T-1} p(S_{t+1}, R_t \mid S_t, A_t)p_\psi(A_{t+1} \mid S_{t+1})p(R_T \mid S_T, A_T), \tag{3.2}$$

where components not influenced by the policy are written without the subscript ψ. Let $\mathbb{E}_\psi$ to denote the expectation with respect to this distribution. The

value of ψ, denoted V^ψ, is defined as the expected sum of rewards under $p_\psi(\tau)$:

$$V^\psi \coloneqq \mathbb{E}_\psi \left[\sum_{t=1}^{T} R_t \right]. \tag{3.3}$$

We may condition the value on the initial state $S_1 = s$, resulting in the state-value function: $V^\psi(s) \coloneqq \mathbb{E}_\psi \left[\sum_{t=1}^{T} R_t \mid S_1 = s \right]$. Similarly, by conditioning on both the initial state and the initial action, $S_1 = s$ and $A_1 = a$, we define the action-value function: $Q^\psi(s, a) \coloneqq \mathbb{E}_\psi \left[\sum_{t=1}^{T} R_t \mid S_1 = s, A_1 = a \right]$.

Reinforcement Learning The MDP formalism for sequential decision-making is commonly used in reinforcement learning (Sutton & Barto, 2018), where the goal is to learn an optimal policy for future decisions. We return to reinforcement learning in Chapter 5, where we discuss policy refinement.

3.2 Clinical Decision-Making

The decision-making process described in the previous section naturally translates to a clinical setting: the decision-making agent and the environment correspond to a physician and their patient, respectively; the state S_t represents the underlying condition of the patient; the action A_t is a medical intervention, such as a treatment administered to the patient; and the reward R_t corresponds to the observed outcome of this intervention. While the exact causes of a patient's response to treatment—that is, the dynamics of the environment—are unknown, we can formulate clinical decision-making in this way to develop policies for future decision-making, as further discussed in Chapters 4 and 5.

Although the high-level translation is conceptually straightforward, the challenges lie in the details. First, the patient's underlying condition is only partially observed, and it becomes our task to choose a state representation based on the available information. Second, the granularity with which actions are defined poses difficulties: should we model continuous medication dosages or discrete treatment options? Finally, formulating rewards requires balancing many important factors. In practice, we may care not only about clinical outcomes but also about patient well-being during treatment, resource utilization, and adherence to medical guidelines (Jayaraman et al., 2024).

Of the three challenges mentioned above, this thesis focuses on the choice of state representation. Regarding the formulation of actions, we assume a finite action space $\mathcal{A} = \{1, \ldots, K\}$, where K denotes the number of available actions, fixed across time. We return to the choice of reward functions in later chapters, but in general, we follow conventions from prior work—for example, assigning a positive signal for survival and a negative signal for death (Komorowski et al., 2018).

To formalize the discussion on how to represent the patient's state, we introduce the notion of patient history. To this end, let $X_t \in \mathcal{X}$ denote the

patient covariates or context available at time t, encompassing all currently accessible information about the patient—for example, demographics, diagnostic test results, and comorbidities. For convenience, we assume that the reward R_t is included in the covariates X_{t+1}. The patient's history up to stage t is defined as

$$H_t := X_1, A_1, X_2, A_2, \ldots, A_{t-1}, X_t. \tag{3.4}$$

The state S_t is defined as a function of the history, $S_t = f(H_t)$, ideally providing a compact summary of relevant historical events (Sutton & Barto, 2018, Chapter 17.3). As discussed in the previous section, a state is said to be Markov if it captures all information necessary to predict the future evolution of the trajectory (3.4) or its distribution. For the purpose of off-policy evaluation, discussed further in Chapter 4, the state should account for any confounding variables—that is, variables that causally affect both the current action and subsequent states or rewards (Namkoong et al., 2020).

In this thesis, we focus on two types of clinical decision-making processes: therapy selection in rheumatoid arthritis (RA) and the management of intravenous fluids and vasopressors in sepsis. These two examples are fundamentally different. While the treatment of RA is often a life-long process, with decision points occurring several months apart, the management of sepsis in the intensive care unit (ICU) typically spans only a few days, with continuous administration of treatment and real-time patient monitoring. Below, we briefly describe how states, actions, and rewards are defined in these two settings.

Therapy Selection in Rheumatoid Arthritis Rheumatoid arthritis is an autoimmune disease that affects the joints, often causing painful inflammation and stiffness. It is typically managed with disease-modifying anti-rheumatic drugs (DMARDs), which fall into three main categories: conventional synthetic DMARDs (csDMARDs), biological DMARDs (bDMARDs), and targeted synthetic DMARDs (tsDMARDs). Methotrexate (MTX), Tumor necrosis factor (TNF) inhibitors, and Janus kinase (JAK) inhibitors are the most commonly used csDMARDs, bDMARDs, and tsDMARDs, respectively. Biological DMARDs can be further divided into TNF inhibitor biologics and non-TNF inhibitor biologics. Both bDMARDs and tsDMARDs are often combined with a csDMARD to form combination therapies—in contrast to monotherapies, where only a single DMARD is used.

In this thesis, we focus on therapy selection beginning with the initiation of the first b/tsDMARD and onward. We define the action space based on these main categories of DMARDs, where an action represents the choice of a DMARD class rather than an individual drug. While the exact state variables will be discussed in more detail later, important factors include measures of disease activity, such as the clinical disease activity index (CDAI). In this context, the reward could, for example, be defined as the reduction in disease activity between two consecutive rheumatology visits.

Sepsis Management Sepsis is a severe, acute condition that occurs when the body's response to an infection causes damage to tissues and organs. It is a leading cause of death among hospitalized patients (Gotts & Matthay, 2016). In addition to administering antibiotics, the management of sepsis involves providing intravenous fluids and vasopressors to control the patient's blood pressure.

Sepsis management has been well-studied in the machine learning context (Komorowski et al., 2018; Luo et al., 2024; Raghu et al., 2017). In these studies, the continuous doses of intravenous fluids and vasopressors are discretized into five levels, which are then combined into a 25-dimensional action space. The most straightforward reward setup is to associate patient survival with a positive reward and patient death with a negative reward. Key state variables include vital signs and various laboratory measurements.

3.3 Interpretable Policy Modeling

The sequential process by which clinicians treat patients over time generates rich data that forms a basis for data-driven policy development. We define the behavior policy μ as representing the treatment patterns observed in this process—averaged over clinicians and patients. We quantify the behavior policy by estimating $p_\mu(A_t \mid S_t)$, the probability of selecting action A_t given state S_t. In the first part of this thesis, we use $\hat{p}_\mu$, or simply $\hat{\mu}$, to refer to a probabilistic model of the behavior policy.

Modeling the behavior policy has several important applications. First, it can be used to understand, describe, and validate current clinical practice (Hüyük et al., 2021; Pace et al., 2022). Second, by standardizing frequent treatment patterns identified by the behavior policy model, we may reduce unwarranted variation in clinical practice—an opportunity further explored in Chapter 5. Finally, behavior policy models are a key component in many approaches to off-policy evaluation (Raghu et al., 2018), which is the main focus of Chapter 4.

Each application benefits from an interpretable model of the behavior policy. Interpretability is essential when the goal is to explain current decision-making strategies; it is also desirable when the model is used to support future decision-making. In fact, interpretability is often considered a prerequisite for gaining the trust of end users in the medical domain (Stiglic et al., 2020). In off-policy evaluation of alternative treatment strategies, an interpretable behavior policy model can help summarize how the proposed strategies differ from current practice (see Paper III).

In Paper I, we ask: How should the patient history H_t, as defined in Equation (3.4), be represented to form a state S_t that supports interpretable and accurate behavior policy modeling? To address this question, we compare sparse methods designed for tabular data (Section 2.1) with interpretable methods based on sequence representation learning (Section 2.2). In sequence representation learning, the entire history H_t is fed into the model, which

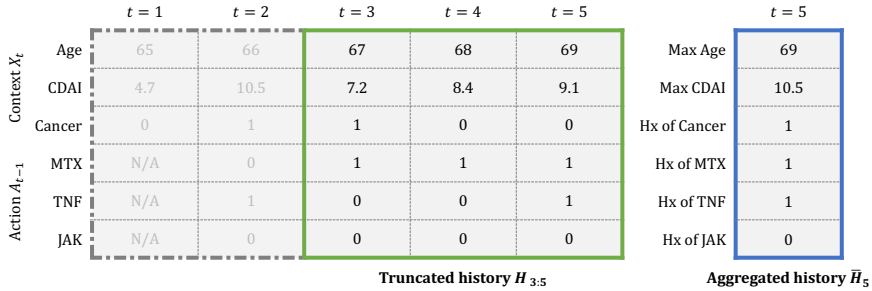


Figure 3.2: Examples of history truncation and history aggregation applied to a fictitious patient trajectory $H_5 = X_1, A_1, \ldots, X_5$ in the context of treatment selection for rheumatoid arthritis. Each context X_t consists of three components: the patient's age, CDAI, and an indicator for co-existing cancer. The simplified action space includes MTX, TNF inhibitors, and JAK inhibitors, with TNF and JAK potentially administered alongside MTX. In history truncation, only a fixed-length window of the most recent history is retained (here, a window of size three is used). In contrast, history aggregation summarizes the full history using an aggregation function, such as the `max` operator.

typically uses an encoder—for example, a recurrent neural network—to produce a representation for downstream prediction tasks (cf. the prototypical neural network described in Section 2.2). In contrast, tabular methods assume a fixed-size input format, requiring the construction of summary variables to represent historical events.

We focus on two summary approaches: history truncation and history aggregation. On the one hand, history truncation selects a fixed-length window of recent history, assuming distant events have limited impact on the action A_t. Formally, let $H_{(t-k):t} := (X_{t-k}, A_{t-k}, \ldots, X_{t-1}, A_{t-1}, X_t)$ denote the truncated history up to stage t, with $k \geq 0$. As illustrated in Figure 3.2, the truncated history with $k = 2$ includes the current context X_t, two preceding contexts, and actions from stages $t-1$, $t-2$, and $t-3$. On the other hand, history aggregation summarizes past information, assuming that the temporal order of events is insignificant for the decision A_t. Let X_t^i denote a component of the covariates X_t. Aggregating this information across time yields $\bar{X}_t^i = \text{agg}_t\, X_t^i$, where typical choices for the aggregation operator are `sum`, `max`, or `mean`. Similarly, binarized actions are aggregated as $\bar{A}_t^i = \text{agg}_t\, A_t^i$, resulting in the aggregated history $\bar{H}_t = \{\bar{X}_t, \bar{A}_{t-1}\}$. Figure 3.2 exemplifies history aggregation using the `max` operator.

Aimed at evaluating the fit of the behavior policy model, we compare models relying on sequence representation learning to models utilizing hand-crafted history representations across four distinct datasets, each representing a clinical decision-making task: (1) whether to order a magnetic resonance imaging (MRI) scan for patients with suspected cognitive impairment (ADNI), (2) treatment selection for patients with RA, and (3) management of sepsis and (4) acute exacerbation of chronic obstructive pulmonary disease (COPD) in

Table 3.1: Average test AUROC, expressed as a percentage, in four tasks: ADNI, RA, sepsis, and COPD. The upper section contains models that rely on hand-crafted states, while the lower section includes representation learning methods that use the entire history H_t as the model input. MLP and RNN are included as benchmarks. History aggregation $\bar{H}$ is performed using the `sum` operator.

	ADNI			**RA**			**Sepsis**			**COPD**		
State	LR	DT	MLP	LR	DT	MLP	LR	DT	MLP	LR	DT	MLP
X_t	56.2	53.9	55.6	61.7	58.8	61.1	82.1	78.2	84.1	77.9	74.7	78.8
A_{t-1}	53.9	53.8	53.7	94.7	94.7	94.7	88.0	90.6	91.1	92.9	95.0	95.0
$H_{(0)}$	56.8	54.3	56.8	95.6	95.7	96.1	91.3	92.1	94.7	94.0	96.0	95.4
$\bar{H}_t$	64.4	64.9	64.1	90.5	92.0	94.0	84.6	85.2	89.1	91.1	89.3	93.5
$H_{(0)}, \bar{H}_t$	65.3	65.0	65.8	96.1	96.5	96.9	91.9	92.3	95.3	94.7	96.7	96.3
$H_{(1)}, \bar{H}_t$	65.6	65.4	66.0	96.0	96.4	96.9	92.2	92.5	95.5	94.7	96.8	96.4
$H_{(2)}, \bar{H}_t$	65.4	65.3	66.8	96.0	96.4	96.7	92.3	92.6	95.5	94.7	96.8	96.3
	ADNI			**RA**			**Sepsis**			**COPD**		
State	PSN	RDT	RNN	PSN	RDT	RNN	PSN	RDT	RNN	PSN	RDT	RNN
H_t	66.7	62.8	68.0	96.2	90.0	96.8	94.9	77.0	95.7	96.2	81.9	96.5

the ICU. We consider the following history representations for logistic regression (LR), decision trees (DT), and multi-layer perceptrons (MLP): X_t, A_{t-1}, $\{X_t, A_{t-1}\} = H_{(t-0):t}$, $\bar{H}_t$, $\{H_{(t-0):t}, \bar{H}_t\}$, $\{H_{(t-1):t}, \bar{H}_t\}$, and $\{H_{(t-2):t}, \bar{H}_t\}$. To simplify notation, we define $H_{(k)} \coloneqq H_{(t-k):t}$. For comparison, we also include a prototypical neural network using a recurrent neural network as a sequence encoder (PSN), a recurrent decision tree (RDT), and a recurrent neural network (RNN). The black-box models, MLP and RNN, are included primarily for reference: How well can the behavior policy be fit using each input type without interpretability constraints?

In Table 3.1, we report the average test set area under the receiver operating characteristic curve (AUROC) across five different train-test splits for each dataset. We highlight the following key observations:

- Accounting for historical information—not just the current context—is crucial for achieving a good model fit. This is particularly evident for the RA and COPD datasets, where incorporating the entire history H_t yields more than a 20 % increase in AUROC compared to using only the current context X_t.

- Combining current observations (X_t), the most recent treatment (A_{t-1}), and summary features of the history ($\bar{H}_t$) captures most of the variance in treatment selection. Including additional history information yields only marginal improvements, as seen by comparing the performance of the $\{H_{(0)}, \bar{H}\}$ and $\{H_{(2)}, \bar{H}\}$ states.

- The best-performing interpretable model, the prototypical neural network, achieves performance comparable to the RNN, suggesting that interpretable policy modeling is generally feasible. Interpretable models using hand-crafted history representations become competitive when

selected aspects of the history are included, as discussed in the previous point.

An interesting observation is that models using the state $S_t = A_{t-1}$ perform surprisingly well in terms of average AUROC across tasks. Intuitively, the previous action alone should not provide a sufficient representation of the patient's history, raising the question: Is average AUROC alone an adequate metric for assessing model fit quality? To further explore this, we refine the evaluation for the sepsis dataset by performing both group-wise and temporal stratifications of the model fit.

In Figure 3.3(a), we stratify the results obtained with LR and PSN into patient groups based on the rate of change of the National Early Warning Score 2 (NEWS2), as defined by Luo et al. (2024). Patients in groups 1 and 6 exhibit large negative and large positive rates of change in the NEWS2 score, respectively, indicating that they are likely to experience higher variation in their treatment compared to patients in other groups. For these patients, the state A_{t-1} provides an insufficient representation of the patient's condition, as evidenced by a sharp decrease in AUROC.

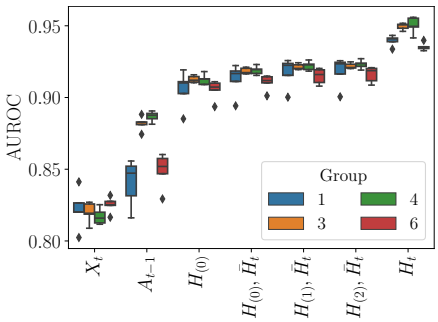
In Figure 3.3(b), we show the AUROC obtained with DT and PSN at different stages of the disease course. In the early stages, using the state $S_t = A_{t-1}$ leads to highly inaccurate predictions compared to using the full history as model input. In contrast, in later stages, the state A_{t-1} is, on average, almost as predictive as the entire history. One possible explanation for this pattern lies in the dynamics of sepsis management: at the onset of sepsis, patients require careful monitoring and frequent treatment adjustments from one decision point to the next, whereas once the patient's condition stabilizes, the default action often becomes maintaining the current treatment across decision points.

3.4 Structured Policy Modeling

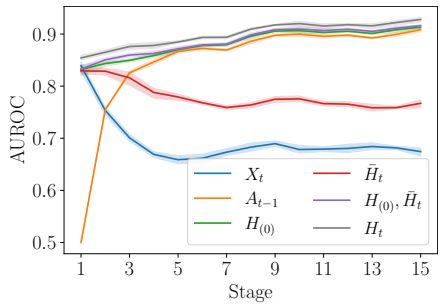
In the previous section, we showed that interpretable models—such as logistic regression and decision trees—can accurately model the behavior policy across diverse clinical decision-making tasks, provided a sufficiently rich state representation. However, as discussed in Section 2.1, interpretability exists on a spectrum, often influenced by a model's sparsity. For instance, while decision trees are generally considered interpretable, the interpretability of a specific tree depends on factors such as its depth and the number of leaf nodes.

In Figure 3.4, we show the topmost nodes of a standard decision tree trained on the RA dataset. The previous treatment variable, A_{t-1}, dominates the left side of the tree, reflecting a common pattern in chronic disease management: the tendency for patients to remain on the same treatment across decision points. Although not shown, this leads to an unbalanced tree with distinct subtrees for each treatment type, resulting in repeated rules across subtrees.

In Paper V, we leverage this structure in the data-generating process to create trees that are more accurate and more interpretable than standard decision trees. Specifically, we construct a meta-estimator (`SwitchTree`) that



(a) Stratification by patient groups.



(b) Stratification by time.

Figure 3.3: By stratifying the predictive performance of various state representations in the sepsis case by patient group (a) and treatment stage (b), we gain valuable insights into the limitations of each representation. In both cases, a prototypical neural network is used to fit the behavior policy using the entire history H_t as input. For the hand-crafted state representations, logistic regression is used in (a), and a decision tree is used in (b). The shortcomings of the A_{t-1} state representation—which appears surprisingly effective in the overall comparison—become evident in both stratified evaluations.

combines two separate decision tree classifiers: (1) a binary classifier that predicts whether a patient switches treatment and (2) a multi-class classifier that predicts the chosen treatment, trained on inputs (s_t, a_t) where a treatment switch occurs.

Let C_t be a binary random variable at stage t, where $C_t = 1$ denotes a change in treatment and $C_t = 0$ indicates continuation of the current treatment. Let $\hat{p}^s_\mu(S_t) \in [0, 1]$ and $\hat{p}^t_\mu(S_t) \in [0, 1]^K$ denote the outputs of classifiers (1) and (2), respectively: $\hat{p}^s_\mu(S_t) \coloneqq \hat{p}_\mu(C_t = 1 \mid S_t)$ and $\hat{p}^t_\mu(k \mid S_t) \coloneqq \hat{p}_\mu(A_t = k \mid S_t)$. To explicitly model the probability of selecting a treatment k given that a change occurs, denoted by $\tilde{p}^t_\mu(k \mid S_t)$, we exclude the probability of continuing the previous treatment a_{t-1} from $\hat{p}^t_\mu(S_t)$. This gives

$$\tilde{p}^t_\mu(k \mid S_t) \coloneqq \hat{p}_\mu(A_t = k \mid S_t, C_t = 1) = \frac{\mathbb{1}[k \neq a_{t-1}]\hat{p}^t_\mu(k \mid S_t)}{\sum_j \mathbb{1}[j \neq a_{t-1}]\hat{p}^t_\mu(j \mid S_t)}. \tag{3.5}$$

Finally, the meta-estimator combines the probabilities of staying on the same treatment and switching treatments:

$$\hat{p}_\mu(A_t = k \mid S_t) = \big(1 - \hat{p}^s_\mu(S_t)\big) \cdot \mathbb{1}[k = a_{t-1}] + \hat{p}^s_\mu(S_t) \cdot \tilde{p}^t_\mu(k \mid S_t). \tag{3.6}$$

In Table 3.2, we compare `SwitchTree` to a standard, single DT and an RNN for behavior policy modeling in RA and sepsis. We measure AUROC and static calibration error (SCE) (Nixon et al., 2019), since a good model of the behavior policy should be both accurate and produce well-calibrated probabilities. We also include another meta-estimator—`BLSwitchTree`—which uses a separate decision tree for treatment classification at the first time step,

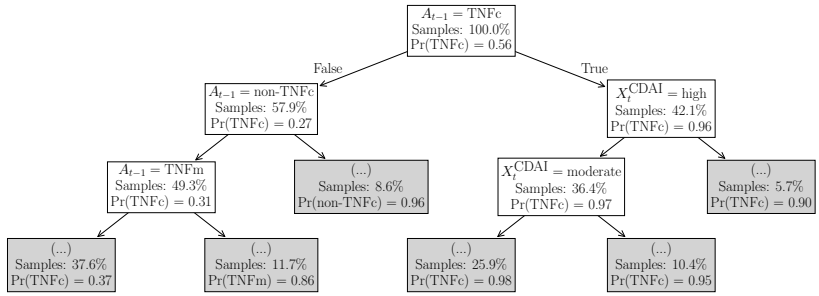


Figure 3.4: The topmost nodes of a decision tree fitted to estimate the behavior policy for RA treatment. The treatment selected at the previous time step, A_{t-1}, dominates the left branch of the tree, effectively creating distinct subtrees for each treatment type. As a result, many splits—such as those based on the CDAI—are repeated across subtrees, making the overall tree unnecessarily complex. The suffixes "m" and "c" indicate monotherapy and combination therapy, respectively.

Table 3.2: Average test AUROC and SCE for different models of the behavior policy across 50 splits of the RA and sepsis datasets.

	RA		**Sepsis**	
Estimator	AUROC (↑)	SCE (↓)	AUROC (↑)	SCE (↓)
DT	92.0 (91.8, 92.3)	2.7 (2.6, 2.8)	86.9 (86.5, 87.3)	0.4 (0.4, 0.4)
`SwitchTree`	92.8 (92.7, 93.0)	2.6 (2.5, 2.7)	86.0 (85.8, 86.2)	0.5 (0.5, 0.6)
`BLSwitchTree`	94.9 (94.7, 95.0)	1.3 (1.2, 1.3)	86.8 (86.6, 87.0)	0.5 (0.5, 0.5)
RNN	91.8 (91.7, 92.0)	2.4 (2.3, 2.5)	88.1 (88.0, 88.2)	0.5 (0.5, 0.5)

commonly referred to as the baseline (BL). This model accounts for the fact that treatment selection at baseline may differ from subsequent decisions. For example, as shown in Paper IV, TNF-based therapies dominate as baseline treatments in RA.

Table 3.2 highlights a key difference between decision-making in RA and sepsis. In the RA setting, explicitly modeling treatment switching—as done in `SwitchTree` and `BLSwitchTree`—improves predictive performance compared to using a standard decision tree or an RNN. The superior performance of `BLSwitchTree` over `SwitchTree`, both in terms of AUROC and SCE, suggests that handling baseline treatment selection separately from post-baseline decisions significantly enhances behavior policy modeling in this case. In contrast, for sepsis management, dividing the prediction task into two steps offers no clear benefit, and the RNN outperforms the tree-based models. This may be due to the continuous nature of sepsis care, where the decision-making process is less naturally framed as a two-step problem.

A benefit of this combined modeling approach is that practitioners can reason about two separate models, gaining insights into the mechanisms that lead to treatment changes. In Figures 3.5 and 3.6, we show the learned switch

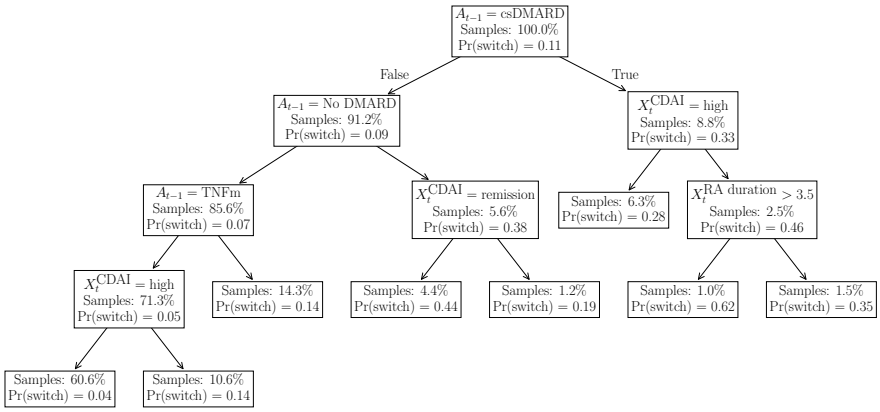


Figure 3.5: A decision tree trained to predict therapy switch events in RA treatment, as part of the `BLSwitchTree` model. The tree estimates the probability $p_\mu(C_t = 1 \mid S_t)$ that a patient in state S_t will switch treatment. The suffixes "m" and "c" indicate monotherapy and combination therapy, respectively.

and treatment trees when fitting `BLSwitchTree` to the RA dataset. The switch tree reveals that patients often change therapy after a period on csDMARDs or without DMARDs—a pattern also noted in Paper IV. This tendency is particularly pronounced when CDAI scores are high. Figure 3.6 shows that the previous treatment variable, A_{t-1}, provides a strong signal for determining post-baseline treatment changes. Specifically, patients tend to switch from TNF monotherapies to TNF combination therapies, from csDMARD therapies to TNF combination therapies, and from non-DMARD therapies to non-TNF monotherapies—transitions consistent with the results presented in Paper IV.

3.5 Handling Missing Data

Missing values are common in healthcare data. While tree-based models, such as decision trees, can be designed to handle missing values by learning default directions for each branch, most machine learning models require complete input variables. To address this, imputation techniques—such as univariate or multivariate feature imputation—can be applied. When missingness itself is informative, adding missingness indicators can improve predictive performance (Van Ness et al., 2023), although this increases the size of the feature set. Arguably, the use of imputation and missingness indicators complicates the relationship between inputs and outputs, thereby reducing the interpretability of the model and its predictions.

In clinical decision-making, decisions must be based on information that is currently available. Ideally, the models we build should reflect this fundamental principle. Sometimes, such as when clinicians gather information to establish a diagnosis for a patient, decisions may be deterministic—dependent on the

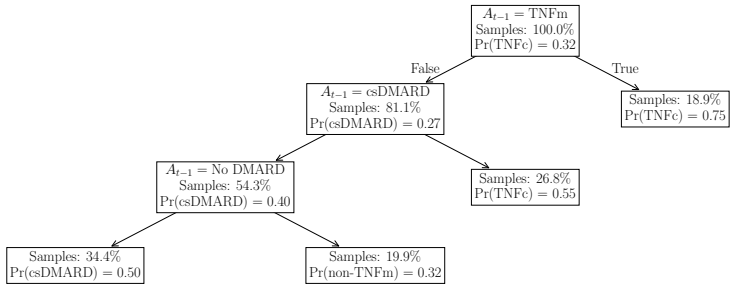
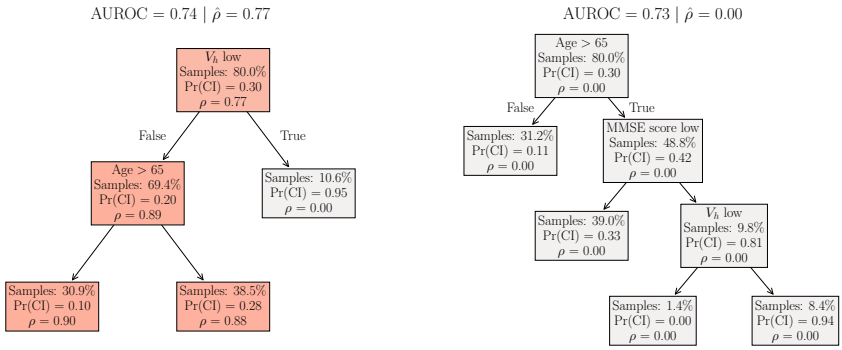


Figure 3.6: A decision tree trained to predict post-baseline treatment selection in RA treatment, as part of the `BLSwitchTree` model. The tree estimates the probability distribution over treatment options given a patient in state S_t, with each node label showing the probability of the most likely treatment. The suffixes "m" and "c" indicate monotherapy and combination therapy, respectively.

outcomes of previous decisions. As an example of a case where missingness patterns follow a clear structure, consider the following data-generating process.

Preventive Care Patients registered with a general healthcare provider undergo annual check-ups to assess their overall health. Demographic variables, such as age, are always recorded, whereas some test results may be missing due to clinical recommendations. For instance, cognitive tests are consistently administered to individuals over 65 years old, ensuring that mini-mental state examination (MMSE) scores are available for all patients in this age group. Patients who receive a low MMSE score subsequently undergo an MRI scan, which assesses hippocampal volume (V_h), categorized as either above or below average. MRI scans may also be ordered for unrelated clinical reasons—for example, to investigate spine or cartilage issues.

Assume we aim to predict whether a patient suffers from cognitive impairment using data collected by the healthcare provider. The dataset contains the following features: the patient's age, the outcome of any MMSE test (classified as normal or low), and the hippocampal volume from any performed MRI scan (also classified as normal or low). Figure 3.7(a) shows a standard decision tree fit to this data. The tree fails to reflect the underlying data-generating process, as it splits on the MRI scan outcome at the root node. Since this information is unavailable for most patients, the tree exhibits high *missingness reliance* ($\hat{\rho}$) (Stempfle & Johansson, 2024), as indicated by the coloring of individual nodes. In contrast, the decision tree shown in Figure 3.7(b) has learned to avoid relying on missing values. It more accurately represents the data-generating process and achieves zero missingness reliance, while maintaining predictive performance.



(a) Standard decision tree. (b) Missingness-avoiding decision tree.

Figure 3.7: Two types of decision trees trained to predict cognitive impairment from data collected at a general healthcare provider. The data-generating process follows a clear structure: the MMSE is administered to all patients over 65 years of age, and MRI scans—which measure hippocampal volume (V_h)—are primarily ordered for patients with low MMSE scores. The standard decision tree (a) fails to capture this structure, splitting on the MRI scan outcome at the root node. Although this variable is highly predictive, it is missing for 77 % of patients, resulting in high missingness reliance ($\hat{\rho}$). In contrast, the missingness-avoiding decision tree (b) accurately captures the data-generating process, resulting in zero missingness reliance. Importantly, both trees achieve similar predictive accuracy as measured by AUROC.

Missingness-Avoiding Machine Learning

In Paper II, we propose missingness-avoiding (MA) machine learning as a general framework for training models that minimize reliance on features with missing values. Let $h \in \mathcal{H}$ be a hypothesis that predicts a target variable $Y \in \mathcal{Y}$ from input variables $X = [X_1, \ldots, X_d]^\top \in (\mathcal{X} \cup \mathbf{na})^d$, where $\mathbf{na}$ denotes a missing value.[1] The missingness in X is determined by a missingness mask $M \in \{0,1\}^d$. Let $a_h(x,j) = 1$ indicate that the hypothesis h requires access to the variable x_j to compute $h(x)$; otherwise, let $a_h(x,j) = 0$. Following Stempfle and Johansson (2024), we define the missingness reliance $\rho(h,x) \in \{0,1\}$ of h for the input x as

$$\rho(h,x) = \max_{j \in [d]} \mathbb{1}[a_h(x,j) = 1 \wedge x_j = \mathbf{na}]. \tag{3.7}$$

The expected missingness reliance of h under the distribution $p(X,M,Y)$ is then defined as $\rho(h) \coloneqq \mathbb{E}_p[\rho(h,X)]$.

The goal of MA learning is to find a suitable trade-off between expected predictive performance and missingness reliance:

$$\min_{h \in \mathcal{H}} \mathbb{E}_p[L(Y, h(X))] + \alpha\rho(h), \tag{3.8}$$

[1]This can be translated to a decision-making context by setting $X = S_t$ and $Y = A_t$.

where L is a loss function and $\alpha \geq 0$ is a trade-off parameter. In Paper II, we provide algorithms for learning missingness-avoiding sparse linear models (`MA-Lasso`), decision trees (`MA-DT`), random forests (`MA-RF`), and gradient-boosted decision trees (`MA-GBT`). Here, we focus on the first two, as they align with the thesis's emphasis on interpretability.

To construct MA trees, we follow the greedy approach described in Section 2.1. Let j_v denote the index of the feature to split on at node v, and let τ_v denote the corresponding threshold. We define missingness reliance at the instance level as

$$\rho(h, x) \coloneqq \max_{v \in \pi_h(x)} \mathbb{1}[x_{j_v} = \mathbf{na}], \tag{3.9}$$

where π_h is the sequence of nodes traversed when making a prediction for input x using the hypothesis h. To construct trees that minimize both prediction error and missingness reliance, we modify the optimization problem from Equation (2.3) as follows:

$$\min_{j,\tau} G(j, \tau) + \alpha \sum_{x \in \mathcal{S}_v} \frac{1}{n_v} \mathbb{1}[x_j = \mathbf{na}], \tag{3.10}$$

where $\mathcal{S}_v$ denotes the set of training samples that reach node v during training and $n_v = |\mathcal{S}_v|$. For simplicity, we have omitted the subscript v in (j_v, τ_v).

For generalized linear models of the form $g(\mathbb{E}[Y \mid X = x]) = \theta^\top x$, where g is the link function (see Section 2.1) and $x = [1, x_1, \ldots, x_d]^\top$, we define missingness reliance at the instance level as

$$\rho(h, x) \coloneqq \max_j \mathbb{1}\left[|\theta_j| > 0\right] \mathbb{1}\left[x_j = \mathbf{na}\right], \tag{3.11}$$

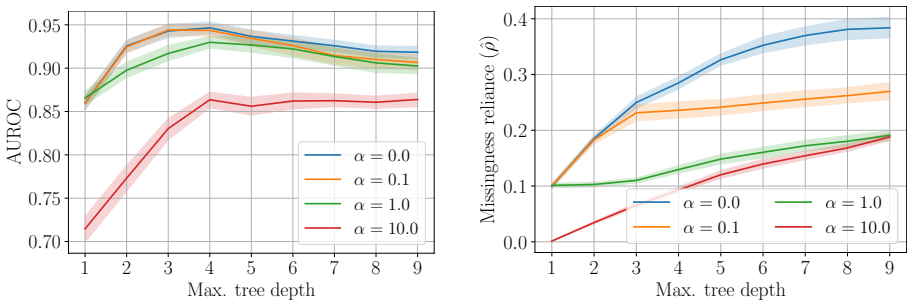
where $h = g^{-1}(\theta^\top x)$. As evident from this definition, generalized linear models cannot avoid missing values in a context-dependent manner as decision trees can. However, we can encourage such models to avoid relying on features with high missingness by modifying the optimization problem in Equation (2.2) as follows:

$$\min_\theta J(\theta) + \sum_{j=1}^{d} (\lambda + \alpha \overline{m}_j) |\theta_j|, \tag{3.12}$$

where λ and α control the strength of the regularization, and $\overline{m}_j = \frac{1}{n} \sum_i m_{i,j}$ is the empirical missingness rate of feature j, computed from the training data $\{(x_i, m_i, y_i)\}_{i=1}^n$. We solve Equation (3.12) by applying L^1 regularization (Lasso) with regularization strength λ' to a dataset with rescaled features $x'_j = \frac{\lambda'}{\lambda_j} x_j$, where $\lambda_j = \lambda + \alpha \overline{m}_j$.

In Paper II, we compare MA models to their unregularized counterparts, as well as to several models specifically designed to handle missing values (Le Morvan et al., 2020; McTavish et al., 2024; Stempfle & Johansson, 2024), across six tabular datasets with varying degrees of missingness. Our results show that, in most cases, MA models match the predictive performance of the baselines while significantly reducing reliance on missing values.

In Figure 3.8, we show how predictive performance (AUROC) and missingness reliance ($\hat{\rho}$) vary with the maximum allowed tree depth when building



(a) AUROC vs. tree depth.

(b) Missingness reliance vs. tree depth.

Figure 3.8: A comparison of predictive performance, measured by AUROC (a), and missingness reliance (b) across different maximum depths of a missingness-avoiding decision tree, using various values of the regularization parameter α. Setting $\alpha = 0$ corresponds to a standard decision tree. While increasing α from 0 to 1 has little effect on predictive performance (a), it substantially reduces missingness reliance (b).

MA trees with different strengths of the missingness regularization parameter α. When $\alpha = 0$, no regularization is applied, recovering the standard node splitting criterion. The task is to predict whether a country's life expectancy is below or above the dataset median, using World Health Organization data with 10 % missingness added to all predictors except the country's region (the country itself is not included as a predictor). As shown in Figure 3.8(a), a near-optimal tree (with depth 3–4) can be obtained for $\alpha \in \{0, 0.1, 1\}$. However, increasing α from 0 to 1 reduces missingness reliance from approximately 0.3 to 0.1—a two-thirds decrease.

Chapter 4

Evaluating Clinical Decision-Making

In the previous chapter, we formalized sequential decision-making and introduced the concept of a policy as a mapping from states to actions. We then focused on estimating a policy from observed data using interpretable machine learning methods, allowing for human verification of the learned policy model. In this chapter, we shift focus to a setting in which an alternative policy is given, and our task is to evaluate it. As we will see, the learned policy model continues to play an important role in this context.

4.1 Policy Evaluation

In Section 3.1, we defined the *value* of a policy ψ as the expected sum of rewards under the trajectory distribution $p_\psi(\tau)$ induced by actions selected according to that policy. For convenience, we restate the definition here:

$$V^\psi \coloneqq \mathbb{E}_\psi \left[\sum_{t=1}^{T} R_t \right]. \tag{4.1}$$

The value provides a natural basis for comparing policies: the higher the value, the better the policy. For example, in clinical settings, we often aim to compare the behavior policy μ with an alternative treatment strategy defined by a *target policy* π. In the policy evaluation setting, the target policy is assumed to be given.

In most practical settings, the expectation defined in Equation (4.1) is intractable, as the dynamics of the environment—and possibly the policy itself—are unknown. Fortunately, assuming access to a dataset of n sample trajectories from $p_\psi(\tau)$, $\mathcal{D}_\psi = \left\{ \left(s_1^{(i)}, a_1^{(i)}, r_1^{(i)}, \ldots, s_T^{(i)}, a_T^{(i)}, r_T^{(i)} \right) \right\}_{i=1}^{n}$, a simple

Monte Carlo estimator provides an unbiased estimate of the policy value:

$$\hat{V}^{\psi} = \frac{1}{n} \sum_{i=1}^{n} \sum_{t=1}^{T} r_t^{(i)}. \tag{4.2}$$

While it is straightforward to compute the value of the behavior policy μ using Equation (4.2) on a dataset $\mathcal{D}_\mu$ collected under that same policy, we typically do not have access to data collected under the target policy π. In clinical settings, executing the target policy without first estimating its value may be impermissible due to ethical concerns and safety regulations. Instead, we need to estimate V^π using the dataset $\mathcal{D}_\mu$—a task known as off-policy evaluation.

To ensure the identifiability of V^π, we must make assumptions about the behavior and target policies, μ and π, as well as the data-generating process underlying $\mathcal{D}_\mu$. Before stating these assumptions and formalizing off-policy evaluation, we introduce a key concept from causal inference: potential outcomes.

4.2 Potential Outcomes

The notion of potential outcomes captures what *would* happen under a specific action (Rubin, 2005). For example, consider a one-stage decision process ($T = 1$) in healthcare, where two treatments, a and a', are available for a patient represented by state s. The potential outcomes under a and a' are denoted by $R(a)$ and $R(a')$, respectively. If treatment a is chosen, we observe the reward r corresponding to the potential outcome $R(a)$. The counterfactual outcome $R(a')$, as well as the treatment effect $\Delta \coloneqq R(a) - R(a')$, remains unobserved. As a result, treatment effects must be studied at the population level—for example, through the conditional average treatment effect $\mathbb{E}[\Delta \mid S]$.

The potential outcomes framework can be extended to multi-stage decision processes (Chakraborty & Moodie, 2013, Chapter 2.1). In this setting, actions influence not only observed rewards but also observed states. Thus, it is useful to define the potential outcome of a general random variable Z. Let $\bar{A}_t = (A_1, \ldots, A_t)$ denote the sequence of actions up to stage t, and let $Z(\bar{A}_t)$ denote the potential outcome of Z resulting from $\bar{A}_t$. Specifically, $S_{t+1}(\bar{A}_t)$ and $R_t(\bar{A}_t)$ represent the potential outcomes of the state S_{t+1} and reward R_t, respectively, under $\bar{A}_t$. As t increases, the full set of potential outcomes—including states $S_1, \ldots, S_t(\bar{A}_{t-1})$ and rewards $R_1(A_1), \ldots, R_t(\bar{A}_t)$—grows rapidly. In this framework, the value of a policy ψ is defined as $\mathbb{E}_\psi \left[\sum_{t=1}^{T} R_t(\bar{A}_t)\right]$.

4.3 Off-Policy Evaluation

Off-policy evaluation poses the question: What would happen if we followed the actions recommended by the target policy π (Uehara et al., 2022)? Answering this requires counterfactual reasoning, since we only observe outcomes from actions taken under a different policy—the behavior policy μ—in a dataset

$\mathcal{D}_\mu \sim p_\mu(\tau)$. This thesis focuses on off-policy evaluation using importance sampling (IS) techniques (Precup et al., 2000). To ensure that the value function V^π is identifiable from the observed data, we assume sequential ignorability and overlap. The assumption of sequential ignorability requires that the behavior policy does not depend on confounding variables that also influence future states or rewards (Namkoong et al., 2020).

Assumption 1 (Sequential ignorability)**.** *For all stages $t = 1, \ldots, T$ and for any sequence of actions $\bar{a}_T$, conditional on the history H_t, the action A_t generated by the behavior policy μ is independent of future potential outcomes $R_t(\bar{a}_t), S_{t+1}(\bar{a}_t), \ldots, S_T(\bar{a}_{T-1}), R_T(\bar{a}_T)$. We say that the behavior policy μ satisfies sequential ignorability.*

Assumption 2 (Overlap)**.** *For all pairs of actions $A_t \in \mathcal{A}$ and histories $H_t \in \mathcal{H}$, $p_\mu(A_t \mid H_t) > 0$ whenever $p_\pi(A_t \mid H_t) > 0$. We say that overlap holds between the target policy and the behavior policy.*

In Assumptions 1 and 2, we condition on the entire history H_t. However, if the state S_t retains all relevant information from the history—that is, if it is a Markov state (see Section 3.2)—then we can replace H_t with S_t in these assumptions. We assume a Markov state in what follows.

In addition to sequential ignorability and overlap, we assume that any uncertainty in the target policy $p_\pi(A_t \mid S_t)$ arises from an exogenous variable. Under these assumptions, the value of the target policy π is defined as

$$V^\pi \coloneqq \mathbb{E}_\pi \left[\sum_{t=1}^{T} R_t \right] = \mathbb{E}_\mu \left[W \sum_{t=1}^{T} R_t \right], \tag{4.3}$$

where the weight W is obtained via importance sampling:

$$W \coloneqq \prod_{t=1}^{T} \frac{p_\pi(A_t \mid S_t)}{p_\mu(A_t \mid S_t)}. \tag{4.4}$$

In practice, the behavior policy $p_\mu(A_t \mid S_t)$ is often unknown. To compute the importance weights in Equation (4.4), we must therefore estimate the behavior policy from data $\mathcal{D}_\mu$ using a model $\hat{p}_\mu(A_t \mid S_t)$. This estimation problem was discussed in detail in Chapter 3. Given such a model, we can compute a sample-based estimate of Equation (4.3):

$$\hat{V}_{\mathrm{IS}}^\pi = \frac{1}{n} \sum_{i=1}^{n} w_i \sum_{t=1}^{T} r_t^{(i)} \quad \text{with} \quad w_i = \prod_{t=1}^{T} \frac{p_\pi\left(a_t^{(i)} \mid s_t^{(i)}\right)}{\hat{p}_\mu\left(a_t^{(i)} \mid s_t^{(i)}\right)}. \tag{4.5}$$

If the target policy is deterministic, we replace $p_\pi\left(a_t^{(i)} \mid s_t^{(i)}\right)$ with the indicator $\mathbb{1}\left[a_t^{(i)} = \pi\left(s_t^{(i)}\right)\right]$. As a result of Equation (4.3), the importance sampling estimator V_{IS}^π is unbiased provided that the importance weights W are correctly specified.

Other Approaches to Off-Policy Evaluation

While IS-based off-policy evaluation is widely used in practice, it suffers from high variance, as discussed in the following section. To reduce variance, we can improve the vanilla IS estimator in Equation (4.5) by leveraging the fact that the reward r_t does not depend on future state-action pairs $(s_{t'}, a_{t'})$ for $t' > t$. This insight leads to the per-decision importance sampling (PDIS) estimator (Precup et al., 2000):

$$\hat{V}^{\pi}_{\text{PDIS}} = \frac{1}{n}\sum_{i=1}^{n}\sum_{t=1}^{T} w_t^{(i)} r_t^{(i)} \quad \text{with} \quad w_t^{(i)} = \prod_{t'=1}^{t} \frac{p_\pi\left(a_{t'}^{(i)} \mid s_{t'}^{(i)}\right)}{\hat{p}_\mu\left(a_{t'}^{(i)} \mid s_{t'}^{(i)}\right)}. \tag{4.6}$$

To further reduce variance, we can form a weighted (per-decision) importance sampling estimator by normalizing the weights.

A different class of off-policy evaluation methods is known as direct methods, which use regression-based techniques to estimate the state-value function $V^\pi(s)$ or the action-value function $Q^\pi(s,a)$ (Voloshin et al., 2021). Model-based approaches model the dynamics of the Markov decision process (MDP), $p(S_{t+1}, R_t \mid S_t, A_t)$, and then use this model to estimate $V^\pi(s)$ or $Q^\pi(s,a)$. Let $\hat{V}^\pi(s)$ and $\hat{Q}^\pi(s,a)$ denote model-based approximations of these functions. These approximations can be obtained, for example, by solving the Bellman equations (Sutton & Barto, 2018, Chapter 3.5) using dynamic programming; see Chapter 5. The overall value of the target policy can then be estimated as

$$\hat{V}^{\pi}_{\text{DM}} = \sum_s p(S_1 = s)\hat{V}^\pi(s) = \sum_{s,a} p(S_1 = s) p_\pi(A_1 = a \mid S_1 = s)\hat{Q}^\pi(s,a), \tag{4.7}$$

where the subscript "DM" denotes that a direct method is used. Alternative, a model-free direct method, such as fitted Q-evaluation (Le et al., 2019), may be used to produce $\hat{V}^\pi(s)$ and $\hat{Q}^\pi(s,a)$.

Direct methods are biased and can be sensitive to the accuracy of the underlying models (Gottesman et al., 2018). To mitigate this issue, doubly robust (DR) estimators combine importance sampling with DM approximations (Farajtabar et al., 2018; N. Jiang & Li, 2016; Thomas & Brunskill, 2016):

$$\hat{V}^{\pi}_{\text{DR}} = \frac{1}{n}\sum_{i=1}^{n}\sum_{t=1}^{T} w_t^{(i)}\left(r_t^{(i)} - \hat{Q}^\pi\left(s_t^{(i)}, a_t^{(i)}\right)\right) + w_{t-1}^{(i)}\hat{V}^\pi\left(s_t^{(i)}\right), \tag{4.8}$$

where the importance sampling weights are computed in a per-decision manner, as defined in Equation (4.6). These hybrid approaches retain the unbiasedness of importance sampling while benefiting from the variance reduction offered by direct methods.

4.4 Challenges With IS-Based Off-Policy Evaluation

The importance sampling estimator (see Equation (4.5)) provides an unbiased estimate of the target policy's value. However, due to the product of importance weights $p_\pi(a_t \mid s_t)/\hat{p}_\mu(a_t \mid s_t)$, the variance of the estimator can become large, especially when the target policy π differs significantly from the behavior policy μ across many state-action pairs. While normalizing the weights or using per-decision importance sampling can reduce variance, it may still remain high in practice, particularly for long time horizons T. The problem is further exacerbated when π is deterministic, since only trajectories $\tau_i \sim p_\mu(\tau)$ that exactly match the actions of π contribute to the value estimate.

To assess the reliability of importance sampling estimates, we can compute the effective sample size n_e (Owen, 2013, Chapter 9), defined as

$$n_e = \frac{\left(\sum_i w_i\right)^2}{\sum_i w_i^2}. \tag{4.9}$$

The effective sample size equals n when the target and behavior policies are identical, and it approaches 1 in the worst case—meaning the estimate is effectively based on a single observation. Finding $n_e \ll n$ indicates that only a few weights dominate the weighted sum, suggesting that the importance sampling estimate may be unreliable.

To illustrate the challenges of IS-based off-policy evaluation, we consider treatment selection for patients with rheumatoid arthritis. Using a cohort of 1,565 patients, we repeatedly split the data into two halves: one for learning a target policy and the other for performing off-policy evaluation. Following the approach for modeling sepsis treatment described by Komorowski et al. (2018), we cluster all observations in the training data to obtain a discrete state space defined by the cluster centroids. Transition probabilities $p(S_{t+1} \mid S_t, A_t)$ are estimated by counting observed transitions between clusters, and the reward r_t is defined as the change in the clinical disease activity index relative to 10—the threshold between low and moderate disease activity (Aletaha & Smolen, 2005).

The (deterministic) target policy is obtained using Q-learning (see Chapter 5) via environment simulation, and off-policy evaluation is performed using weighted importance sampling. Across 50 iterations, an average of 779 patient trajectories are used for off-policy evaluation. However, the effective sample size is, on average, only 3, indicating that the IS estimate relies on less than one percent of the available trajectories. If we soften the policy so that every action has a 1 % probability of being chosen, the effective sample size increases to 5—still a very small number.

4.5 Case-Based Off-Policy Evaluation

A fundamental challenge with off-policy evaluation is that the ground truth value V^π is unknown. In addition, the sequential ignorability assumption (see Assumption 1) cannot be verified by statistical means (Rosenbaum, 2010), and

the extent of overlap (see Assumption 2) is unknown when the behavior policy μ is unknown—which is typically the case in clinical settings. Consequently, assessing the quality of the value estimate $\hat{V}^{\pi}$ ultimately requires human expertise.

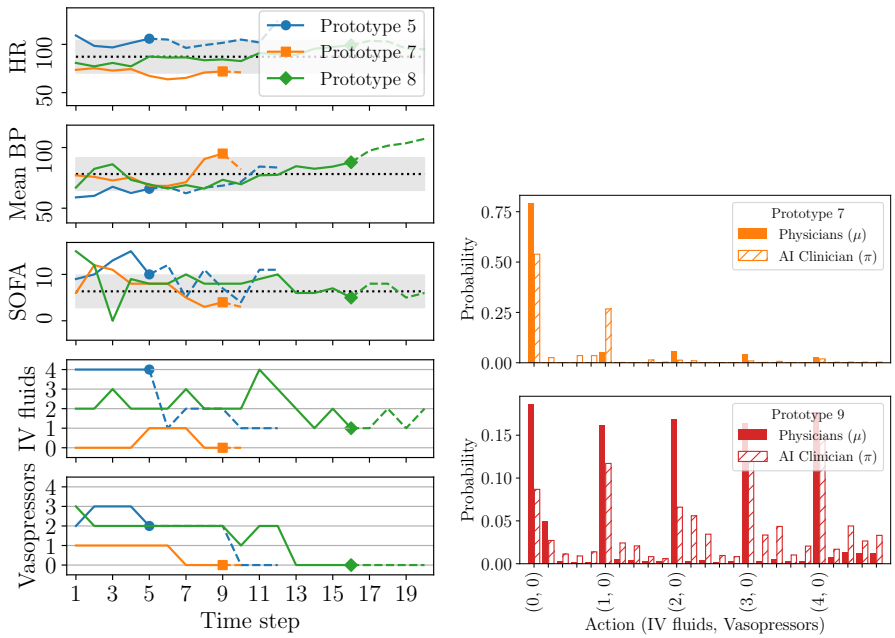
As discussed in the previous section, the analyst may compute the effective sample size to understand potential variance issues and diagnose the overall reliability of the value estimate. In addition, by inspecting individual weights w_i and estimated propensities $\hat{p}_{\mu}(a_t \mid s_t)$, the analyst can obtain a sample-wise view of evaluation, allowing for the removal of samples with excessive weights (Crump et al., 2009). However, both the average and sample-based perspectives fail to reveal patterns in states, actions, and rewards. In which situations do the target and behavior policies recommend substantially different actions? And when are the actions suggested by π preferable to those suggested by μ?

In Paper III, we develop a diagnostic tool for off-policy evaluation by estimating the unknown behavior policy $p_{\mu}(A_t \mid S_t)$ using prototypical learning (Li et al., 2018; Ming et al., 2019). As described in Section 2.2, a prototypical neural network is an interpretable deep learning architecture that combines an encoder with a prototype layer containing m prototypes—each corresponding to an input identified during training—followed by a linear output layer. The prototype layer compares a testing input to each prototype using a user-defined similarity metric, producing a similarity vector that is used for output regression. We use a recurrent neural network as the encoder, allowing the state S_t to be formed based on the entire patient history H_t.

The parameters of the model, including the prototypes, can be estimated using maximum likelihood estimation. By regularizing the learning objective to encourage a clustering structure in the encoding space—where each cluster is associated with a unique prototype—the learned prototypes induce a soft clustering of the state space. Since the prototypes are learned under action supervision, they intuitively describe common treatment patterns under the behavior policy. Moreover, because each prototype corresponds to a state in the training data $\mathcal{D}_{\mu}$, they can be readily interpreted by domain experts.

Assuming the number of prototypes, m, is relatively small, the learned prototypes provide a useful tool for understanding differences between the target and behavior policies. By comparing the actions recommended by the target and behavior policies in each of the prototypical states, the analyst can assess the degree of overlap between the policies and evaluate the validity of the actions suggested by the target policy. In Paper III, we demonstrate this idea using the example of sepsis management, comparing the Artificial Intelligence (AI) Clinician proposed by Komorowski et al. (2018) to the behavior policy followed by physicians.

In Figure 4.1(a), we visualize 3 of the 10 prototypes learned by the model, each corresponding to a particular state of a distinct patient in the training data. For reference, Figure 4.2(a) shows a principal component analysis (PCA) plot of the encoded training data, with each prototype numbered from 1 to 10. By plotting the trajectories of three key features (SOFA score, mean blood pressure, and heart rate), as well as the actions selected at each time



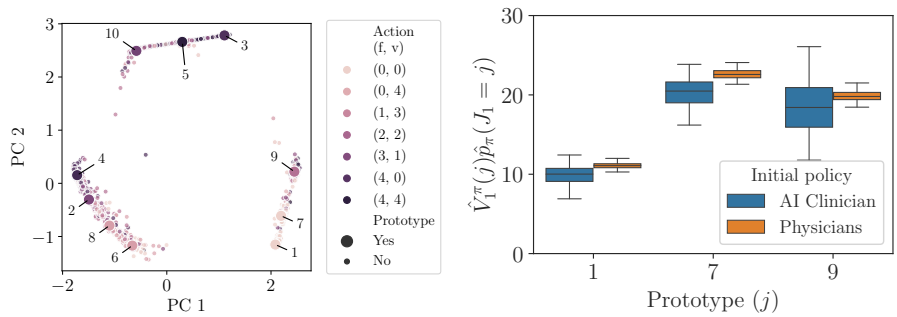
(a) Prototype examples. (b) Policy comparison.

Figure 4.1: By modeling the behavior policy $p_\mu(A_t \mid S_t)$ for sepsis management using prototypical learning, we obtain a set of learned prototypes, each corresponding to a state s_t in the training data $\mathcal{D}_\mu$. Each prototype can be interpreted by visualizing covariates and selected actions up to and including stage t (a). The time points corresponding to the state s_t of each prototype are indicated with filled markers. We use the prototypes as a diagnostic tool for off-policy evaluation by comparing the actions recommended by the behavior policy μ to those recommended by the target policy π for each prototypical state (b). For prototype 9, the AI Clinician (Komorowski et al., 2018) recommends a more aggressive use of vasopressors compared to physicians.

point, we gain insight into the types of patients the prototypes represent.[1] For example, the patient corresponding to prototype 5 has a high heart rate, low blood pressure, and a high SOFA score—signs of severe sepsis—and receives aggressive treatment. In contrast, the prototype 7 patient, who has a lower heart rate, higher blood pressure, and a lower SOFA score, receives low doses of intravenous (IV) fluids and vasopressors.

At the initial stage of treatment ($t = 1$), the encoded state s_1 of each patient in $\mathcal{D}_\mu$ is most similar to one of two prototypes: prototype 7, which clearly corresponds to a relatively healthy patient, and prototype 9, which corresponds to a patient for whom the model assigns equal probability to each dose of

[1]SOFA is an abbreviation for sequential organ failure assessment. The SOFA score is used to evaluate the severity of a patient's condition in the ICU by measuring the extent of organ dysfunction.



(a) PCA plot of encoded training data. (b) Value stratification by prototype.

Figure 4.2: A PCA plot of the encoded training data for modeling sepsis management, including the learned prototypes numbered 1–10, is shown in panel (a). The abbreviations "f" and "v" refer to fluids and vasopressors, respectively. The learned prototypes allow for stratifying the value estimates for the target and behavior policies (panel (b)), effectively breaking down the estimated values by types of situations.

intravenous fluids. As shown in Figure 4.1(b), by comparing the distribution over treatments according to the behavior policy model $\hat{p}_\mu(A \mid S)$ and the target policy $p_\pi(A \mid S)$ for each of these prototypical states, we gain insight into how the policies differ at the initial stage of treatment. For example, as shown in the lower panel, the target policy suggests a more aggressive use of vasopressors compared to the behavior policy, potentially violating the overlap assumption required for identifying V^π through off-policy evaluation. A domain expert may ask: Is this strategy medically sound?

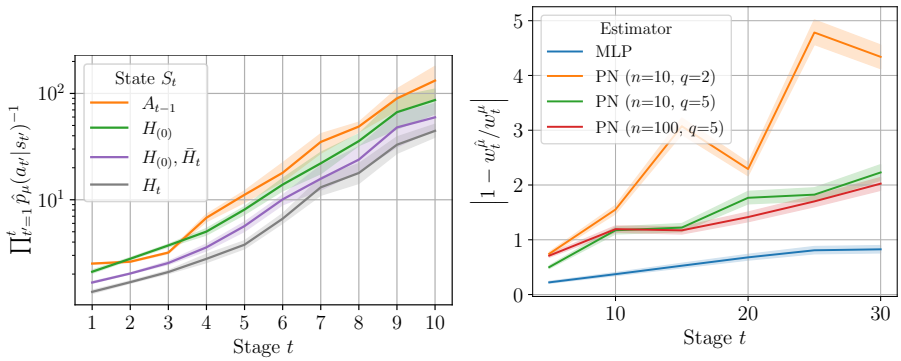
The learned prototypes also provide a natural way of stratifying the value estimate $\hat{V}^\pi$. Following the notation from Section 2.2, the latent prototypes—which reside in the latent space induced by the encoder e—are denoted by $\tilde{z}_1, \ldots, \tilde{z}_m$. Let $J_t \in \{1, \ldots, m\}$ be a random variable representing the assignment of a state s_t to prototype j at time t. The probability of s_t being assigned to prototype j at time t is defined as

$$p(J_t = j \mid S_t = s_t) \coloneqq \frac{s(\tilde{z}_j, e(s_t))}{\sum_{k=1}^{m} s(\tilde{z}_k, e(s_t))}, \tag{4.10}$$

where $s(\cdot, \cdot)$—in accordance with the notation in Section 2.2—is the user-defined similarity metric. The value $V_t^\pi(j)$ of prototype j at time t is defined as

$$V_t^\pi(j) \coloneqq \mathbb{E}_\pi \left[\sum_{t'=t}^{T} R_{t'} \mid J_t = j \right]. \tag{4.11}$$

With $p_\pi(J_t = j) = \mathbb{E}_\pi[p(J_t = j \mid S_t = s_t)]$ denoting the marginal probability of being assigned to prototype j at time t under the target policy π, we obtain,



(a) Inverse probability products. (b) Ratio of weights.

Figure 4.3: For a fixed target policy π and evaluation dataset $\mathcal{D}_\mu$, an IS-based estimate of V^π depends on the estimated behavior policy probabilities $\hat{p}_\mu(A_t \mid S_t)$. Panel (a) shows the median of the inverse probability products, $\prod_{t'=1}^{t} \hat{p}_\mu(a_{t'} \mid s_{t'})^{-1}$, across stages t, using different state representations in rheumatoid arthritis. A logistic regression model is used for all states except $S_t = H_t$, for which a prototypical network (PN) is used. Note the logarithmic scale on the y-axis: using a state based solely on the previous action results in rapidly increasing probability products compared to using the full history as state. Panel (b) illustrates how the ratio of IS weights, computed using the true behavior policy μ and its estimate $\hat{\mu}$, diverges from 1 as the number of stages increases, using a multi-layer perceptron (MLP) and various prototypical networks as behavior policy models in a synthetic experiment.

for any t,

$$V^\pi = \sum_{j=1}^{m} p_\pi(J_t = j) V_t^\pi(j). \tag{4.12}$$

Each term j in the sum in Equation (4.12), which can be estimated from observed data $\mathcal{D}_\mu$ using importance sampling, reflects the contribution of prototype j at time t to the overall value. This effectively stratifies the value by types of clinical situations.

Figure 4.2(b) shows the prototype-based value contributions for prototypes 1, 7, and 9—each belonging to the same cluster in Figure 4.2(a)—under both the target and behavior policies at the initial stage. Notably, for the target policy (AI Clinician), the variance of the estimate for prototype 9 is larger than that for prototype 7, reflecting the differences in overlap illustrated in Figure 4.1(b).

4.6 The Choice of History Representation

The prototype-based stratification of the value V^π (see Equation (4.12)) raises an important question: How does the number of prototypes, m, affect the

overall value estimate? If m is too small, the model class may become overly restrictive, making it difficult to learn an accurate behavior policy. Conversely, if m is too large, the learned model may become difficult to interpret, limiting its usefulness as a diagnostic tool for off-policy evaluation.

In the case of sepsis management, as shown in Paper III, we find that a relatively small number of prototypes (e.g., 10) is sufficient for accurately modeling the behavior policy. At test time, it is typically enough to retain only the 2–3 largest components of the similarity vector for output regression, meaning that predictions can be made using only the most similar prototypes (so-called prediction prototypes). However, since the true behavior policy is unknown, it remains difficult to precisely quantify the bias introduced into the importance weights used for off-policy evaluation.

To better understand this effect, we use the simple sepsis simulator provided by Oberst and Sontag (2019). We estimate the parameters of the underlying MDP from transitions sampled in each state-action pair and then learn an optimal behavior policy using policy iteration (Sutton & Barto, 2018, Chapter 4.3); see Chapter 5. Next, we collect trajectories $s_1, a_1, r_1, \ldots, s_t, a_t, r_t$ by executing the behavior policy in the environment for up to t stages. These samples are then used to estimate the behavior policy using prototypical networks with 10 and 100 prototypes, respectively. In Figure 4.3(b), we compare the ratio of per-decision importance weights $w_t^{\hat{\mu}}/w_t^{\mu}$ at each stage.[2] Compared to a multi-layer perceptron, the prototype-based models introduce a larger bias in the importance weights, and this bias increases with the trajectory length.

In Paper I, we conduct a similar experiment to assess the bias introduced in $\hat{V}^{\pi}$ by different history representations in rheumatoid arthritis therapy selection. In this setting, the true behavior policy is unknown and considerably more complex than in the simulator example. However, assuming a fixed target policy π and evaluation dataset $\mathcal{D}_{\mu}$, we can still assess the relative bias introduced by different history representations by comparing the median of the inverse probability products $\prod_{t'=1}^{t} \hat{p}_{\mu}(a_{t'} \mid s_{t'})^{-1}$ across varying numbers of stages t, as shown in Figure 4.3(a). We observe that using a state based only on the previous action, $S_t = A_{t-1}$, results in probability products that deviate significantly from those obtained when using the full history H_t as the state. This underscores the importance of evaluating policy models in the context of their intended use.

[2]The target policy is arbitrary in this comparison, as the target policy probabilities cancel in the ratio $w^{\hat{\mu}}/w^{\mu}$.

Chapter 5

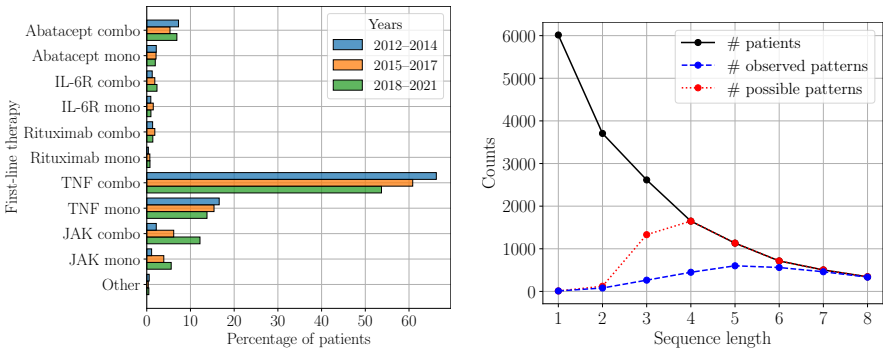
Refining Clinical Decision-Making

In the previous two chapters, we discussed various aspects of policy modeling and evaluation in a clinical context. Until now, we have assumed that the target policy is given. In this chapter, we shift focus to the process of proposing alternative treatment strategies based on observational health data. The goal is to improve clinical decision-making within the constraints imposed by the available data. Fundamentally, any target policy must remain sufficiently similar to the observed behavior policy to enable reliable off-policy evaluation.

Because of this constraint, improving clinical decisions requires a clear understanding of current practice patterns: What treatment sequences are commonly observed in the data? We begin by investigating this question in the context of rheumatoid arthritis (RA) management. We then introduce reinforcement learning (RL) as a method for optimizing treatment strategies using the Markov decision process (MDP) formulation introduced in Chapter 3. Next, we focus on the offline setting, where the learning agent cannot interact with the environment during learning. Finally, we present a pragmatic approach for generating target policy candidates based on a model of the behavior policy, allowing for control over the degree of policy overlap to ensure reliable off-policy evaluation.

5.1 Understanding Observational Health Data

In Paper IV, we study patterns in the sequential treatment of RA. The goal is to understand the marginal distribution over actions, $p(A_1, \ldots, A_T)$, without accounting for individual patient variation. There are several motivations for this analysis. First, summarizing common treatment sequences helps identify which alternative strategies can feasibly be evaluated in an observational setting. If observed treatment paths follow a narrow set of patterns, radically different strategies may not be evaluable without strong assumptions. Second, the marginal distribution $p(A_1, \ldots, A_T)$ provides an upper bound on the estimate



(a) First-line b/tsDMARD therapies. (b) Counts of patients and patterns.

Figure 5.1: In Paper IV, we analyze patterns in the sequential treatment of patients with rheumatoid arthritis using data from the CorEvitas RA registry (Kremer, 2016). A treatment pattern is defined as a unique sequence of k therapy changes, starting with and including the first-line b/tsDMARD therapy (baseline). Understanding these patterns is important for evaluating the potential of observational data to inform alternative treatment strategies. Panel (a) shows the distribution of b/tsDMARD therapies at baseline. While TNF inhibitors are the most common, JAK inhibitors have become increasingly prevalent in recent years. Panel (b) shows how the number of patients varies with sequence length k. As k increases, both the total number of patients and the number of patients per treatment pattern decrease.

of variation in clinical practice. Once patient-specific factors are taken into account, this variation decreases, and fewer distinct treatment patterns are observed within each subgroup. Third, unlike behavior policy modeling (as discussed in Chapter 3), analyzing treatment sequences requires no modeling assumptions and can be based purely on counts from the data.

The treatment of RA typically begins with a conventional synthetic disease-modifying anti-rheumatic drug (csDMARD) (Smolen et al., 2020). When initial therapy proves ineffective and poor prognostic factors—such as high disease activity—are present, clinical guidelines recommend adding a biologic DMARD (bDMARD) or a targeted synthetic DMARD (tsDMARD), thereby initiating the first line of b/tsDMARD treatment. Among these options, Tumor necrosis factor (TNF) inhibitors—a subgroup of bDMARDs—are the most frequently prescribed. Because clinical guidance is less clear on how to proceed when patients do not respond to the initial TNF therapy, many studies have examined the choice of second-line b/tsDMARD (Keystone et al., 2009; Salliot et al., 2011). Other research has investigated transitions between therapy lines (Fletcher et al., 2022; Zhao et al., 2022) or traced pathways to specific drugs (Solomon et al., 2021), but few studies provide a comprehensive overview of coherent treatment sequences.

Our analysis is based on data from the CorEvitas RA registry (Smolen et al., 2020), an ongoing longitudinal clinical registry in the United States,

covering the period from January 2012 to December 2021. We identify a cohort of 6,015 b/tsDMARD-naïve patients who initiated their first b/tsDMARD therapy during this period. The distribution of first-line b/tsDMARD therapies is shown in Figure 5.1(a). We focus on drug classes rather than individual medications, with combination therapies defined as a b/tsDMARD administered alongside one or more csDMARDs.[1] As expected, therapies involving TNF inhibitors are most common, although a shift from TNF inhibitors to Janus kinase (JAK) inhibitors (the main class of tsDMARDs) is observed over the study period.

We define a treatment pattern as a unique sequence of k therapy changes starting from and including the first-line b/tsDMARD therapy, which we consider the baseline. Due to censoring, the number of observed therapy changes varies across patients in the cohort. As shown in Figure 5.1(b), just over half of the patients experience one post-baseline therapy change; fewer than one-sixth of the patients undergo five or more therapy changes after baseline. For comparison, we also show how the number of patterns grows with sequence length. For longer sequences of therapy changes, the number of patterns approaches the number of patients, indicating that most patients follow a distinct treatment sequence.

In Figure 5.2, we summarize the most common treatment patterns of length $k = 3$. In total, 2,615 patients underwent at least three therapy changes. A frequent theme among these patterns is therapy cycling, where patients return to a drug class previously used. For example, 423 patients resumed a TNF inhibitor combination therapy after a period on a csDMARD-only regimen. Similarly, 85 patients returned to TNF inhibitor monotherapy after a period without any DMARD treatment. Among patients who switched to a new b/tsDMARD as their third-line therapy, combination therapies involving JAK inhibitors and abatacept—a bDMARD—were the most common.

From the perspective of proposing and evaluating alternative treatment strategies based on these data, the diversity in treatment selection presents both opportunities and challenges. On the one hand, variation in clinical practice allows for the evaluation of strategies that deviate from standard care. On the other hand, statistically robust evaluation of such strategies requires large datasets. As a result, in Section 5.4, we propose a pragmatic approach to derive target policy candidates from a model of the behavior policy, allowing control over the degree of overlap between target and behavior policies. We compare this approach to policies learned using reinforcement learning techniques, which we introduce in the following sections.

5.2 Reinforcement Learning

Reinforcement learning is a machine learning paradigm for solving sequential decision-making problems. An RL problem is typically formalized as an MDP $(\mathcal{S}, \mathcal{A}, \mathcal{R}, p)$ (see Section 3.1). The goal of RL is to learn an optimal policy π^*

[1] See the paper for the full list of included therapies.

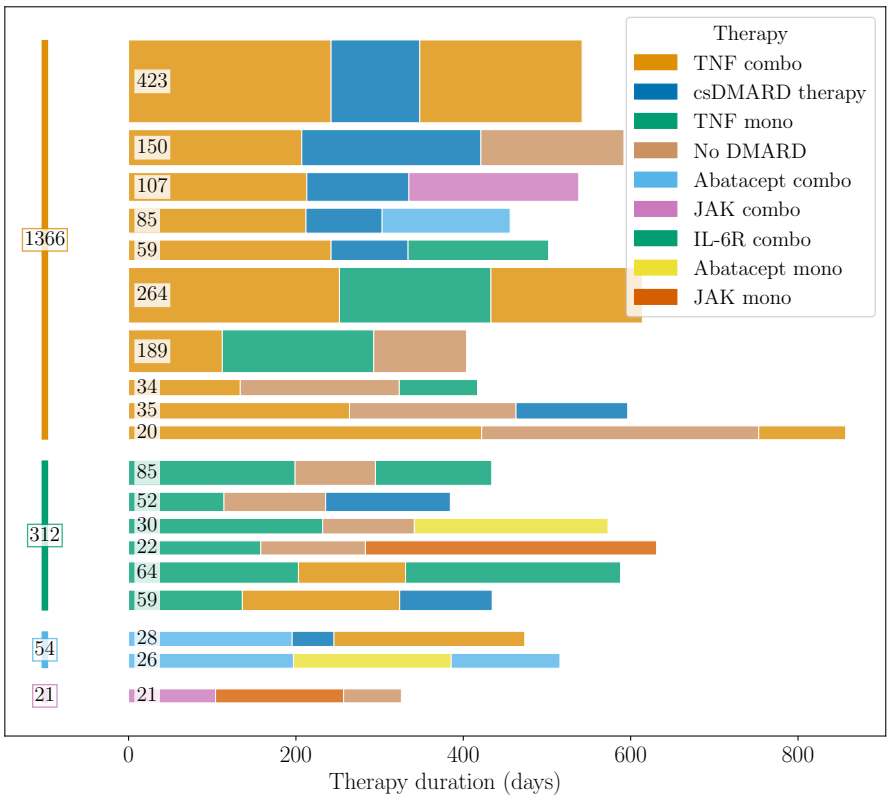


Figure 5.2: The most common treatment patterns of length tree in a cohort of 6,015 patients extracted from the CorEvitas RA registry. A treatment pattern is defined as a unique sequence of therapy changes starting from and including the first-line b/tsDMARD therapy, which was defined as the baseline. In total, 2,615 patients underwent at least two therapy changes after baseline. Only patterns observed in at least 20 patients are shown. The length of each segment reflects the average duration of that therapy among patients in the cohort.

that maximizes the expected return:

$$\pi^* = \arg\max_{\pi} V^{\pi}, \tag{5.1}$$

where V^{π} denotes the value of policy π, as defined in Equation (3.3).

RL algorithms can broadly be categorized into model-based and model-free methods. Model-based RL assumes that the dynamics of the environment, $p(S_{t+1}, R_t \mid S_t, A_t)$, are either known or can be learned. In contrast, model-free RL makes no such assumptions about the dynamics. Model-free RL algorithms are further divided into value-based and policy-based approaches. On the one hand, value-based methods focus on learning action values by estimating the action-value function $Q(s, a)$. On the other hand, policy-based methods

directly learn a parameterized policy without necessarily relying on any value estimates.

Providing a comprehensive overview of reinforcement learning algorithms is beyond the scope of this thesis. Instead, we focus on two algorithms that are used in Paper III and Paper V: policy iteration and Q-learning.

Policy Iteration

When the dynamics $p(S_{t+1}, R_t \mid S_t, A_t)$ of an MDP with finite state and action spaces are known, it is possible to use dynamic programming (DP) to compute an optimal policy π^*. DP relies on the Bellman equation for the state-value function $V^\pi(s)$ under a deterministic policy π (Sutton & Barto, 2018, Chapter 3.5):

$$V^\pi(s) = \sum_{s',r} p(s', r \mid s, \pi(s)) \left[r + V^\pi(s')\right]. \tag{5.2}$$

Because DP assumes knowledge of the dynamics, it is considered a model-based reinforcement learning approach.

A classic DP algorithm is policy iteration, which alternates between evaluating a given policy and improving it. The algorithm begins with an arbitrary initial policy π, for example $\pi(s) = a$ for all $s \in \mathcal{S}$, and arbitrary initial values, such as $V^\pi(s) = 0$ for all $s \in \mathcal{S}$. In each iteration, the algorithm first performs policy evaluation by updating $V^\pi(s)$ for all $s \in \mathcal{S}$ according to Equation (5.2) until convergence. Then, it performs policy improvement by updating the policy as follows:

$$\pi(s) = \arg\max_a \sum_{s',r} p(s', r \mid s, a)[r + V^\pi(s')]. \tag{5.3}$$

The policy improvement step yields a greedy policy, meaning a policy that always selects the action with the highest estimated value. According to the policy improvement theorem, the new policy is guaranteed to perform at least as well as the previous one (Sutton & Barto, 2018, Chapter 4.2). As a result, repeated application of policy evaluation and improvement ensures that policy iteration converges to an optimal policy π^*.

As discussed in previous chapters, the dynamics of the environment are typically unknown in clinical contexts. However, if the state and action spaces are finite and a model $\hat{p}(s_{t+1}, r_t \mid s_t, a_t)$ is available, it may still be possible to apply policy iteration in such settings. For example, Komorowski et al. (2018) constructed a finite state space by clustering continuous states and assigning each of them to its nearest cluster centroid; the set of centroids then defined the discrete states. Given observed transitions (s_t, a_t, r_t, s_{t+1}), one can estimate the transition probabilities $\hat{p}(s' \mid s, a)$ and the expected reward $\hat{r}(s, a, s')$, where $r(s, a, s') \coloneqq \mathbb{E}[R_t \mid S_t = s, A_t = a, S_{t+1} = s']$, using the

following formulas (Moerland et al., 2023):

$$\hat{p}(s' \mid s, a) = \frac{n(s, a, s')}{\sum_{s'} n(s, a, s')} \quad \text{and} \quad \hat{r}(s, a, s') = \frac{1}{n(s, a, s')} \sum_{i:(s_i=s, a_i=a, s'_i=s')} r_i, \tag{5.4}$$

where $n(s, a, s')$ denotes the number of times the transition from state s to state s' occurred after taking action a.

Q-Learning

Q-learning is arguably the most widely used model-free RL algorithm in practice. As the name suggests, Q-learning estimates the action-value function, or Q-function, $Q(s, a)$. Assuming a finite state space, the classical Q-learning algorithm (Watkins & Dayan, 1992) iteratively updates the initial action-values—which may be chosen arbitrarily—based on observations (s_t, a_t, r_t, s_{t+1}) according to:

$$Q(s, a) \leftarrow Q(s, a) + \alpha \left[r + \gamma \max_{a'} Q(s', a') - Q(s, a) \right], \tag{5.5}$$

where α is a step size parameter and $\gamma \in [0, 1]$ is a discount factor that determines the relative importance of future versus immediate rewards. A higher discount factor places greater emphasis on long-term rewards, while a lower value prioritizes short-term gains.

Classical Q-learning is an online algorithm, meaning it assumes access to the environment during learning. Starting from an initial state $s_1 \sim p(S_1)$, the action-values are incrementally updated as the agent interacts with the environment. For each visited state s_t, the action a_t is chosen according to a policy derived from the current action-values. A common choice is the epsilon-greedy policy: with probability $1 - \epsilon$, the agent selects the action $\arg\max_a Q(s_t, a)$, and with probability ϵ, it selects an action uniformly at random.

In practice, the state space may be infinite or too large for a tabular representation of the Q-function. In such cases, the Q-function can be approximated using a parameterized function, such as a neural network with parameters θ. Let $Q_\theta(s, a)$ denote this parameterized Q-function. The parameters θ can be updated via gradient descent to minimize the squared error between the current estimate $Q_\theta(s, a)$ and the target $r + \gamma \max_{a'} Q_\theta(s', a')$:

$$\theta \leftarrow \theta - \alpha \nabla_\theta \left(r + \gamma \max_{a'} Q_\theta(s', a') - Q_\theta(s, a) \right)^2. \tag{5.6}$$

Intuitively, this update step encourages the Q-function to satisfy the Bellman equation for the optimal Q-function Q^*:

$$Q^*(s, a) = \mathbb{E}\left[R_t + \gamma \max_{a'} Q^*(S_{t+1}, a') \mid S_t = s, A_t = a \right]. \tag{5.7}$$

There are many variants of the original Q-learning algorithm. For example, the fitted Q-learning algorithm (Ernst et al., 2005) updates parameters based

on a batch of transitions rather than a single one. Modern approaches often rely on a replay buffer (L. Lin, 1992), alternating between adding new transitions to the buffer and updating the parameters using a sampled batch. This approach is, for example, used in the deep Q-network (DQN) algorithm (Mnih et al., 2013). It is also common to use a lagged version of the parameters, denoted by θ', to compute the targets (Mnih et al., 2013).

5.3 Offline Reinforcement Learning

In healthcare, it is typically unrealistic to assume that an agent can interact with the environment during learning. Offline reinforcement learning refers to the setting in which the agent must learn solely from a static dataset of transitions, $\mathcal{D} = \left\{\left(s_t^{(i)}, a_t^{(i)}, r_t^{(i)}, s_{t+1}^{(i)}\right)\right\}_{i=1}^{n}$. This setting is generally more appropriate in clinical contexts than the classic online setting. In practice, $\mathcal{D}$ can be derived from a dataset of trajectories $\mathcal{D}_\mu$ collected under a behavior policy μ.

Fitted Q-learning algorithms such as DQN can be adapted to the offline setting by pre-populating the replay buffer with transitions from $\mathcal{D}$, effectively solving the following optimization problem:

$$\arg\min_{\theta} \underbrace{\mathbb{E}_{s,a,r,s' \sim \mathcal{D}}\left[\left(r + \gamma \max_{a'} Q_{\theta'}(s', a') - Q_\theta(s, a)\right)^2\right]}_{J_{\mathrm{DQN}}(\theta)}. \tag{5.8}$$

However, as discussed by Levine et al. (2020), naively applying Q-learning in this way may fail due to distributional shift between the state-action distribution observed during training and that encountered during deployment. In particular, actions that are rarely observed in the dataset often have overestimated values (Kumar et al., 2020). To address this issue, a common approach is to constrain the learned policy π to remain close to the behavior policy μ (Fujimoto et al., 2019; Kumar et al., 2019; Siegel et al., 2020), or to regularize the Q-network in order to avoid overestimation for out-of-distribution actions (Kumar et al., 2020). We discuss two such methods in the remainder of this section.

Batch Constrained Q-Learning

A key challenge in applying Q-learning to the fully offline setting is that the target value, $r + \gamma \max_{a'} Q^\pi(s', a')$, is computed using a target policy that is implicitly defined by the Q-function: $\pi(s) = \arg\max_{a'} Q^\pi(s, a')$. If the resulting action distribution $p_\pi(A = a \mid S = s) = \delta(a = \arg\max_{a'} Q^\pi(s, a'))$, where δ denotes the Kronecker delta, differs substantially from the behavior policy $p_\mu(A \mid S)$ observed in the training data, the target value may become highly unreliable (Levine et al., 2020). Because the target policy is optimized to maximize action-values, it can become biased toward out-of-distribution actions whose values are erroneously overestimated (Kumar et al., 2020). In

the online setting, such overestimations are naturally corrected as the agent receives feedback from actual transitions. However, in the offline setting, this correction is not possible.

Batch-Constrained Q-learning (BCQ) (Fujimoto et al., 2019) addresses this problem by restricting the distribution over actions used to compute the target values, ensuring it stays close to the behavior policy distribution $p_\mu(A \mid S)$ during training. The learning objective can written as

$$\arg\min_\theta \mathbb{E}_{s,a,r,s' \sim \mathcal{D}} \left[\left(r + \gamma \max_{a' \in \mathcal{A}_\phi(s')} Q_{\theta'}(s', a') - Q_\theta(s, a) \right)^2 \right], \tag{5.9}$$

where $\mathcal{A}_\phi(s')$ denotes a set of actions in state s' that are likely under the behavior policy. This set is generated using a conditional variational autoencoder (Kingma & Welling, 2013) G with parameters ϕ, which learns to model the behavior policy's action distribution: $\mathcal{A}_\phi(s') = \{a_i \sim G_\phi(s')\}$. After training, a policy can then be constructed by sampling candidate actions from $G_\phi(s')$ and selecting the one with the highest Q-value under Q_θ.

Conservative Q-Learning

Conservative Q-Learning (CQL), proposed by Kumar et al. (2020), addresses the issue of overestimated action-values by learning a conservative estimate of the Q-function—that is, one that serves as a lower bound on the true values. In practice, this is done by adding a regularization term to the standard Q-learning objective:

$$\arg\min_\theta J_{\mathrm{DQN}}(\theta) + \lambda \mathbb{E}_{s \sim \mathcal{D}} \left[\log \sum_a \exp(Q_\theta(s, a)) - \mathbb{E}_{a \sim \hat{p}_\mu(\cdot|s)}[Q_\theta(s, a)] \right], \tag{5.10}$$

where λ is a regularization parameter, and $\hat{p}_\mu(\cdot \mid s)$ denotes the empirical distribution over actions taken in state s under the behavior policy. This regularization penalizes Q-functions that assign high values to actions not seen in the data, thereby reducing overestimation for out-of-distribution actions.

5.4 Policy Refinement via Interpretable Behavior Modeling

While applying offline RL in clinical settings holds great promise, it also presents several well-known challenges (Jayaraman et al., 2024). One major difficulty is performing reliable off-policy evaluation when there are substantial differences between the target and behavior policies. This issue is especially pronounced with importance sampling-based techniques, but it also affects other methods. For instance, the direct methods described in Section 4.3 rely on extrapolation to account for the mismatch between policies. Although constraining the policy learning process can help mitigate this issue (see Section 5.3), another key challenge is interpretability: policies that are difficult to understand may

struggle to gain trust within the medical community (Lipton, 2017; Pace et al., 2022).

Motivated by a growing interest in transparent decision-making, interpretable reinforcement learning has emerged as an active area of research. For instance, decision trees can be used to parameterize either the action-value function (Ernst et al., 2005) or the policy directly (Likmeta et al., 2020; Silva et al., 2020). Differentiable decision trees enable gradient-based optimization (Pace et al., 2022; Silva et al., 2020). Another approach involves reformulating the MDP into an equivalent MDP whose optimal solution corresponds to a decision tree policy in the original problem formulation (Topin et al., 2021). Beyond decision trees, interpretable RL methods have also employed, for example, first-order logic (Delfosse et al., 2023; Z. Jiang & Luo, 2019) and program synthesis (Qiu & Zhu, 2022; Verma et al., 2019).

Most existing approaches to interpretable RL are not designed for the offline setting. Combining ideas from offline and interpretable RL to learn robust, transparent policies solely from batch data is a promising direction for future work. As a pragmatic starting point, however, we propose using behavior policy modeling to derive interpretable and evaluable target policies for clinical decision-making. This approach rests on two key principles. First, by constructing the target policy based on the most frequently chosen treatments in each state, as estimated by the behavior policy model, we can ensure sufficient overlap between the target and behavior policies, making the target policy amenable to reliable off-policy evaluation. Second, by using an interpretable model for the behavior policy, we preserve transparency in the derived target policy, facilitating trust and understanding among practitioners.

In Paper V, we implement this idea using the tree-based meta-estimator introduced in Section 3.4 as the behavior policy model. Once trained, decision trees naturally partition the state space with respect to treatment, making them well-suited for identifying common treatment patterns across groups of patients (Keramati et al., 2022). In contrast, while sparse generalized linear models are also interpretable, they do not induce such a natural partitioning. Assuming that the state captures all relevant confounding variables, variation in treatment across the leaves of a trained decision tree reflects clinical practice variation that is not due to confounding. Importantly, matching subjects on the propensity score is sufficient to adjust for confounding when estimating causal effects (Rosenbaum & Rubin, 1983).

We construct target policy candidates based on the set of the k actions with the highest probability in state s_t under the behavior policy model $\hat{\mu}$, denoted as Top-$k(s_t; \hat{\mu})$. Formally, we define the "most common" (`MC`) target policy as:

$$p_\pi(A_t = a_t \mid S_t = s_t) \coloneqq \begin{cases} \hat{p}_\mu\left(a_t \mid S_t = s_t\right)/Z_k, & \text{if } a_t \in \text{Top-}k(s_t; \hat{\mu}); \\ 0, & \text{otherwise,} \end{cases} \tag{5.11}$$

where the normalization constant $Z_k = \sum_{a \in \text{Top-}k(s_t;\hat{\mu})} \hat{p}_\mu(a \mid S_t = s_t)$ ensures that p_π defines a valid probability distribution. By adjusting the parameter k, we can control the degree of overlap between the target and behavior policies. When $k = 1$, the resulting policy is deterministic, recommending for each state

s the single most commonly chosen treatment according to the behavior policy model. For $1 < k < K$, the policy becomes stochastic, emphasizing the k most frequently used treatments while ignoring the remaining $K - k$.

We can extend this idea to account for observed outcomes among patients in each leaf of the trained decision tree. Let $O(s, a; \hat{\mu})$ denote the average observed outcome for patients in state s who received treatment a under the behavior policy model $\hat{\mu}$. Then, in state s_t, an outcome-guided target policy (`MC+O`) selects the action

$$\underset{a \in \text{Top-}k(s_t;\hat{\mu})}{\arg\max} \quad O(s_t, a; \hat{\mu}), \tag{5.12}$$

that is, the treatment with the highest observed outcome among the k most common treatments in that state. While this approach may yield policies with higher estimated value than those based solely on treatment frequency, it is sensitive to unmeasured confounders. If such confounders are present within the leaves of a fully grown tree, the estimated value of an outcome-guided policy may be biased and potentially overstated.

We evaluate our approach using the case studies of sepsis management and treatment selection in RA, as introduced in Chapter 3. For sepsis, we adopt a standard reward formulation used in prior work (Komorowski et al., 2018; Luo et al., 2024), assigning a positive reward for patient survival and a penalty for death. In the RA case, we define the reward function as $R_t := 10 - I_{t+1}$, where I is the clinical disease activity index (CDAI).[2] In Table 5.1, we report the estimated value using weighted importance sampling (Precup et al., 2000) of policies derived using our framework, alongside policies learned using several RL methods: standard Q-learning applied to a finite MDP with estimated parameters (QL), DQN, BCQ, and CQL; see Section 5.3. We compare these policies to a random policy, which selects a treatment uniformly at random for each state, as well as the behavior policy followed by clinicians. In addition to the estimated policy value, we report the effective sample size (see Equation (4.9)) to indicate the reliability of the evaluation.

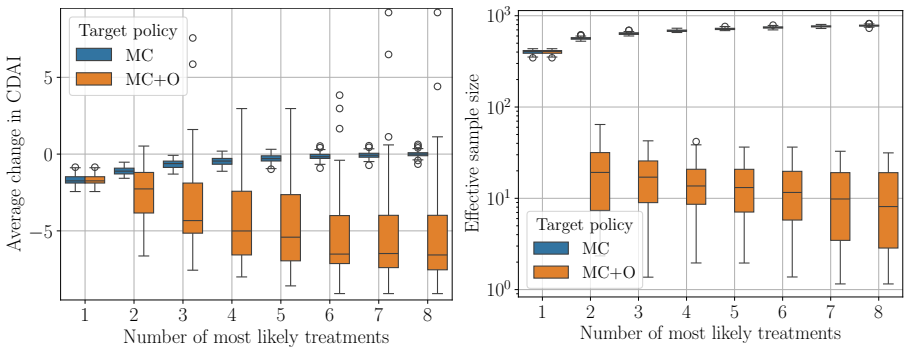
In both case studies, we find that the target policy based on the most common treatment ($k = 1$) is, on average, estimated to have a higher value than the behavior policy. This difference is particularly pronounced in the RA case. While RL-based policies show promise—especially for sepsis—their estimated values tend to exhibit high variance, resulting in small effective sample sizes. In contrast, policies derived under our framework offer direct control over variance through the parameter k. For the `MC` policies, increasing k reduces variance; interestingly, in the RA case, the variance remains relatively constant across value estimates. For the `MC+O` policies, the opposite holds: reducing k leads to lower variance. While the outcome-guided policies appear promising in terms of estimated value, the high variance suggests that these estimates should be interpreted with caution.

In Figure 5.3, we take a closer look at the RA results for the `MC` and `MC+O` policies across different values of k. By normalizing the estimated target policy

[2] A CDAI of 10 marks the threshold between low and moderate-to-high disease activity.

Table 5.1: The average value estimate $\hat{V}$ using weighted importance sampling (WIS) and effective sample size (ESS) for different target policies in RA and sepsis. The value of the behavior policy is estimated as the average reward in the data. The confidence intervals represent the interquartile range of each distribution.

	RA		**Sepsis**	
Target policy	$\hat{V}^{\pi}_{\text{WIS}}$ (↑)	ESS (↑)	$\hat{V}^{\pi}_{\text{WIS}}$ (↑)	ESS (↑)
`MC` $(k = 1)$	0.7 (0.4, 0.8)	406.1 (388.1, 415.6)	74.1 (66.8, 82.9)	64.1 (46.1, 80.8)
`MC` $(k = 2)$	0.0 (−0.2, 0.2)	566.1 (553.3, 575.0)	74.6 (72.0, 79.4)	277.7 (250.6, 296.0)
`MC` $(k = 3)$	−0.5 (−0.6, −0.2)	639.2 (624.6, 650.3)	75.1 (73.7, 76.6)	628.9 (604.9, 647.7)
`MC+O` $(k = 1)$	0.7 (0.4, 0.8)	406.1 (388.1, 415.6)	74.1 (66.8, 82.9)	64.1 (46.1, 80.8)
`MC+O` $(k = 2)$	1.2 (0.1, 2.7)	19.2 (7.4, 31.7)	80.7 (75.7, 93.9)	15.5 (7.8, 24.7)
`MC+O` $(k = 3)$	3.2 (0.8, 4.1)	17.1 (9.0, 25.7)	85.3 (68.5, 95.5)	6.9 (3.0, 14.2)
RL (QL)	0.0 (−4.2, 3.7)	3.0 (2.1, 4.3)	86.0 (80.3, 92.3)	14.0 (6.8, 24.5)
RL (DQN)	0.0 (−4.3, 3.0)	5.3 (2.4, 9.5)	89.3 (70.9, 98.9)	1.7 (1.1, 8.0)
RL (BCQ)	−1.8 (−4.6, 0.4)	7.3 (4.4, 10.5)	83.0 (76.3, 88.2)	19.3 (12.7, 28.9)
RL (CQL)	−0.9 (−5.3, 1.2)	5.7 (2.8, 11.4)	67.3 (34.4, 85.2)	6.5 (2.7, 11.5)
Random	1.7 (−4.8, 5.1)	1.3 (1.0, 2.0)	97.9 (59.7, 99.6)	1.6 (1.1, 2.3)
Behavior policy	−1.1 (−1.2, −1.0)	779.0 (770.3, 788.8)	71.6 (70.7, 72.2)	2297.0 (2297.0, 2297.0)



(a) Average change in CDAI.

(b) Effective sample size.

Figure 5.3: Off-policy evaluation using weighted importance sampling of target policies derived from an interpretable model of the behavior policy in the case of RA. The target policies are based on the most commonly selected treatments under the model (`MC`), optionally incorporating observed outcomes (`MC+O`). Panel (a) shows the average change in CDAI that could be achieved by replacing the behavior policy with each target policy. As shown in panel (b), high variance in the estimates is typically associated with smaller effective sample sizes.

values relative to the value of the behavior policy, see Figure 5.3(a), we interpret the results as the average change in CDAI per patient and stage if the target policies were used instead of the behavior policy. Although the estimates for the outcome-guided policies (`MC+O`) exhibit high variance—consistent with the small effective sample sizes shown in Figure 5.3(b)—the `MC` policies with $k = \{1, 2, 3\}$

consistently suggest a potential reduction in CDAI through standardization of the most commonly used therapies. Given that practice variability is a well-documented issue in RA management (DeMaria et al., 2014), standardizing the most frequently used treatment patterns may improve care quality and reduce treatment costs.

Chapter 6

Concluding Remarks

In this thesis, we studied three interconnected areas of clinical decision-making: policy modeling, off-policy evaluation, and the development of alternative treatment strategies (policy refinement). Across all these tasks, we emphasized interpretability—supporting sanity checks, facilitating clinical validation, and enhancing end-user trust in the models. For policy modeling, we investigated different representations of patient history to enable accurate and interpretable descriptions of the observed behavior policy. We also proposed methods for leveraging structure in the data-generating process to further improve model interpretability. In the context of off-policy evaluation, we introduced the use of prototype learning to identify representative cases in the observed data, enabling interpretable comparisons between the target and behavior policies and extending evaluation beyond value estimation. Finally, we proposed a pragmatic approach to deriving alternative treatment strategies that can be reliably evaluated using an interpretable model of the behavior policy—effectively connecting the three focus areas.

Each of the included works has its own limitations, which are discussed in the appended papers (see Part II). However, there are a few overarching limitations worth highlighting. First, this thesis primarily focuses on off-policy evaluation based on importance sampling (IS). While IS-based techniques are widely used in practice, alternative methods—such as doubly robust estimators and model-based approaches—can sometimes provide lower variance and more reliable value estimates. Second, the datasets used in this thesis are mainly tabular and relatively small in size. In contrast, much of the data in healthcare is unstructured (Kong, 2019), highlighting the need for models capable of handling multi-modal data. Third, this thesis assumes the absence of unmeasured confounders. In practice, however, we are limited to observed variables, and unmeasured confounding may exist in both the rheumatoid arthritis and sepsis case studies.

As a direction for future work, it remains an open question how the bias introduced by restricting behavior policy models to use prototypes propagates to the importance weights and the estimated policy values. In the meantime, we recommend a pragmatic approach to using prototypes in off-policy evaluation:

use them primarily for diagnostic and interpretability purposes, and, if needed, fit a more accurate behavior policy model to compute importance weights. Another promising avenue is the integration of interpretable reinforcement learning (RL) with offline RL, which could enable the development of optimal policies that are both robust to distributional shifts and interpretable to end users. Alternatively, it would be interesting to investigate how the set of (interpretable) policies that can be reliably evaluated might itself be learned. Finally, our results on different representations of patient history suggest that stage-dependent representations offer a compelling direction for future research.

The potential impact of this work lies in applying machine learning to improve and personalize clinical decision-making based on observational data, particularly for chronic conditions such as rheumatoid arthritis. From a practical standpoint, we believe that the emphasis on interpretability can enhance the reliability of new policies and their estimated values, thereby increasing the potential utility of the policies in clinical practice. We also hope that the focus on interpretable models will help bridge the gap between computer scientists and medical practitioners, as such models facilitate dialogue and build trust among end users. Finally, we believe that our framework of missingness-avoiding machine learning offers a new perspective on how missing data should be handled in clinical applications.

During the time this research was conducted, the field of machine learning and artificial intelligence (AI) underwent dramatic changes. Most notably, the rise of foundational models—especially large language models—has opened up new possibilities, making AI a central topic of public and scientific discussion. In the healthcare domain, a medical foundational model could provide treatment recommendations (Moor et al., 2023), effectively serving as a policy for personalized decision-making. However, while these models are capable of explaining their reasoning (Huang & Chang, 2022; Wei et al., 2022), their internal workings remain complex and intractable, underscoring the need to validate their behavior against current medical practice.

Bibliography

Aamodt, A., & Plaza, E. (1994). Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches. *AI Communications*, *7*(1), 39–59 (cit. on p. 13).

Afnan, M. A. M., Rudin, C., Conitzer, V., Savulescu, J., Mishra, A., Liu, Y., & Afnan, M. (2021). Ethical Implementation of Artificial Intelligence to Select Embryos in In Vitro Fertilization. *Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society*, 316–326 (cit. on p. 9).

Aletaha, D., & Smolen, J. (2005). The Simplified Disease Activity Index (SDAI) and the Clinical Disease Activity Index (CDAI): A Review of Their Usefulness and Validity in Rheumatoid Arthritis. *Clinical and Experimental Rheumatology*, *23*(5), S100–S108 (cit. on p. 37).

Chakraborty, B., & Moodie, E. E. (2013). *Statistical Methods for Dynamic Treatment Regimes* (Vol. 2). Springer. (Cit. on pp. 3, 34).

Crump, R. K., Hotz, V. J., Imbens, G. W., & Mitnik, O. A. (2009). Dealing With Limited Overlap in Estimation of Average Treatment Effects. *Biometrika*, *96*(1), 187–199 (cit. on p. 38).

Delfosse, Q., Shindo, H., Dhami, D., & Kersting, K. (2023). Interpretable and Explainable Logical Policies via Neurally Guided Symbolic Abstraction. *Advances in Neural Information Processing Systems*, *36*, 50838–50858 (cit. on p. 51).

DeMaria, L., Acelajado, M. C., Luck, J., Ta, H., Chernoff, D., Florentino, J., & Peabody, J. W. (2014). Variations and Practice in the Care of Patients With Rheumatoid Arthritis: Quality and Cost of Care. *JCR: Journal of Clinical Rheumatology*, *20*(2), 79–86 (cit. on p. 54).

Deuschel, J., Ellington, C., Luo, Y., Lengerich, B., Friederich, P., & Xing, E. P. (2024). Contextualized Policy Recovery: Modeling and Interpreting Medical Decisions With Adaptive Imitation Learning. *Proceedings of the 41st International Conference on Machine Learning*, *PMLR 235*, 10642–10660 (cit. on p. 4).

Ernst, D., Geurts, P., & Wehenkel, L. (2005). Tree-Based Batch Mode Reinforcement Learning. *Journal of Machine Learning Research*, *6*, 503–556 (cit. on pp. 6, 48, 51).

Farajtabar, M., Chow, Y., & Ghavamzadeh, M. (2018). More Robust Doubly Robust Off-Policy Evaluation. *Proceedings of the 35th International*

Conference on Machine Learning, *PMLR 80*, 1447–1456 (cit. on pp. 5, 36).

Fisher, R. A. (1936). The Use of Multiple Measurements in Taxonomic Problems. *Annals of Eugenics*, *7*(2), 179–188 (cit. on p. 12).

Fletcher, A., Lassere, M., March, L., Hill, C., Barrett, C., Carroll, G., & Buchbinder, R. (2022). Patterns of Biologic and Targeted-Synthetic Disease-Modifying Antirheumatic Drug Use in Rheumatoid Arthritis in Australia. *Rheumatology*, *61*(10), 3939–3951 (cit. on p. 44).

Frosst, N., & Hinton, G. (2017). Distilling a Neural Network Into a Soft Decision Tree. *arXiv preprint arXiv:1711.09784* (cit. on p. 14).

Fujimoto, S., Meger, D., & Precup, D. (2019). Off-Policy Deep Reinforcement Learning Without Exploration. *Proceedings of the 36th International Conference on Machine Learning*, *PMLR 97*, 2052–2062 (cit. on pp. 6, 49, 50).

Glanois, C., Weng, P., Zimmer, M., Li, D., Yang, T., Hao, J., & Liu, W. (2024). A Survey on Interpretable Reinforcement Learning. *Machine Learning*, *113*(8), 5847–5890 (cit. on p. 6).

Gottesman, O., Johansson, F., Meier, J., Dent, J., Lee, D., Srinivasan, S., Zhang, L., Ding, Y., Wihl, D., Peng, X., et al. (2018). Evaluating Reinforcement Learning Algorithms in Observational Health Settings. *arXiv preprint arXiv:1805.12298* (cit. on pp. 3–6, 36).

Gotts, J. E., & Matthay, M. A. (2016). Sepsis: Pathophysiology and Clinical Management. *BMJ*, *353*, i1585 (cit. on p. 21).

Guidotti, R., Monreale, A., Ruggieri, S., Turini, F., Giannotti, F., & Pedreschi, D. (2018). A Survey of Methods for Explaining Black Box Models. *ACM Computing Surveys*, *51*(5), 1–42 (cit. on p. 9).

Huang, J., & Chang, K. C. (2022). Towards Reasoning in Large Language Models: A Survey. *arXiv preprint arXiv:2212.10403* (cit. on p. 56).

Hüyük, A., Jarrett, D., & van der Schaar, M. (2021). Explaining by Imitating: Understanding Decisions by Interpretable Policy Learning. *Proceedings of the 9th International Conference on Learning Representations* (cit. on pp. 4, 21).

Jayaraman, P., Desman, J., Sabounchi, M., Nadkarni, G. N., & Sakhuja, A. (2024). A Primer on Reinforcement Learning in Medicine for Clinicians. *NPJ Digital Medicine*, *7*(1), 337 (cit. on pp. 19, 50).

Jiang, N., & Li, L. (2016). Doubly Robust Off-Policy Value Evaluation for Reinforcement Learning. *Proceedings of the 33rd International Conference on Machine Learning*, *PMLR 48*, 652–661 (cit. on pp. 5, 36).

Jiang, Z., & Luo, S. (2019). Neural Logic Reinforcement Learning. *Proceedings of the 36th International Conference on Machine Learning*, *PMLR 97*, 3110–3119 (cit. on p. 51).

Johnson, A. E., Pollard, T. J., Shen, L., Lehman, L. H., Feng, M., Ghassemi, M., Moody, B., Szolovits, P., Anthony Celi, L., & Mark, R. G. (2016). MIMIC-III, A Freely Accessible Critical Care Database. *Scientific Data*, *3*(1), 1–9 (cit. on p. 7).

Keramati, R., Gottesman, O., Celi, L. A., Doshi-Velez, F., & Brunskill, E. (2022). Identification of Subgroups With Similar Benefits in Off-Policy

Policy Evaluation. *Proceedings of the Conference on Health, Inference, and Learning*, *174*, 397–410 (cit. on p. 51).

Keystone, E., Emery, P., Peterfy, C. G., Tak, P. P., Cohen, S., Genovese, M. C., Dougados, M., Burmester, G. R., Greenwald, M., Kvien, T. K., et al. (2009). Rituximab Inhibits Structural Joint Damage in Patients With Rheumatoid Arthritis With an Inadequate Response to Tumour Necrosis Factor Inhibitor Therapies. *Annals of the Rheumatic Diseases*, *68*(2), 216–221 (cit. on p. 44).

Kingma, D. P., & Welling, M. (2013). Auto-Encoding Variational Bayes. *arXiv preprint arXiv:1312.6114* (cit. on p. 50).

Koller, D., & Friedman, N. (2009). *Probabilistic Graphical Models: Principles and Techniques*. MIT press. (Cit. on p. 18).

Komorowski, M., Celi, L. A., Badawi, O., Gordon, A. C., & Faisal, A. A. (2018). The Artificial Intelligence Clinician Learns Optimal Treatment Strategies for Sepsis in Intensive Care. *Nature Medicine*, *24*(11), 1716–1720 (cit. on pp. 19, 21, 37–39, 47, 52).

Kong, H. (2019). Managing Unstructured Big Data in Healthcare System. *Healthcare Informatics Research*, *25*(1), 1–2 (cit. on p. 55).

Kostrikov, I., Nair, A., & Levine, S. (2022). Offline Reinforcement Learning With Implicit Q-Learning. *Proceedings of the 10th International Conference on Learning Representations* (cit. on p. 6).

Kremer, J. M. (2016). The Corrona US Registry of Rheumatic and Autoimmune Diseases. *Clinical and Experimental Rheumatology*, *34*(5 (Suppl. 101)), S96–S99 (cit. on pp. 7, 44).

Kumar, A., Fu, J., Soh, M., Tucker, G., & Levine, S. (2019). Stabilizing Off-Policy Q-Learning via Bootstrapping Error Reduction. *Advances in Neural Information Processing Systems*, *32*, 11784–11794 (cit. on p. 49).

Kumar, A., Zhou, A., Tucker, G., & Levine, S. (2020). Conservative Q-Learning for Offline Reinforcement Learning. *Advances in Neural Information Processing Systems*, *33*, 1179–1191 (cit. on pp. 6, 49, 50).

Le, H., Voloshin, C., & Yue, Y. (2019). Batch Policy Learning Under Constraints. *Proceedings of the 36th International Conference on Machine Learning*, *PMLR 97*, 3703–3712 (cit. on p. 36).

Le Morvan, M., Josse, J., Moreau, T., Scornet, E., & Varoquaux, G. (2020). NeuMiss Networks: Differentiable Programming for Supervised Learning With Missing Values. *Advances in Neural Information Processing Systems*, *33*, 5980–5990 (cit. on p. 30).

Levine, S., Kumar, A., Tucker, G., & Fu, J. (2020). Offline Reinforcement Learning: Tutorial, Review, and Perspectives on Open Problems. *arXiv preprint arXiv:2005.01643* (cit. on pp. 6, 49).

Li, O., Liu, H., Chen, C., & Rudin, C. (2018). Deep Learning for Case-Based Reasoning Through Prototypes: A Neural Network That Explains Its Predictions. *Proceedings of the AAAI Conference on Artificial Intelligence*, *32*, 3530–3537 (cit. on pp. 6, 38).

Likmeta, A., Metelli, A. M., Tirinzoni, A., Giol, R., Restelli, M., & Romano, D. (2020). Combining Reinforcement Learning With Rule-Based Con-

trollers for Transparent and General Decision-Making in Autonomous Driving. *Robotics and Autonomous Systems*, *131*, 103568 (cit. on p. 51).

Lin, J., Zhong, C., Hu, D., Rudin, C., & Seltzer, M. (2020). Generalized and Scalable Optimal Sparse Decision Trees. *Proceedings of the 37th International Conference on Machine Learning*, *PMLR 119*, 6150–6160 (cit. on p. 12).

Lin, L. (1992). Self-Improving Reactive Agents Based on Reinforcement Learning, Planning and Teaching. *Machine Learning*, *8*, 293–321 (cit. on p. 49).

Lipton, Z. C. (2017). The Doctor Just Won't Accept That! *arXiv preprint arXiv:1711.08037* (cit. on p. 51).

Luo, Z., Pan, Y., Watkinson, P., & Zhu, T. (2024). Position: Reinforcement Learning in Dynamic Treatment Regimes Needs Critical Reexamination. *Proceedings of the 41st International Conference on Machine Learning*, *PMLR 235*, 33432–33465 (cit. on pp. 21, 24, 52).

McTavish, H., Donnelly, J., Seltzer, M., & Rudin, C. (2024). Interpretable Generalized Additive Models for Datasets With Missing Values. *Advances in Neural Information Processing Systems*, *37* (cit. on p. 30).

Ming, Y., Xu, P., Qu, H., & Ren, L. (2019). Interpretable and Steerable Sequence Learning via Prototypes. *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 903–913 (cit. on pp. 6, 13, 38).

Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., & Riedmiller, M. (2013). Playing Atari With Deep Reinforcement Learning. *arXiv preprint arXiv:1312.5602* (cit. on pp. 6, 49).

Moerland, T. M., Broekens, J., Plaat, A., Jonker, C. M., et al. (2023). Model-Based Reinforcement Learning: A Survey. *Foundations and Trends® in Machine Learning*, *16*(1), 1–118 (cit. on p. 48).

Moor, M., Banerjee, O., Abad, Z. S. H., Krumholz, H. M., Leskovec, J., Topol, E. J., & Rajpurkar, P. (2023). Foundation Models for Generalist Medical Artificial Intelligence. *Nature*, *616*(7956), 259–265 (cit. on p. 56).

Namkoong, H., Keramati, R., Yadlowsky, S., & Brunskill, E. (2020). Off-Policy Policy Evaluation for Sequential Decisions Under Unobserved Confounding. *Advances in Neural Information Processing Systems*, *33*, 18819–188313 (cit. on pp. 20, 35).

Nixon, J., Dusenberry, M., Jerfel, G., Nguyen, T., Liu, J., Zhang, L., & Tran, D. (2019). Measuring Calibration in Deep Learning. *arXiv preprint arXiv:1904.01685* (cit. on p. 25).

Oberst, M., & Sontag, D. (2019). Counterfactual Off-Policy Evaluation With Gumbel-Max Structural Causal Models. *Proceedings of the 36th International Conference on Machine Learning*, *PMLR 97*, 4881–4890 (cit. on p. 42).

Owen, A. B. (2013). *Monte Carlo Theory, Methods and Examples.* https://artowen.su.domains/mc/. (Cit. on p. 37).

Pace, A., Chan, A., & van der Schaar, M. (2022). POETREE: Interpretable Policy Learning With Adaptive Decision Trees. *Proceedings of the 10th*

International Conference on Learning Representations (cit. on pp. 4, 14, 17, 21, 51).

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al. (2011). Scikit-Learn: Machine Learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830 (cit. on p. 12).

Precup, D., Sutton, R. S., & Singh, S. (2000). Eligibility Traces for Off-Policy Policy Evaluation. *Proceedings of the 17th International Conference on Machine Learning*, 759–766 (cit. on pp. 3, 5, 35, 36, 52).

Qiu, W., & Zhu, H. (2022). Programmatic Reinforcement Learning Without Oracles. *Proceedings of the 10th International Conference on Learning Representations* (cit. on p. 51).

Raghu, A., Gottesman, O., Liu, Y., Komorowski, M., Faisal, A., Doshi-Velez, F., & Brunskill, E. (2018). Behaviour Policy Estimation in Off-Policy Policy Evaluation: Calibration Matters. *arXiv preprint arXiv:1807.01066* (cit. on p. 21).

Raghu, A., Komorowski, M., Ahmed, I., Celi, L., Szolovits, P., & Ghassemi, M. (2017). Deep Reinforcement Learning for Sepsis Treatment. *arXiv preprint arXiv:1711.09602* (cit. on p. 21).

Rajkomar, A., Oren, E., Chen, K., Dai, A. M., Hajaj, N., Hardt, M., Liu, P. J., Liu, X., Marcus, J., Sun, M., et al. (2018). Scalable and Accurate Deep Learning With Electronic Health Records. *NPJ Digital Medicine*, *1*(1), 18 (cit. on p. 3).

Rosenbaum, P. R. (2010). *Design of Observational Studies* (2nd ed.). Springer. (Cit. on pp. 4, 37).

Rosenbaum, P. R., & Rubin, D. B. (1983). The Central Role of the Propensity Score in Observational Studies for Causal Effects. *Biometrika*, *70*(1), 41–55 (cit. on p. 51).

Rubin, D. B. (2005). Causal Inference Using Potential Outcomes: Design, Modeling, Decisions. *Journal of the American Statistical Association*, *100*(469), 322–331 (cit. on p. 34).

Rudin, C. (2019). Stop Explaining Black Box Machine Learning Models for High Stakes Decisions and Use Interpretable Models Instead. *Nature Machine Intelligence*, *1*(5), 206–215 (cit. on p. 4).

Rudin, C., Chen, C., Chen, Z., Huang, H., Semenova, L., & Zhong, C. (2022). Interpretable Machine Learning: Fundamental Principles and 10 Grand Challenges. *Statistic Surveys*, *16*, 1–85 (cit. on pp. 9–11, 13).

Sackett, D. L., Rosenberg, W. M., Gray, J. M., Haynes, R. B., & Richardson, W. S. (1996). Evidence Based Medicine: What It Is and What It Isn't. *BMJ*, *312*(7023), 71–72 (cit. on p. 3).

Salliot, C., Finckh, A., Katchamart, W., Lu, Y., Sun, Y., Bombardier, C., & Keystone, E. (2011). Indirect Comparisons of the Efficacy of Biological Antirheumatic Agents in Rheumatoid Arthritis in Patients With an Inadequate Response to Conventional Disease-Modifying Antirheumatic Drugs or to an Anti-Tumour Necrosis Factor Agent: A Meta-Analysis. *Annals of the Rheumatic Diseases*, *70*(2), 266–271 (cit. on p. 44).

Shortreed, S. M., Laber, E., Lizotte, D. J., Stroup, T. S., Pineau, J., & Murphy, S. A. (2011). Informing Sequential Clinical Decision-Making Through Reinforcement Learning: An Empirical Study. *Machine Learning*, *84*, 109–136 (cit. on p. 3).

Siegel, N., Springenberg, J. T., Berkenkamp, F., Abdolmaleki, A., Neunert, M., Lampe, T., Hafner, R., Heess, N., & Riedmiller, M. (2020). Keep Doing What Worked: Behavior Modelling Priors for Offline Reinforcement Learning. *Proceedings of the 8th International Conference on Learning Representations* (cit. on p. 49).

Silva, A., Gombolay, M., Killian, T., Jimenez, I., & Son, S. (2020). Optimization Methods for Interpretable Differentiable Decision Trees Applied to Reinforcement Learning. *Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics*, *PMLR 108*, 1855–1865 (cit. on pp. 6, 14, 51).

Smolen, J. S., Landewé, R. B., Bijlsma, J. W., Burmester, G. R., Dougados, M., Kerschbaumer, A., McInnes, I. B., Sepriano, A., Van Vollenhoven, R. F., De Wit, M., et al. (2020). EULAR Recommendations for the Management of Rheumatoid Arthritis With Synthetic and Biological Disease-Modifying Antirheumatic Drugs: 2019 Update. *Annals of the Rheumatic Diseases*, *79*(6), 685–699 (cit. on p. 44).

Solomon, D. H., Xu, C., Collins, J., Kim, S. C., Losina, E., Yau, V., & Johansson, F. D. (2021). The Sequence of Disease-Modifying Anti-Rheumatic Drugs: Pathways to and Predictors of Tocilizumab Monotherapy. *Arthritis Research & Therapy*, *23*, 1–9 (cit. on p. 44).

Stempfle, L., & Johansson, F. (2024). MINTY: Rule-Based Models That Minimize the Need for Imputing Features With Missing Values. *Proceedings of the 27th International Conference on Artificial Intelligence and Statistics*, *PMLR 238*, 964–972 (cit. on pp. 28–30).

Stiglic, G., Kocbek, P., Fijacko, N., Zitnik, M., Verbert, K., & Cilar, L. (2020). Interpretability of Machine Learning-Based Prediction Models in Healthcare. *WIREs Data Mining and Knowledge Discovery*, *10*(5), e1379 (cit. on p. 21).

Sutton, R. S., & Barto, A. G. (2018). *Reinforcement Learning: An Introduction* (2nd ed.). MIT Press. (Cit. on pp. 6, 18–20, 36, 42, 47).

Thomas, P., & Brunskill, E. (2016). Data-Efficient Off-Policy Policy Evaluation for Reinforcement Learning. *Proceedings of the 33rd International Conference on Machine Learning*, *PMLR 48*, 2139–2148 (cit. on pp. 5, 36).

Tibshirani, R. (1996). Regression Shrinkage and Selection via the Lasso. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, *58*(1), 267–288 (cit. on p. 11).

Topin, N., Milani, S., Fang, F., & Veloso, M. (2021). Iterative Bounding MDPs: Learning Interpretable Policies via Non-Interpretable Methods. *Proceedings of the AAAI Conference on Artificial Intelligence*, *35*, 9923–9931 (cit. on p. 51).

Uehara, M., Shi, C., & Kallus, N. (2022). A Review of Off-Policy Evaluation in Reinforcement Learning. *arXiv preprint arXiv:2212.06355* (cit. on p. 34).

Ustun, B., & Rudin, C. (2019). Learning Optimized Risk Scores. *Journal of Machine Learning Research*, *20*(150), 1–75 (cit. on p. 12).

Van Ness, M., Bosschieter, T. M., Halpin-Gregorio, R., & Udell, M. (2023). The Missing Indicator Method: From Low to High Dimensions. *Proceedings of the 29th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 5004–5015 (cit. on p. 27).

Verma, A., Le, H., Yue, Y., & Chaudhuri, S. (2019). Imitation-Projected Programmatic Reinforcement Learning. *Advances in Neural Information Processing Systems*, *32*, 15752–15763 (cit. on pp. 6, 51).

Voloshin, C., Le, H. M., Jiang, N., & Yue, Y. (2021). Empirical Study of Off-Policy Policy Evaluation for Reinforcement Learning. *Advances in Neural Information Processing Systems Datasets and Benchmarks*, *1* (cit. on pp. 6, 36).

Watkins, C. J., & Dayan, P. (1992). Q-Learning. *Machine Learning*, *8*, 279–292 (cit. on p. 48).

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E., Le, Q. V., Zhou, D., et al. (2022). Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. *Advances in Neural Information Processing Systems*, *35*, 24824–24837 (cit. on p. 56).

Zhao, S. S., Kearsley-Fleet, L., Bosworth, A., Watson, K., BSRBR-RA Contributors Group & Hyrich, K. L. (2022). Effectiveness of Sequential Biologic and Targeted Disease Modifying Anti-Rheumatic Drugs for Rheumatoid Arthritis. *Rheumatology*, *61*(12), 4678–4686 (cit. on p. 44).