How Should Language Models See Tree Models? Adaptive Representations for Reasoning

Anonymous authors
Paper under double-blind review

Abstract

Language models increasingly reason over compact artifacts produced by machine learning pipelines rather than raw data. For tabular problems, a fitted decision tree or boosted tree ensemble is a natural such artifact, yet the language model sees only a textual representation of it. Prior work typically fixes this representation, making its effect difficult to separate from the rest of the pipeline. We introduce a representation-controlled framework that varies only the tree-to-language representation, holding the fitted model, the language model, and the evaluation pipeline fixed. Across two tasks (rule generation and budgeted counterfactual search), five public datasets, and three language models, representation choice has a large and highly heterogeneous effect. No representation is consistently best across datasets or models, and on some datasets the spread exceeds sixty recall points.

This variability motivates studying the representation as part of an iterative inference process. We first study fixed-view iteration, which keeps the tree representation unchanged across rounds, and then introduce DRALoop, which re-points the exposed information toward the current unresolved state while keeping both models frozen. Iteration alone provides substantial gains over one-shot reasoning, while DRALoop adds beyond this matched control in regimes with residual headroom. On rule generation, it exceeds the best one-shot fixed representation on all five datasets for all three language models. On counterfactual search, its strongest configuration reaches 92.0% of the attainable validity bound, compared with 81.6% for matched fixed-view iteration. These results suggest that when one learned system reasons over another, the representation connecting them can be treated as a controllable and adaptive component of inference.

1 Introduction

Language models increasingly reason over learned models and structured intermediate artifacts rather than raw data alone. In tabular learning, prior work has explored serializing rows for language-model prediction (Hegselmann et al., 2023), using language models to construct features or transformations (Hollmann et al., 2023; Han et al., 2024; Abhyankar et al., 2026; Nam et al., 2024), and exposing tree-derived intermediates to language models (Liu & Mazumder, 2025; Ye et al., 2025). These results show that language models can make use of structured information from tabular pipelines, but the representation is typically fixed as part of the overall system and evaluated end to end. Its effect is therefore difficult to isolate from differences in the fitted model, downstream optimization, data access, or search procedure.

This leaves a basic question unresolved: *if the fitted model is held fixed, how much does the representation through which a language model sees that model matter?* A fitted tree or ensemble can be exposed as thresholds, root-to-leaf paths, class statistics or failure patterns. These views preserve different information about the same learned object, yet this interface is usually treated as an implementation choice rather than as a scientific variable in its own right.

Tree and boosted-tree models provide a clean setting in which to study this question. They capture nontrivial tabular structure while admitting multiple interpretable textual views of the same fitted model. We therefore introduce a *representation-controlled* formulation in which the fitted model, language model, parser, evaluator, data access, and proposal budget are held fixed, and only the tree-to-language representation varies. This control is particularly important in tabular LLM evaluation,

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

where benchmark memorization can confound apparent gains (Bordt et al., 2024). Related work also underscores the importance of strong tree baselines (Huertas, 2024) and leakage control in temporally structured feature engineering (Najafabadi et al., 2026).

We study two complementary reasoning tasks: *global rule generation*, which asks the language model to synthesize a small set of high-precision rules from the encoded tree, and *budgeted counterfactual search*, which asks it to find a sparse, in-domain edit that crosses the frozen model's decision boundary under a fixed candidate budget. We treat counterfactual search as model-level boundary search rather than actionable or causal recourse, which requires stronger feasibility and intervention assumptions (Ustun et al., 2019; Karimi et al., 2022; Wachter et al., 2017; Mothilal et al., 2020). Across five public tabular datasets and three language models, the static comparison shows that representation choice is consequential but highly regime-dependent: no single encoding is consistently best, and the spread between encodings can exceed sixty recall points on the same underlying model.

This motivates studying the representation as part of an iterative inference process. We first use a fixed-view loop that incorporates feedback from earlier proposals while keeping the tree representation unchanged. We then introduce DRALOOP, which additionally re-points the representation toward the current unresolved state at each round. Both the fitted model and the language model remain frozen. This design lets us measure the gains from iteration and from representation adaptation separately.

In rule generation, iteration provides substantial gains over one-shot reasoning, and DRALOOP exceeds the best one-shot fixed encoding on all five datasets for all three language models. In counterfactual search, DRALOOP adds to matched fixed-view iteration where unresolved headroom remains: its strongest configuration reaches 92.0% of the attainable validity bound, compared with 81.6% for the fixed-view loop. More broadly, these results suggest that when one learned system reasons over another, the representation connecting them can be a controllable and adaptive component of inference.

Our contributions are the following:

- **Representation-controlled formulation.** We formulate tree-to-language reasoning as a controlled representation problem, varying only the encoding while holding the fitted model, language model, parser, evaluator, data access, and budgets fixed.
- **Regime-dependent representation effects.** Across 5 datasets, 3 language models, and 2 reasoning tasks, we show that representation choice has a substantial but highly non-uniform effect, with no single encoding consistently best across settings.
- **DRALOOP: test-time representation adaptation.** We introduce a residual-adaptive inference procedure that re-points the representation toward unresolved state. It exceeds the best one-shot encoding on all five rule-generation datasets for all three language models, and closes additional residual headroom beyond matched iteration in counterfactual search.

2 RELATED WORK

Language models are already used in tabular settings, for prediction, feature construction, and tree-conditioned prompting (Hegselmann et al., 2023; Hollmann et al., 2023; Han et al., 2024; Abhyankar et al., 2026; Nam et al., 2024; Liu & Mazumder, 2025; Ye et al., 2025), but the neighboring literatures ask a different question. Row-serialization work such as TabLLM studies how tables or rows should be exposed to a language model for prediction (Hegselmann et al., 2023): representation matters there too, but the object represented is the data, not a learned intermediate model.

A second line uses language models for feature engineering or structured search, proposing features, rules, or transformations whose value is measured through downstream retraining or iterative search (Hollmann et al., 2023; Han et al., 2024; Abhyankar et al., 2026; Nam et al., 2024). These are close in spirit, OCTree especially, since it already uses tree-derived feedback, though its iterative gains arise in a setting where generated features and the downstream predictive model change across rounds. Closest is the tree-conditioned prompting literature: TreePrompt and DeLTa supply tree-derived intermediates in place of a raw row serialization (Liu & Mazumder, 2025; Ye et al., 2025). Our object of study differs. Rather than proposing one tree-derived textualization in service of a downstream predictor, we compare multiple representation families of the same fitted tree under a common evaluator and across reasoning tasks.

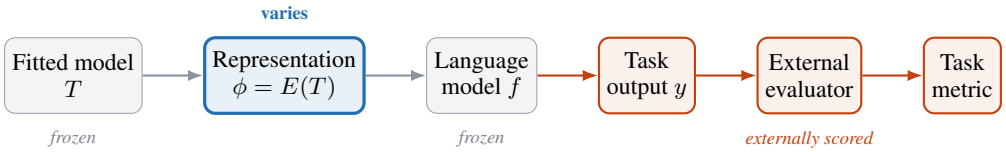

Figure 1: Representation-controlled evaluation. The fitted model T, language model f, parser, evaluator, and budgets are held constant while the representation $\phi = E(T)$ varies. The metrics are $R^*(\tau)$ for rule generation and, for counterfactual search, validity and edit cost.

Our first task, rule generation, asks for a small set of conjunctive rules with high coverage under a precision constraint, connecting directly to rule induction and rule-set learning. Sequential covering methods such as RIPPER and CN2 grow rules greedily from data (Cohen, 1995; Clark & Niblett, 1989); CORELS and Scalable Bayesian Rule Lists optimize rule lists (Angelino et al., 2018; Yang et al., 2017); PRIM searches for high-precision boxes (Friedman & Fisher, 1999); and RuleCOSI+ and ruleopt extract compact rule sets from fitted ensembles (Obregon & Jung, 2023; Röber et al., 2025). We score methods whose outputs match our unordered union-of-k objective: RIPPER and depth-three CART (Breiman et al., 1984) serve as candidate generators, while TE2Rules restricted to its first stage reads the fitted ensemble directly (Lal et al., 2022).

Counterfactual explanations and algorithmic recourse study minimal changes that reverse an unfavorable prediction, often with feasibility or actionability requirements (Ustun et al., 2019; Karimi et al., 2022; Wachter et al., 2017; Mothilal et al., 2020). Our task is narrower: budgeted counterfactual search asks for sparse, in-domain changes that cross a frozen model's decision boundary under a fixed candidate budget. Related tree-specific methods use path search, mixed-integer optimization, or differentiable approximations (Tolomei et al., 2017; Parmentier & Vidal, 2021; Lucic et al., 2022). We compare against model-agnostic methods that can be evaluated under the same sparsity, domain, and candidate-budget constraints: NICE (Brughmans et al., 2024), DiCE-genetic (Mothilal et al., 2020), FACE (Poyiadzi et al., 2020), and random feasible search.

Because the representations contain fitted-tree statistics, tree fitting, representation construction, feedback, candidate acceptance, and portfolio selection use only the generation window. The disjoint reporting window is used only for final evaluation.

3 Problem Formulation

In this section, we define the static representation-controlled problem studied in this paper. Section 5 extends this formulation to the adaptive setting. Let T denote a fitted tree or boosted-tree model and f a language model, both frozen throughout. The language model reasons about T through a textual encoding E, with $\phi = E(T)$ denoting the rendered representation supplied in the prompt. We compare a family of encodings $\{E_1, \ldots, E_M\}$ while holding the fitted model, language model, parser, evaluator, data access, and proposal budgets fixed. Downstream reasoning is $y = f(\phi, z)$, where z denotes the task input, and an external evaluator scores the resulting output. We evaluate this representation-controlled setup on two reasoning tasks that are described in Section 3.1. Figure 1 summarizes the setup.

3.1 Task Families

We study two reasoning tasks under the same representation-controlled setup: rule generation and budgeted counterfactual search that are detailed in this section.

Rule generation: Given a representation of T, the language model produces candidate conjunctive rules. The evaluator selects a set S of at most k rules subject to a minimum-support requirement and a union-precision floor τ. Let $p_D(S)$ and $r_D(S)$ denote union precision and union recall on data split D, crediting each covered positive once.

Candidate acceptance and portfolio selection are performed on the generation window:

Representation	Contents
`schema_only`	Task and field schema; no fitted-model information
`univariate_summary`	Per-feature class-conditional statistics
`leaf_statistics`	Tree hierarchy and leaf-level class counts
`node_statistics`	Tree hierarchy and class counts at leaves and internal nodes
`node_impurity`	Node statistics with impurity values
`failure_geometry`	Residual profiles, near misses, and high-depth regions
`rule_set`	Precision-qualified tree rules with coverage statistics
`path`	Target-class root-to-leaf paths and thresholds

Table 1: Tree representations evaluated in this work

$$S^*_\tau = \arg \max_{S:\, |S| \le k,\; p_{\mathrm{gen}}(S) \ge \tau} r_{\mathrm{gen}}(S),$$

subject also to the minimum-support requirement. The selected portfolio is then evaluated once on the sealed reporting window, reporting $r_{\mathrm{rep}}(S^*_\tau)$ together with $p_{\mathrm{rep}}(S^*_\tau)$. The precision floor applies to selection on the generation window. The reporting-window precision measures generalization and is not guaranteed to meet that floor.

Maximizing coverage subject to a precision floor is related to the high-precision rule objective used by Anchors (Ribeiro et al., 2018) and the operating point reported by RIFF (Martins et al., 2024). Our setting additionally imposes a fixed budget of at most k rules.

Budgeted counterfactual search: Given a representation of T, an input x, and a protocol-fixed target class $y^\star$, the language model proposes modified instances x'. Each candidate may change at most K mutable features within their allowed domains. The model proposes up to R candidates in each of Q rounds, giving a total candidate budget $B = RQ$, with every candidate applied independently to the original x.

Success for an input is

$$\mathrm{valid}_B(x) = \mathbb{1}[\exists j \le B : T(x \oplus \delta_j) = y^\star,\ |\mathrm{supp}(\delta_j)| \le K]\,.$$

Validity@B is the mean of $\mathrm{valid}_B(x)$ over evaluated inputs. For successful inputs, proximity is the normalized L_1 cost of the lowest-cost valid candidate (lower is better); sparsity is its number of changed fields; and feasibility requires all edits to satisfy the mutability and domain constraints. This task measures sparse boundary crossing for the frozen model under a fixed candidate budget rather than causal or actionable recourse.

4 Representation Families

We compare seven shared encodings and one counterfactual-specific path encoding. They expose the schema, marginal statistics, tree structure, error patterns, or extracted rules, as summarized in Table 1. These encodings deliberately span an information-exposure gradient rather than paraphrasing a single fixed content: `schema_only` carries no information about the fitted model and serves as an information-free floor, while `rule_set` and `path` expose precision-qualified structure extracted from T. We therefore treat "representation" as spanning both how the same fitted object is rendered and how much of it is exposed, and we read differences across encodings in that light.

The experiments below distinguish three inference regimes. In one-shot inference, the language model receives a single fixed view $\phi = E(T)$. In fixed-view iteration, the reasoning state changes across rounds while the rendered tree view remains fixed. In adaptive refinement, the view itself also depends on the current unresolved state. State-dependent adaptive refinement with DRALoop is described in Section 5.

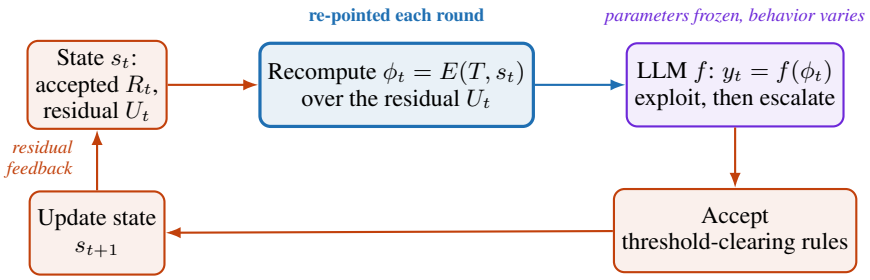

Figure 2: One DRALoop round. The representation is recomputed from the current residual s_t; accepted rules update the residual used in the next round. The fitted model and language model are not updated.

5 Adaptive Refinement: the DRALoop

DRALoop makes the representation depend on the current reasoning state. Let s_t denote the positives in the generation window not covered by the rules accepted through round t. At the next round, the representation is recomputed as $\phi_t = E(T, s_t)$, and the language model produces $y_t = f(\phi_t, s_t)$. The loop first selects from precision-qualified rules extracted from T; after a round with no additional accepted coverage, it requests new rules for the residual. A stalled round permits up to five redraws. Accepted rules update s_{t+1} and the next representation.

Three degrees of state-dependence are compared. A *non-adaptive* control runs the loop with ϕ held constant: the per-feature statistics are computed over all positives and the rendered rule set is the same at every round, so nothing the model sees depends on s_t. A *residual-updated* view recomputes the statistics over the uncovered positives alone. A *portfolio-aligned* view additionally re-orders the rendered rules by their marginal contribution to the rules already deployed, scoring each candidate by the positives it would add that the deployed set misses, less the false positives it would add, weighted by $\tau/(1-\tau)$.

The loop also needs a policy for when to stop reusing the tree's rules and start proposing new ones. Two are compared: a *scheduled* policy that switches after a fixed number of rounds, and a *model-chosen* policy in which the language model is shown the union precision and recall of the rules accepted so far and decides for itself, each round, whether to continue harvesting or to extend into the residual.

6 Experiments

6.1 Datasets

We evaluate on five public tabular datasets: Online Shoppers Purchasing Intention, UCI Bank Marketing, Adult (Census Income), UCI Default of Credit Card Clients, and Phishing Websites. We use canonical train-test partitions when available and otherwise construct a stratified outer split; Appendix D gives the split sizes. Fields that leak the target or encode sensitive attributes are withheld from the rule language.

6.2 Models

We evaluate three language models: Qwen 3 235B as the primary model, with GPT-OSS 120B and DeepSeek-R1 as cross-model comparisons. All three receive representations of the same fitted trees under identical task budgets and evaluators.

The underlying predictor is a LightGBM ensemble fitted with identical settings across datasets, conditions, and seeded runs. We use 100 boosting rounds, `max_depth` 6, `num_leaves` 31, `min_child_samples` 50, `learning_rate` 0.05, `feature_fraction` 0.8, `bagging_fraction` 0.8 with `bagging_freq` 1, binary objective, and model seed 42.

6.3 EXPERIMENTAL SETUP

The central experimental principle is to hold fixed the dataset split, fitted tree T, language model f, prompt constants, parser, external evaluator, and proposal budget while varying the representation exposed to the language model. In the static conditions this representation is $\phi = E(T)$; in the adaptive conditions it becomes state-dependent, $\phi_t = E(T, s_t)$, while T and f remain frozen.

For each dataset, we use its provided train-test partition when an official split is available; otherwise we construct a stratified split. We treat the training portion as the *generation window* and the disjoint test portion as the sealed *reporting window*. This yields dataset-specific split proportions rather than a common 75/25 split. The reporting window is never used for tree fitting, representation construction, search-time feedback, rule acceptance, or portfolio selection, and is opened only for final evaluation. Within the generation window, we form a balanced subset containing all positives and an equal-sized sample of negatives. We use 80% of this subset to fit the tree and 20% as an internal holdout for search-time statistics.

6.4 TASK SETUP, BASELINES, AND METRICS

Rule generation: Rule generation provides the main cross-model comparison. We use rule budget k=3 and precision floor τ=70. We compare with three non-LLM baselines: RIPPER (Cohen, 1995), a depth-three CART (Breiman et al., 1984), and TE2Rules restricted to its first stage (Lal et al., 2022). RIPPER and CART candidates pass through the same parser, precision filter, selector, and rule budget as the language-model conditions. TE2Rules candidates are extracted from the fitted ensemble and processed by the same filter and selector. We compare these references with one-shot encodings, an iterative fixed-view control, and the DRALOOP variants. Iterative conditions run for eight rounds, render 160 tree-derived rules, and request twenty proposals per call, with up to five redraws after a round produces no admissible rule. The main metric is reporting-window union recall. Union precision is reported alongside it, with values below the floor marked explicitly since the floor is enforced during generation-window selection and need not hold on unseen data. Appendix A reports threshold sweeps and per-model grids.

Budgeted counterfactual search: Counterfactual search uses the same five datasets and frozen fitted models. We compare the representation-based conditions against random feasible search, NICE (Brughmans et al., 2024), DiCE-genetic (Mothilal et al., 2020), and FACE (Poyiadzi et al., 2020); an exact threshold-cell MILP provides an attainable validity bound rather than a competing method. We fix at most K=2 edited fields, R=3 candidates per round, Q=4 rounds, and hence B=12 candidate evaluations per instance. The iterative boundary-search conditions use the `path` representation, with matched one-round and multi-round fixed-view controls to separate candidate breadth from sequential feedback. The target is a protocol-defined model class rather than a claim about a desirable real-world outcome. Each result reports validity and proximity: validity is the fraction of instances for which the frozen model reaches the target class within the candidate budget, while proximity is the mean normalized L_1 cost of the lowest-cost valid candidate. Sparsity is enforced as a hard constraint, and all edits must satisfy the declared mutability and domain constraints. Because proximity is defined only for successful instances, validity and proximity are interpreted jointly. Full sparsity, feasibility, budget, and cross-model results are reported in Appendix B.

6.5 RESULTS AND DISCUSSION

The main results are summarized in Tables 2 and 4, with full cross-model grids, threshold sweeps, and additional controls in the appendix. We organize the discussion around three questions: whether any fixed representation is consistently best, how much iteration contributes independently of representation adaptation, and when state-dependent re-pointing provides additional gains.

6.5.1 REPRESENTATION EFFECTS VARY ACROSS SETTINGS

No fixed representation is consistently best across all settings. In rule generation, the strongest encoding changes across datasets, and Table 3 shows that its identity also changes with the language model. The differences are substantial: on the same fitted model, changing only the representation can move held-out recall by tens of points. The non-LLM references show a similarly important baseline effect: CART and RIPPER generally produce high-precision but narrow rules under the fixed

Method	OS	Adult	Credit	Phish	BM
Non-LLM reference points					
CART depth 3 (Breiman et al., 1984)	32.6 (-34%), 82.2	23.2 (-53%), 96.6	14.3 (-56%), 75.8	82.5 (-0%), 95.6	13.1 (-27%), 73.4
RIPPER (Cohen, 1995)	41.8 (-15%), 74.3	25.2 (-49%), 76.8	11.6 (-64%), 62.2*	71.9 (-13%), 97.5	12.3 (-32%), 71.9
Tree baseline (Lal et al., 2022)	49.1, 71.6	49.4, 76.6	32.3, 68.5*	82.9, 90.3	**18.0**, 69.1*
Static representations: one fixed view, $\phi = E(T)$, asked once					
`schema_only`	42.1 (–14%), 72.1	18.9 (–62%), 65.7*	17.7 (–45%), 69.7*	22.7 (–73%), 28.8*	7.1 (–60%), 57.9*
`univariate_summary`	45.9 (–7%), 71.1	20.2 (–59%), 85.7	12.8 (–61%), 58.9*	62.9 (–24%), 76.7	10.9 (–40%), 61.9*
`leaf_statistics`	50.0 (+2%), 69.7*	28.5 (–42%), 77.7	16.5 (–49%), 70.5	90.5 (+9%), 83.2	10.5 (–42%), 59.3*
`node_statistics`	50.5 (+3%), 70.1	25.8 (–48%), 86.5	11.6 (–64%), 72.1	88.5 (+7%), 84.9	3.1 (–83%), 37.0*
`node_impurity`	48.2 (–2%), 70.0	25.8 (–48%), 82.6	11.2 (–65%), 57.0*	81.3 (–2%), 91.3	12.9 (–28%), 70.3
`failure_geometry`	48.6 (–1%), 70.8	25.2 (–49%), 62.8*	28.2 (–13%), 69.8*	78.5 (–5%), 84.6	9.9 (–45%), 73.4
`rule_set`	46.9 (–5%), 70.7	33.2 (–33%), 81.4	20.9 (–35%), 70.8	85.1 (+3%), 87.8	9.4 (–48%), 71.0
Iterative, view held fixed across rounds					
Fixed view (control)	54.7 (+11%), 69.3*	50.2 (+2%), 76.4	32.3 (–0%), 68.6*	93.6 (+13%), 79.4	**18.0** (–0%), 68.9*
DRALoop: view re-pointed at the residual, $\phi_t = E(T, s_t)$					
Residual-updated statistics	55.3 (+13%), 69.4*	51.8 (+5%), 74.0	32.2 (–0%), 68.8*	92.6 (+12%), 78.7	17.5 (–3%), 69.2*
+ scheduled exploration	51.8 (+6%), 69.9*	52.2 (+6%), 73.8	32.5 (+1%), 68.8*	**94.1** (+13%), 78.8	16.9 (–6%), 68.2*
Portfolio-aligned view	54.1 (+10%), 68.5*	51.8 (+5%), 74.2	32.6 (+1%), 68.6*	92.5 (+12%), 81.8	17.3 (–4%), 69.0*
+ model-chosen exploration	**55.8** (+14%), 68.8*	**54.7** (+11%), 73.5	**32.8** (+2%), 69.0*	93.3 (+13%), 81.0	**18.0** (–0%), 68.5*

Table 2: Rule generation with Qwen 3 235B at the deployment budget k=3 and precision floor τ=70. Each cell reports reporting-window union recall, relative change from the tree baseline in parentheses, and union precision after the comma. * marks reporting-window precision below the floor. (Note that it does not indicate a violation of the generation-window selection constraint.) Results are means over six seeded runs; column maxima are in **bold**.

Best-encoding $R^*(\tau$=70)	OS	BM	Adult	Credit	Phish
Tree heuristic baseline	49.10	18.00	49.40	32.30	82.90
Qwen 3 235B (best rep.)	50.50	12.90	33.20	28.20	90.50
GPT-OSS 120B (best rep.)	51.50	15.40	27.30	22.80	91.20
DeepSeek-R1 (best rep.)	51.00	17.10	29.60	23.50	92.00

Table 3: Model-capability view: the best-encoding ceiling $\max_i R^*(\tau$=70) over the seven representations for each (model, dataset) cell, with the tree-heuristic baseline for reference (%), at k=3 over six seeded runs sharing the same sealed reporting window. The argmax encoding varies by cell, and by model as well as by dataset, so the ceiling is not attributable to one encoding. Per-model grids in Appendix A.

k=3 budget, while direct access to rules from the fitted ensemble provides a substantially stronger reference on several datasets.

The ordering of the static encodings also changes across tasks (Table 4). Thus, an encoding's performance depends on the task, dataset, fitted model, and language model. Phishing Websites is a partial exception: a structure-based encoding (`leaf_statistics`, `node_statistics`, or `node_impurity`) is the strongest one-shot view there for all three models, reflecting that dataset's strong, well-separated tree baseline rather than contradicting the broader instability.

6.5.2 Iteration Helps, and Adaptive Refinement Adds Beyond Iteration

The results separate two effects: iteration provides a substantial gain on its own, and adaptive refinement can provide additional gains beyond matched iteration. In rule generation, repeatedly reasoning over a fixed view improves over the best one-shot representation on all five datasets for Qwen 3 235B. This makes the fixed-view iterative condition the appropriate control for DRALoop. Against that control, state-dependent re-pointing provides additional gains in several settings.

The same decomposition is clearer in counterfactual search (Table 12). Iterative reasoning over a fixed representation improves validity on every dataset. Re-pointing the view provides further gains over this matched control, most notably from 58.3% to 87.5% on Bank Marketing and from 53.3% to 65.0% on Credit Default. Pooled and paired within run, the strongest refined view gains 9.2 percentage points, while residual-updated statistics alone gains only 0.5 points. The pooled mean carries a large standard deviation ($\pm$15.7; Table 12) because the effect is bimodal rather than noisy: it is concentrated on the two datasets that retain headroom (Bank Marketing and Credit Default) and is

Method	OS	Adult	Credit	Phish	BM	% of bound
Exact bound. A property of the problem under the declared action domain, not a competing method.						
Threshold-cell MILP *(exact bound)*	100.0±0.0, 0.02	100.0±0.0, 0.73	85.8±8.6, 0.56	100.0±0.0, 1.27	100.0±0.0, 0.78	
Model-agnostic references: frozen-model scores and training rows only						
Random feasible search	59.2±24.0, 0.52	21.7±9.3, 1.46	26.7±14.4, 1.59	10.8±5.8, 2.17	82.5±16.4, 2.80	41.1
NICE-Q12 (Brughmans et al., 2024)	83.3±5.2, 0.66	94.2±3.8, 1.20	20.8±7.4, 0.06	84.2±4.9, 2.60	24.2±9.7, 0.93	62.0
DiCE-genetic (Mothilal et al., 2020)	50.0±7.7, 6.00	75.0±8.4, 5.04	43.3±10.3, 4.93	29.2±6.6, 2.79	36.7±6.8, 2.05	48.3
FACE (Poyiadzi et al., 2020)	67.5±12.5, 0.41	89.2±8.6, 1.59	23.3±6.1, 0.20	73.3±4.1, 3.03	22.5±10.4, 1.08	55.9
Static representations: one fixed view, $\phi = E(T)$, reused across rounds						
`schema_only`	**100.0±0.0**, 2.25	**100.0±0.0**, 5.42	57.5±9.9, 3.17	5.0±4.5, 5.67	49.2±13.9, 4.50	64.2
`univariate_summary`	**100.0±0.0**, 2.00	**100.0±0.0**, 3.03	55.0±13.8, 2.97	65.0±16.1, 4.16	69.2±19.9, 2.31	79.7
`leaf_statistics`	99.2±2.0, 1.06	**100.0±0.0**, 1.65	50.0±12.6, 2.80	50.8±8.0, 3.05	77.5±19.7, 3.56	77.2
`node_statistics`	98.3±4.1, **0.73**	**100.0±0.0**, **1.54**	46.7±8.8, 3.30	50.0±8.4, 3.38	83.3±14.0, 4.11	77.2
`node_impurity`	99.2±2.0, 0.81	**100.0±0.0**, 1.60	50.0±10.0, 2.02	51.7±15.7, 3.28	76.7±14.0, 3.95	77.2
`failure_geometry`	**100.0±0.0**, 2.06	93.3±4.1, 2.92	51.7±12.9, 2.36	82.5±12.9, 3.97	75.8±9.7, 6.28	82.4
`rule_set`	**100.0±0.0**, 1.96	96.7±4.1, 2.22	57.5±14.4, 2.14	85.0±11.8, 4.24	77.5±11.7, 3.98	85.2
Budget-allocation controls: boundary view held fixed						
Fixed view (R=12, Q=1)	93.3±10.3, 1.46	70.0±8.4, 1.84	25.8±10.7, 1.76	55.8±14.6, 3.79	30.0±17.9, 2.39	55.8
Fixed view (R=3, Q=4)	**100.0±0.0**, 1.56	98.3±2.6, 2.12	53.3±11.7, 1.90	89.2±15.3, 4.08	58.3±11.3, 2.65	81.6
DRALoop: view updated or re-pointed at the unsolved residual						
Residual-updated statistics	**100.0±0.0**, 1.59	97.5±2.7, 2.55	55.0±11.4, 3.24	90.0±10.0, 4.15	59.2±20.6, 2.62	82.2
+ scheduled exploration	**100.0±0.0**, 1.59	99.2±2.0, 2.75	60.0±11.8, 2.48	95.8±5.8, 4.14	60.0±17.6, 2.61	85.0
Portfolio-aligned view	**100.0±0.0**, 1.52	98.3±2.6, 1.82	**65.0±14.1**, 1.97	95.0±6.3, 4.10	86.7±8.2, 2.55	**91.2**
+ model-chosen exploration	**100.0±0.0**, 1.58	99.2±2.0, 1.77	**65.0±10.0**, 2.23	**97.5±2.7**, 4.27	**87.5±12.1**, 2.71	**92.0**

Table 4: Budgeted counterfactual search at K=2 with Qwen 3 235B. Each cell reports [validity, proximity] over six seeded runs with N=20 instances per run. The final column reports mean validity as a percentage of the exact bound across datasets. Bold marks column maxima within the language-model blocks

near zero on the datasets already at saturation. We therefore read the per-dataset decomposition, not the pooled mean, as the primary evidence.

6.5.3 When does adaptation help?

The gains from adaptive refinement are not uniform. They are largest where fixed-view iteration leaves substantial unresolved headroom: Adult in rule generation, and Bank Marketing and Credit Default in counterfactual search. In contrast, refinement adds little when the fixed-view condition is already near saturation. Figure 3 shows this pattern across rounds. At K=1, where the attainable bounds leave more room for separation, the strongest refined configuration reaches 81.6% of the bound, compared with 58.9% for the strongest model-agnostic reference; at K=2 the corresponding values are 92.0% and 62.0%.

7 Conclusion

We isolate the effect of tree-to-language representation while holding the fitted model, language model, and evaluation pipeline fixed. Across two tasks, five datasets, and three language models, no fixed encoding is consistently best. DRALoop updates the representation from the unresolved state at each round. In rule generation, it exceeds the best one-shot representation on all five datasets for Qwen 3 235B, by up to 21.5 recall points, although its gains over a matched iterative fixed-view control are smaller and vary by dataset. In counterfactual search, adaptation provides its largest gains over that control on Bank Marketing and Credit Default. The strongest configuration reaches 92% of the attainable validity bound, compared with 62% for the strongest model-agnostic baseline.

Future work will examine whether the observed representation rankings and the value of residual adaptation transfer to other tree learners (CART, random forests, XGBoost) and to neural models that admit tree-like explanations.

Ethics Statement

The counterfactual task measures changes that alter a model prediction; it does not assess causal, actionable, or advisable interventions. This distinction matters for the credit, income, and marketing datasets. The Bank Marketing action domain is permissive: only `age` is immutable, so economic

indicators remain editable. This partly explains the strength of random search on that dataset. The datasets are public benchmarks and we use only their published fields.

REPRODUCIBILITY STATEMENT

Sections 3 to 6 and the appendices specify the representations, task protocols, dataset partitions, model configuration, metrics, baselines, prompts, and inference budgets. Tree fitting, representation construction, feedback, candidate acceptance, and portfolio selection use the generation window; the reporting window is used only for final evaluation.

The code artifact includes dataset adapters, encodings, the rule parser and selector, both task runners, baselines, and table-generation scripts. Each run records the data partition, seed, fitted-model configuration, and evaluation protocol. Model outputs are not deterministic, so comparisons are averaged over six runs and paired within run.

The code artifact is currently under institutional release review and, subject to completion of that review, will be released anonymously as supplementary material during the public ICLR review period, together with pinned dependencies and reproduction instructions.

AI USE STATEMENT

Qwen 3 235B, GPT-OSS 120B, and DeepSeek-R1 are evaluated as frozen reasoners. A general-purpose coding assistant was used to help implement analysis and scoring scripts, run experiments, and edit the manuscript. The authors reviewed the resulting code, text, and quantitative outputs.

REFERENCES

Nikhil Abhyankar, Parshin Shojaee, and Chandan K. Reddy. LLM-FE: Automated feature engineering for tabular data with LLMs as evolutionary optimizers. *Transactions on Machine Learning Research*, 2026. Accepted; arXiv:2503.14434.

Elaine Angelino, Nicholas Larus-Stone, Daniel Alabi, Margo Seltzer, and Cynthia Rudin. Learning certifiably optimal rule lists for categorical data. *Journal of Machine Learning Research*, 18(234): 1–78, 2018.

Sebastian Bordt, Harsha Nori, Vanessa Rodrigues, Besmira Nushi, and Rich Caruana. Elephants never forget: Memorization and learning of tabular data in large language models. *arXiv preprint arXiv:2404.06209*, 2024.

Leo Breiman, Jerome H. Friedman, Richard A. Olshen, and Charles J. Stone. *Classification and Regression Trees*. Wadsworth, 1984.

Dieter Brughmans, Pieter Leyman, and David Martens. Nice: an algorithm for nearest instance counterfactual explanations. *Data Mining and Knowledge Discovery*, 38:266–304, 2024.

Peter Clark and Tim Niblett. The CN2 induction algorithm. *Machine Learning*, 3(4):261–283, 1989.

William W. Cohen. Fast effective rule induction. In *Proceedings of the Twelfth International Conference on Machine Learning*, pp. 115–123, 1995.

Jerome H. Friedman and Nicholas I. Fisher. Bump hunting in high-dimensional data. *Statistics and Computing*, 9(2):123–143, 1999.

Sungwon Han, Jinsung Yoon, Sercan Ö. Arik, and Tomas Pfister. Large language models can automatically engineer features for few-shot tabular learning. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 17454–17479, 2024.

Stefan Hegselmann, Alejandro Buendia, Hunter Lang, Monica Agrawal, Xiaoyi Jiang, and David Sontag. Tabllm: Few-shot classification of tabular data with large language models. In *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*, volume 206 of *Proceedings of Machine Learning Research*, pp. 5549–5581, 2023.

Noah Hollmann, Samuel Müller, and Frank Hutter. Large language models for automated data science: Introducing CAAFE for context-aware automated feature engineering. In *Advances in Neural Information Processing Systems*, 2023.

Carlos Huertas. Gradient boosting trees and large language models for tabular data few-shot learning. *arXiv preprint arXiv:2411.04324*, 2024.

Amir-Hossein Karimi, Gilles Barthe, Bernhard Schölkopf, and Isabel Valera. A survey of algorithmic recourse: Contrastive explanations and consequential recommendations. *ACM Computing Surveys*, 2022.

G. Roshan Lal, Xiaotong Chen, and Varun Mithal. TE2Rules: Explaining tree ensembles using rules. *arXiv preprint arXiv:2206.14359*, 2022.

Brian Liu and Rahul Mazumder. Treeprompt: Distilling boosted tree ensembles for in-context learning in LLMs. OpenReview / KDD 2025 Workshop on SKnow-LLM, 2025. OpenReview id: 0ZALcXRKAc.

Ana Lucic, Hinda Haned, and Maarten de Rijke. FOCUS: Flexible optimizable counterfactual explanations for tree ensembles. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 5313–5322, 2022.

João Martins, João Bravo, Ana Paula Gomes, Carlos Soares, and Pedro Bizarro. RIFF: Inducing rules for fraud detection from decision trees. In *Rules and Reasoning (RuleML+RR)*, 2024. doi: 10.1007/978-3-031-72407-7_5.

Ramaravind K. Mothilal, Amit Sharma, and Chenhao Tan. Explaining machine learning classifiers through diverse counterfactual explanations. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency (FAT*)*, 2020.

Maryam Khanian Najafabadi, Bushra Naeem, Touraj Khodadadi, Saman Shojae Chaeikar, and Zawar Shah. LLM-guided automated feature engineering for time series data with temporal leakage control. *AI*, 7(7):245, 2026. doi: 10.3390/ai7070245.

Jaehyun Nam, Kyuyoung Kim, Seunghyuk Oh, Jihoon Tack, Jaehyung Kim, and Jinwoo Shin. Optimized feature generation for tabular data via LLMs with decision tree reasoning. In *Advances in Neural Information Processing Systems*, 2024.

Josue Obregon and Jae-Yoon Jung. RuleCOSI+: Rule extraction for interpreting classification tree ensembles. *Information Fusion*, 89:355–381, 2023.

Axel Parmentier and Thibaut Vidal. Optimal counterfactual explanations in tree ensembles. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 8422–8431, 2021.

Rafael Poyiadzi, Kacper Sokol, Raul Santos-Rodriguez, Tijl De Bie, and Peter Flach. FACE: Feasible and actionable counterfactual explanations. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society (AIES)*, pp. 344–350, 2020.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Anchors: High-precision model-agnostic explanations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018. doi: 10.1609/aaai.v32i1.11491.

Tabea Röber, Kanak Lumadjeng, M. Hakan Akyüz, and Ş. İlker Birbil. Rule generation for classification: Scalability, interpretability, and fairness. *Computers and Operations Research*, 183:107163, 2025.

Gabriele Tolomei, Fabrizio Silvestri, Andrew Haines, and Mounia Lalmas. Interpretable predictions of tree-based ensembles via actionable feature tweaking. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 465–474, 2017.

Berk Ustun, Alexander Spangher, and Yang Liu. Actionable recourse in linear classification. In *Proceedings of the Conference on Fairness, Accountability, and Transparency (FAT*)*, 2019.

Condition	OS				BM				Adult				Credit				Phish			
	τ60	τ70	τ80	τ90	τ60	τ70	τ80	τ90	τ60	τ70	τ80	τ90	τ60	τ70	τ80	τ90	τ60	τ70	τ80	τ90
Tree baseline	55.10	49.10	41.00	25.40	22.70	18.00	10.10	0.60	58.40	49.40	32.60	25.80	37.10	32.30	10.80	0.00	84.40	82.90	81.90	76.30
`univariate_summary`	47.20	45.90	21.50	3.90	17.80	10.90	1.10	0.00	12.70	20.20	19.80	14.40	14.40	12.80	0.00	0.00	90.50	62.90	77.00	67.30
`schema_only`	55.40	42.10	15.50	1.80	16.00	7.10	0.80	0.00	30.60	18.90	17.40	12.90	20.50	17.70	1.20	0.00	37.90	22.70	20.00	10.50
`leaf_statistics`	61.30	50.00	24.40	5.60	17.20	10.50	0.20	0.00	30.90	28.50	22.90	18.10	27.80	16.50	0.20	0.00	92.90	90.50	87.10	76.30
`node_statistics`	65.10	50.50	24.50	5.50	19.40	3.10	0.20	0.00	30.50	25.80	20.80	18.70	27.00	11.60	0.00	0.00	83.20	88.50	85.80	69.60
`node_impurity`	54.10	48.20	30.50	6.60	15.00	12.90	0.00	0.00	31.80	25.80	16.80	16.50	27.90	11.20	0.40	0.00	91.90	81.30	87.70	81.90
`failure_geometry`	54.00	48.60	34.10	9.40	19.90	9.90	2.00	0.00	31.80	25.20	27.00	13.70	22.30	28.20	4.80	0.30	49.10	78.50	64.10	67.80
`rule_set`	48.10	46.90	32.60	10.70	17.90	9.40	1.30	0.00	48.60	33.20	25.00	18.80	29.20	20.90	0.20	0.00	82.40	85.10	82.70	82.80

Table 5: Rule-generation results for Qwen 3 235B: threshold-constrained recall $R^*(\tau)$ at $\tau \in \{60, 70, 80, 90\}$ (%).

Method	OS	Adult	Credit	Phish	BM
Tree baseline	49.1, 71.6	49.4, 76.6	32.3, 68.5*	**82.9**, 90.3	18.0, 69.1*
Static representations: one fixed view, $\phi = E(T)$, asked once					
`schema_only`	30.0 (–39%), 79.8	16.9 (–66%), 79.2	14.1 (–56%), 57.8*	35.7 (–57%), 43.9*	3.3 (–82%), 47.7*
`univariate_summary`	26.8 (–45%), 79.0	16.5 (–67%), 89.0	19.1 (–41%), 73.2	77.4 (–7%), 85.1	9.3 (–48%), 72.9
`leaf_statistics`	46.8 (–5%), 72.3	23.4 (–53%), 81.8	17.1 (–47%), 71.1	84.8 (+2%), 87.5	13.3 (–26%), 72.4
`node_statistics`	50.4 (+3%), 69.8*	27.3 (–45%), 80.7	14.3 (–56%), 70.0	90.9 (+10%), 82.0	15.4 (–14%), 71.7
`node_impurity`	51.5 (+5%), 69.4*	26.0 (–47%), 81.9	12.8 (–60%), 58.1*	91.2 (+10%), 81.5	13.3 (–26%), 73.1
`failure_geometry`	38.9 (–21%), 76.8	21.4 (–57%), 77.9	22.8 (–29%), 71.1	74.9 (–10%), 89.0	14.4 (–20%), 71.2
`rule_set`	40.1 (–18%), 76.2	22.6 (–54%), 81.1	22.3 (–31%), 70.0	76.9 (–7%), 90.4	11.1 (–38%), 74.5
Iterative, view held fixed across rounds					
Fixed view (control)	53.1 (+8%), 69.6*	51.4 (+4%), 74.6	31.5 (–2%), 69.1*	93.4 (+13%), 79.6	18.4 (+2%), 68.4*
DRALoop: view re-pointed at the residual, $\phi_t = E(T, s_t)$					
Residual-updated statistics	53.7 (+9%), 69.6*	52.3 (+6%), 73.6	32.9 (+2%), 68.4*	92.3 (+11%), 84.2	18.4 (+2%), 69.0*
+ scheduled exploration	51.3 (+4%), 70.0	52.0 (+5%), 74.3	33.4 (+3%), 68.6*	92.8 (+12%), 82.3	18.4 (+2%), 68.0*
Portfolio-aligned view	**55.0** (+12%), 68.5*	51.6 (+4%), 74.9	**33.7** (+4%), 67.8*	**93.8** (+13%), 80.0	18.4 (+2%), 68.2*
+ model-chosen exploration	52.7 (+7%), 69.8*	**52.6** (+6%), 74.7	33.6 (+4%), 68.3*	92.2 (+11%), 83.7	**19.1** (+6%), 68.0*

Table 6: Rule generation with GPT-OSS 120B at the deployment budget k=3 with precision floor τ=70. Each cell reports reporting-window union recall, relative change from the tree baseline in parentheses, and union precision after the comma. * marks precision below the floor. Results are means over six seeded runs; column maxima are in **bold**.

Sandra Wachter, Brent Mittelstadt, and Chris Russell. Counterfactual explanations without opening the black box: Automated decisions and the GDPR. *Harvard Journal of Law & Technology*, 31, 2017.

Hongyu Yang, Cynthia Rudin, and Margo Seltzer. Scalable bayesian rule lists. In *Proceedings of the 34th International Conference on Machine Learning*, 2017.

Hangting Ye, Jinmeng Li, He Zhao, Dandan Guo, and Yi Chang. LLM meeting decision trees on tabular data. In *Advances in Neural Information Processing Systems*, 2025.

A Full rule-generation grids

The main text reports $R^*(\tau{=}70)$ at the deployment budget k=3 for the primary model (Qwen 3 235B). Tables 5–9 give the complete grids: the threshold-constrained recall $R^*(\tau)$ at $\tau \in \{60, 70, 80, 90\}$ for each of the three models (Qwen 3 235B, GPT-OSS 120B, DeepSeek-R1), and reporting-window union recall paired with union precision at k=3 for GPT-OSS 120B and DeepSeek-R1, in both cases for the tree baseline and all seven representations across the five datasets. Union precision for the primary model is in Table 2.

B Budgeted counterfactual search: full metrics

Table 10 adds sparsity to the validity and proximity results in the main text. Feasibility and domain constraints are enforced before a candidate can count as valid.

B.1 Matched grids at the other edit budgets

Table 11 reports the same conditions at K=1, alongside the K=2 grid of Table 4 in the main text. Every row in each table is a run executed at that budget, so no value is re-scored from a different one.

Condition	OS				BM				Adult				Credit				Phish			
	τ60	τ70	τ80	τ90	τ60	τ70	τ80	τ90	τ60	τ70	τ80	τ90	τ60	τ70	τ80	τ90	τ60	τ70	τ80	τ90
Tree baseline	55.10	49.10	41.00	25.40	22.70	18.00	10.10	0.60	58.40	49.40	32.60	25.80	37.10	32.30	10.80	0.00	84.40	82.90	81.90	76.30
`univariate_summary`	26.70	26.80	22.00	4.50	19.70	9.30	0.70	0.00	22.40	16.50	18.20	11.10	30.20	19.10	0.00	0.00	57.70	77.40	54.20	42.50
`schema_only`	47.20	30.00	16.90	5.60	14.70	3.30	0.00	0.00	7.40	16.90	10.00	7.50	22.30	14.10	0.00	0.00	63.80	35.70	32.60	3.70
`leaf_statistics`	63.90	46.80	26.10	10.80	18.50	13.30	1.30	0.00	24.30	23.40	18.60	17.40	31.40	17.10	0.20	0.00	94.50	84.80	85.40	87.10
`node_statistics`	60.90	50.40	34.80	16.30	17.90	15.40	0.30	0.00	32.00	27.30	18.80	15.30	29.80	14.30	0.00	0.00	84.00	90.90	88.20	86.90
`node_impurity`	60.90	51.50	35.40	17.80	18.50	13.30	0.90	0.00	30.00	26.00	15.00	13.20	28.90	12.80	0.00	0.00	90.90	91.20	85.30	87.40
`failure_geometry`	42.00	38.90	32.60	20.80	17.90	14.40	5.90	0.00	22.20	21.40	15.80	17.10	31.00	22.80	3.60	0.00	85.10	74.90	71.50	60.40
`rule_set`	37.20	40.10	33.90	15.40	14.70	11.10	0.30	0.00	22.60	22.60	18.50	17.10	29.50	22.30	1.60	0.00	73.40	76.90	73.10	66.70

Table 7: Rule-generation results for GPT-OSS 120B: threshold-constrained recall $R^*(\tau)$ at $\tau \in \{60, 70, 80, 90\}$ (%).

Method	OS	Adult	Credit	Phish	BM
Tree baseline	49.1, 71.6	49.4, 76.6	32.3, 68.5*	82.9, 90.3	18.0, 69.1*
Static representations: one fixed view, $\phi = E(T)$, asked once					
`schema_only`	37.0 (–25%), 74.5	25.3 (–49%), 80.4	23.5 (–27%), 71.0	28.9 (–65%), 29.4*	5.8 (–68%), 72.3
`univariate_summary`	17.4 (–65%), 22.6*	19.5 (–61%), 72.4	10.5 (–67%), 36.1*	65.5 (–21%), 74.1	13.1 (–27%), 72.5
`leaf_statistics`	49.5 (+1%), 70.0	26.0 (–47%), 81.2	20.6 (–36%), 68.8*	90.8 (+10%), 83.5	15.1 (–16%), 71.0
`node_statistics`	49.7 (+1%), 69.1*	27.9 (–44%), 83.7	22.2 (–31%), 70.9	92.0 (+11%), 77.4	17.1 (–5%), 69.9*
`node_impurity`	45.7 (–7%), 71.6	29.6 (–40%), 82.6	21.6 (–33%), 72.4	86.5 (+4%), 86.4	16.3 (–9%), 71.6
`failure_geometry`	51.0 (+4%), 71.2	27.9 (–44%), 79.9	22.7 (–30%), 70.8	83.9 (+1%), 85.1	12.2 (–32%), 75.0
`rule_set`	45.4 (–8%), 71.7	26.0 (–47%), 77.7	19.5 (–40%), 71.6	88.2 (+6%), 82.5	11.0 (–39%), 73.1
Iterative, view held fixed across rounds					
Fixed view (control)	52.1 (+6%), 68.3*	50.6 (+3%), 76.3	33.7 (+4%), 68.5*	93.6 (+13%), 81.9	18.2 (+1%), 68.2*
DRALoop: view re-pointed at the residual, $\phi_t = E(T, s_t)$					
Residual-updated statistics	**55.3** (+13%), 68.1*	**54.0** (+9%), 72.8	**34.2** (+6%), 68.4*	92.5 (+12%), 83.6	18.6 (+3%), 68.3*
+ scheduled exploration	53.9 (+10%), 68.9*	51.2 (+4%), 74.6	32.3 (–0%), 69.0*	94.2 (+14%), 79.8	18.4 (+2%), 68.3*
Portfolio-aligned view	54.8 (+12%), 68.6*	52.2 (+6%), 74.7	33.6 (+4%), 67.7*	**94.7** (+14%), 75.4	18.4 (+2%), 68.1*
+ model-chosen exploration	54.3 (+11%), 67.9*	51.4 (+4%), 75.3	33.8 (+5%), 68.3*	94.4 (+14%), 77.6	**18.9** (+5%), 67.3*

Table 8: Rule generation with DeepSeek-R1 at the deployment budget k=3 with precision floor τ=70. Each cell reports reporting-window union recall, relative change from the tree baseline in parentheses, and union precision after the comma. * marks precision below the floor. Results are means over six seeded runs; column maxima are in **bold**.

Mean sparsity is omitted at K=1 because the constraint pins it: every condition on every dataset averages exactly 1.00 changed fields.

Reading Table 11 and Table 4 together shows why the reported budget matters. The exact bound rises from 61.7 to 85.8 on Credit Default and from 60.0 to 100.0 on Phishing, so by K=2 four of the five bounds are at 100 and validity alone can no longer separate encodings on those columns. The distance to the bound is therefore most informative at K=1: there the strongest language-model configuration recovers 81.6% of the bound against 58.9% for the best model-agnostic reference, whereas at K=2 the same comparison compresses to 92.0% against 62.0% and preserves the ordering of mean validity exactly.

B.2 The effect across edit budgets and across models

Table 12 states the re-pointing effect as a paired within-partition difference against the loop's own fixed-view control at each of the three budgets, each from a separately executed run. Residual-updated statistics, which reports the residual while leaving the rendered view unchanged, is the control for prompt growth. The effect declines as the budget loosens because the metric saturates, and it is concentrated on the datasets that retain headroom: at K=2 the gain is 28.3 points on Bank Marketing and 11.7 on Credit Default, and exactly zero on Online Shoppers and Adult, where both arms already attain 100.0 and 98.3.

Table 13 repeats the paired comparison for all three models at K=1. Each model is differenced against its own fixed-view control, so the three rows are independent replications rather than a comparison of model quality. The second column is the check on prompt length: that condition reports the same information about what remains unsolved, so its prompt grows by the same amount, but the rendered view is left unchanged. Re-pointing the view gains between 8.2 and 10.8 points in all three models, whereas reporting the residual without re-pointing gains at most 1.0 and is negative for GPT-OSS 120B. A longer prompt therefore does not account for the effect. Four DeepSeek-R1 cells, on Phishing Websites and Credit Default, contained between one and three failed model calls

Condition	OS				BM				Adult				Credit				Phish			
	τ60	τ70	τ80	τ90	τ60	τ70	τ80	τ90	τ60	τ70	τ80	τ90	τ60	τ70	τ80	τ90	τ60	τ70	τ80	τ90
Tree baseline	55.10	49.10	41.00	25.40	22.70	18.00	10.10	0.60	58.40	49.40	32.60	25.80	37.10	32.30	10.80	0.00	84.40	82.90	81.90	76.30
`univariate_summary`	28.40	17.40	13.70	3.60	15.90	13.10	1.60	0.00	23.20	19.50	18.00	12.40	13.40	10.50	0.50	0.00	81.00	65.50	49.50	34.90
`schema_only`	32.80	37.00	17.20	5.80	15.10	5.80	0.80	0.00	28.10	25.30	21.90	11.90	35.70	23.50	0.30	0.00	39.00	28.90	21.20	0.00
`leaf_statistics`	53.20	49.50	31.80	8.70	18.70	15.10	2.60	0.00	36.90	26.00	20.30	15.30	27.20	20.60	0.60	0.00	95.00	90.80	74.30	88.50
`node_statistics`	41.80	49.70	30.80	6.10	19.50	17.10	0.30	0.00	27.50	27.90	21.20	16.50	26.20	22.20	1.20	0.00	87.60	92.00	84.30	86.50
`node_impurity`	48.40	45.70	26.90	5.90	18.70	16.30	2.10	0.00	27.80	29.60	15.50	17.00	27.00	21.60	1.20	0.00	92.60	86.50	88.30	86.70
`failure_geometry`	56.40	51.00	34.30	12.20	17.10	12.20	3.80	0.00	42.00	27.90	23.80	15.70	33.90	22.70	2.20	0.00	87.20	83.90	77.00	64.20
`rule_set`	49.60	45.40	35.80	11.30	17.30	11.00	2.20	0.00	37.80	26.00	16.10	11.30	29.80	19.50	0.40	0.00	61.70	88.20	79.00	55.20

Table 9: Rule-generation results for DeepSeek-R1: threshold-constrained recall $R^*(\tau)$ at $\tau \in \{60, 70, 80, 90\}$ (%).

Method	OS	Adult	Credit	Phish	BM
Non-LLM reference points					
Random feasible search	69.2, 0.73, 2.22	23.3, 2.34, 2.30	34.2, 1.60, 2.08	15.8, 3.68, 2.57	84.2, 3.12, 2.30
NICE-Q12 (budgeted adaptation)	76.7, 0.77, 1.76	98.3, 1.27, 1.40	19.2, 0.08, 1.58	98.3, 2.97, 1.77	26.7, 1.36, 2.03
Static representations					
`schema_only`	100.0, 2.23, 1.85	100.0, 4.31, 1.98	65.0, 5.08, 2.32	4.2, 8.13, 3.00	61.7, 4.88, 2.46
`univariate_summary`	100.0, 2.63, 2.28	100.0, 2.57, 2.04	61.7, 3.60, 2.40	51.7, 6.15, 2.67	90.8, 3.01, 2.68
`leaf_statistics`	100.0, 1.01, 1.37	100.0, 1.94, 1.50	62.5, 2.57, 1.96	58.3, 3.76, 1.73	97.5, 4.27, 2.34
`node_statistics`	96.7, 0.93, 1.28	100.0, 1.93, 1.47	56.7, 1.92, 1.76	55.8, 3.27, 1.56	97.5, 4.62, 2.22
`node_impurity`	98.3, 0.83, 1.33	100.0, 1.85, 1.45	59.2, 2.16, 1.81	40.0, 3.40, 1.60	98.3, 4.30, 2.39
`failure_geometry`	100.0, 2.40, 1.85	99.2, 3.23, 2.22	61.7, 2.75, 2.30	90.0, 5.64, 2.55	95.0, 7.09, 2.88
`rule_set`	100.0, 2.40, 2.21	100.0, 2.36, 2.11	71.7, 2.22, 2.58	96.7, 5.98, 2.74	96.7, 5.59, 2.70
Iterative, boundary view held fixed across rounds					
Fixed view (control)	100.0, 1.93, 2.16	100.0, 2.13, 1.83	62.5, 2.67, 2.35	99.2, 5.81, 2.75	77.5, 3.36, 2.44
DRALoop					
Residual-updated statistics	100.0, 1.93, 2.16	100.0, 2.04, 1.83	65.8, 2.66, 2.40	100.0, 5.83, 2.75	75.0, 3.38, 2.51
+ scheduled exploration	100.0, 1.93, 2.16	100.0, 2.11, 1.82	70.8, 2.95, 2.40	98.3, 5.80, 2.74	84.2, 3.52, 2.56
Portfolio-aligned view	100.0, 1.93, 2.16	100.0, 1.99, 1.80	69.2, 2.14, 1.97	99.2, 5.65, 2.68	94.2, 3.07, 2.23
+ model-chosen exploration	100.0, 1.93, 2.16	100.0, 1.99, 1.80	82.5, 3.09, 2.38	98.3, 5.82, 2.75	99.2, 3.37, 2.46

Table 10: Full budgeted-counterfactual-search grid (Qwen 3 235B). Each cell reports mean validity, proximity and sparsity, in that order, over six seeded runs. All budget-matched methods use K=3, B=12, and N=20 instances per run; the LLM rows allocate this budget as R=3, Q=4.

among twenty instances; a failed call is scored as an unsuccessful attempt rather than dropped, so it counts against that model's validity.

B.3 Provenance of the exact bound

An optimal formulation for counterfactuals in tree ensembles is established by Parmentier & Vidal (2021). We verified our threshold-cell program against their reference implementation on a common ensemble: the two agree on the optimal number of changed fields for eleven of twelve instances, and on no instance does the reference attain a sparser counterfactual. We report our own solver because the reference implementation could not be applied to this benchmark. Its optimal-sparsity objective is available only through a commercial solver whose licensed problem size our ensembles exceed by roughly an order of magnitude, and its interface does not express the immutable-field constraint this protocol enforces.

Two of the model-agnostic references lose validity under a hard sparsity budget for a shared and predictable reason. DiCE draws candidate values from observed target-class instances and FACE returns an existing instance reached through a graph over observed points, so both bias proposals toward the data manifold, whereas flipping the frozen model within one or two fields usually requires moving a field to an extreme value. The prediction that follows is that both should recover as the budget loosens, and they do: across K=1, K=2 and K=3, DiCE attains 25.8, 43.3 and 49.2 on Credit Default and FACE attains 13.3, 23.3 and 29.2.

B.4 Cross-model counterfactual-search results

Tables 14 and 15 report the same validity-proximity pair as Table 4 for the two additional language models.

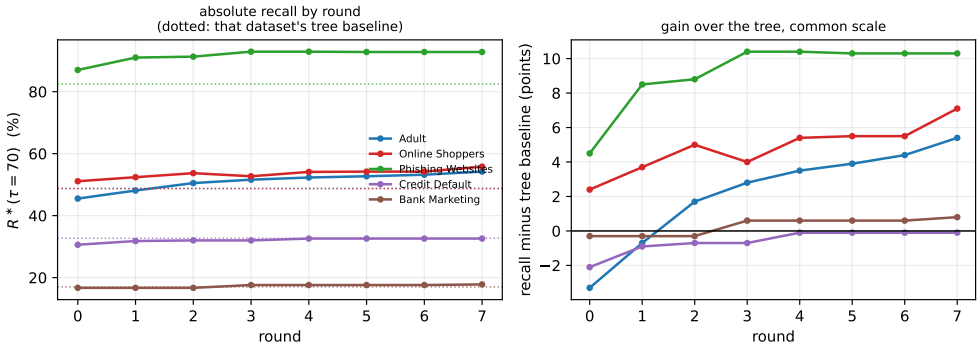

Figure 3: Rule-generation recall by DRALoop round at k=3 and τ=70 for the portfolio-aligned view with model-chosen exploration, shown as the median over six Qwen 3 235B runs. Left: absolute recall, with the tree baseline dotted. Right: recall difference from the tree baseline.

Method	OS	Adult	Credit	Phish	BM
Exact bound					
Threshold-cell MILP *(exact bound)*	100.0, 0.04	100.0, 0.76	61.7, 0.33	60.0, 0.91	96.7, 0.84
Model-agnostic references					
Random feasible search	56.7, 0.17	22.5, 1.06	20.0, 1.43	10.0, 1.34	69.2, 2.08
NICE-Q12	83.3, 0.65	63.3, 0.93	25.8, 0.28	40.0, 1.96	38.3, 0.92
DiCE-genetic	15.8, 6.75	37.5, 2.93	25.8, 7.65	8.3, 1.92	24.2, 2.04
FACE	46.7, 0.17	67.5, 0.96	13.3, 0.09	34.2, 1.57	14.2, 0.14
Static representations					
`schema_only`	100.0, 2.96	97.5, 10.12	39.2, 2.85	2.5, 2.95	38.3, 3.94
`univariate_summary`	100.0, 1.38	88.3, 3.16	29.2, 1.83	30.0, 2.21	25.8, 1.90
`leaf_statistics`	89.2, 0.67	86.7, 2.15	27.5, 1.25	30.0, 2.21	54.2, 2.20
`node_statistics`	81.7, 0.52	85.8, 2.95	32.5, 1.43	31.7, 2.05	65.8, 2.12
`node_impurity`	85.0, 0.61	85.0, 2.23	30.8, 4.54	30.8, 2.17	63.3, 2.04
`failure_geometry`	100.0, 1.26	60.0, 2.81	32.5, 1.71	40.8, 2.29	50.0, 4.45
`rule_set`	100.0, 1.10	74.2, 1.47	33.3, 1.13	37.5, 2.13	45.0, 1.86
`path`	99.2, 0.66	75.0, 1.72	29.2, 1.34	47.5, 2.17	27.5, 1.72
Budget-allocation control					
Fixed view (R=3, Q=4)	100.0, 0.76	73.3, 1.76	30.8, 1.36	45.8, 2.13	30.0, 1.52
DRALoop					
Residual-updated statistics	100.0, 0.67	75.0, 2.34	30.0, 1.80	47.5, 2.12	28.3, 1.57
+ scheduled exploration	99.2, 0.63	75.8, 2.25	28.3, 1.25	47.5, 2.18	25.8, 1.45
Portfolio-aligned view	100.0, 0.60	85.0, 1.13	43.3, 1.17	53.3, 2.22	48.3, 1.52
+ model-chosen exploration	100.0, 0.65	95.8, 1.28	41.7, 1.37	54.2, 2.26	52.5, 1.52

Table 11: Budgeted counterfactual search at K=1 with Qwen 3 235B. Each cell reports [validity, proximity] over six seeded runs with N=20 instances per run. Sparsity is omitted because it is fixed at 1.00 by the edit constraint.

C Additional Details

C.1 A worked representation example

One toy tree anchors these abstractions. Suppose the fitted tree first splits on `monthly_spend > 70`, then on `support_tickets <= 1` on the high-spend side and on `plan = pro` on the low-spend side. `schema_only` names only the fields and target; `univariate_summary` reports that positives skew toward higher spend and fewer tickets, with no branch structure; `leaf_statistics` exposes the branches with each leaf's class counts, `node_statistics` adds the internal-split counts, and `node_impurity` annotates each node's impurity. `failure_geometry` instead

Paired difference against the fixed-view control	$K{=}1$	$K{=}2$	$K{=}3$
Re-pointed view	$+10.0_{\pm 14.6}$	$+9.2_{\pm 15.7}$	$+5.0_{\pm 8.5}$
Residual statistics only, rendering fixed	$+0.2_{\pm 4.3}$	$+0.5_{\pm 7.1}$	$+0.3_{\pm 3.5}$

Table 12: Across edit budgets: paired validity difference from the fixed-view iterative control, pooled over five datasets. $K{=}1$ and $K{=}2$ use six seeded runs; $K{=}3$ uses three. Values report mean $\pm$ standard deviation.

Model	Re-pointed view	Residual statistics only
Qwen 3 235B	$+10.0_{\pm 14.6}$	$+0.2_{\pm 4.3}$
GPT-OSS 120B	$+10.8_{\pm 14.1}$	$-1.8_{\pm 6.9}$
DeepSeek-R1	$+8.2_{\pm 13.0}$	$+1.0_{\pm 8.7}$

Table 13: Across language models: paired validity difference from each model's fixed-view control at $K{=}1$, pooled over five datasets and six seeded runs. Values report mean $\pm$ standard deviation.

foregrounds errors, such as a missed high-spend positive with two tickets or a contaminated `plan = pro` branch, and `rule_set` lists the leaves the tree deploys as rules with their union precision and recall. The source tree is identical throughout; only the representation shown changes.

C.2 Dataset characteristics

Bank Marketing has low tree-rule coverage; its `duration` field is withheld because it is known only after contact. Phishing Websites has a strong tree baseline. For Adult, we withhold age, sex, race, relationship, and native country; for Credit Default, we withhold demographic fields.

Method	OS	Adult	Credit	Phish	BM
Non-LLM reference points					
Random feasible search	69.2±20.8, 0.73	23.3±6.8, 2.34	34.2±9.7, 1.60	15.8±9.7, 3.68	84.2±10.7, 3.12
NICE-Q12 (budgeted adaptation)	76.7±4.1, 0.77	98.3±2.6, 1.27	19.2±8.0, 0.08	98.3±2.6, 2.97	26.7±12.9, 1.36
Static representations: one fixed view reused across rounds					
`schema_only`	100.0±0.0, 6.51	100.0±0.0, 4.40	82.5±4.2, 8.86	10.8±3.8, 7.39	93.3±7.5, 5.94
`univariate_summary`	100.0±0.0, 3.10	100.0±0.0, 2.77	85.8±8.0, 4.83	63.3±21.6, 5.64	87.5±24.0, 2.78
`leaf_statistics`	99.2±2.0, 2.10	100.0±0.0, 1.77	61.7±5.2, 3.31	94.2±4.9, 4.91	100.0±0.0, 4.10
`node_statistics`	100.0±0.0, 0.90	100.0±0.0, 1.64	52.5±10.8, 2.66	89.2±7.4, 5.43	100.0±0.0, 4.06
`node_impurity`	100.0±0.0, 0.96	100.0±0.0, 1.64	57.5±10.8, 2.91	93.3±5.2, 5.49	100.0±0.0, 3.93
`failure_geometry`	100.0±0.0, 1.53	100.0±0.0, 2.92	76.7±11.3, 4.02	57.5±12.9, 5.49	99.2±2.0, 6.51
`rule_set`	100.0±0.0, 0.94	100.0±0.0, 1.81	76.7±8.2, 2.38	56.7±18.1, 5.99	100.0±0.0, 4.45
One-round breadth control					
Fixed view (R=12, Q=1)	83.3±15.7, 0.52	60.8±9.2, 1.18	42.5±11.3, 1.03	49.2±15.0, 3.54	35.8±22.2, 1.58
Iterative, boundary view held fixed across rounds					
Fixed view (R=3, Q=4)	99.2±2.0, 0.87	96.7±2.6, 1.93	75.0±5.5, 2.87	52.5±14.4, 5.10	78.3±18.9, 2.84
DRALoop: view updated or re-pointed at the unsolved residual					
Residual-updated statistics	96.7±6.1, 0.84	95.0±4.5, 2.02	67.5±10.4, 2.70	64.2±18.3, 5.24	75.0±27.7, 2.64
+ scheduled exploration	99.2±2.0, 0.86	96.7±4.1, 1.96	70.0±8.4, 3.61	68.3±16.3, 5.45	86.7±23.6, 2.88
Portfolio-aligned view	100.0±0.0, 0.71	90.8±6.6, 1.62	62.5±9.9, 2.07	74.2±6.6, 4.12	83.3±17.2, 2.00
+ model-chosen exploration	99.2±2.0, 0.78	96.7±4.1, 1.68	67.5±12.9, 2.02	76.7±12.1, 4.31	87.5±14.1, 2.15

Table 14: Budgeted counterfactual search with GPT-OSS 120B. Each cell reports validity as mean ± standard deviation over six seeded runs, followed by mean normalized L_1 proximity among valid candidates. All budget-matched methods use K=3, B=12, and N=20 instances per run. The breadth control allocates the budget as R=12, Q=1; the remaining LLM rows use R=3, Q=4.

Method	OS	Adult	Credit	Phish	BM
Non-LLM reference points					
Random feasible search	69.2±20.8, 0.73	23.3±6.8, 2.34	34.2±9.7, 1.60	15.8±9.7, 3.68	84.2±10.7, 3.12
NICE-Q12 (budgeted adaptation)	76.7±4.1, 0.77	98.3±2.6, 1.27	19.2±8.0, 0.08	98.3±2.6, 2.97	26.7±12.9, 1.36
Static representations: one fixed view reused across rounds					
`schema_only`	100.0±0.0, 2.73	100.0±0.0, 2.60	87.5±7.6, 10.95	40.8±23.3, 7.31	100.0±0.0, 5.73
`univariate_summary`	100.0±0.0, 1.80	100.0±0.0, 2.25	85.8±6.6, 3.99	85.0±11.0, 5.67	100.0±0.0, 3.40
`leaf_statistics`	100.0±0.0, 1.92	100.0±0.0, 1.55	84.2±5.8, 3.46	99.2±2.0, 5.19	100.0±0.0, 2.75
`node_statistics`	100.0±0.0, 1.02	100.0±0.0, 1.60	83.3±7.5, 3.28	91.7±18.1, 5.19	100.0±0.0, 2.17
`node_impurity`	100.0±0.0, 0.93	100.0±0.0, 1.50	80.0±8.9, 3.21	99.2±2.0, 5.40	100.0±0.0, 2.18
`failure_geometry`	100.0±0.0, 2.76	100.0±0.0, 2.78	85.0±6.3, 3.71	99.2±2.0, 6.11	99.2±2.0, 6.15
`rule_set`	100.0±0.0, 1.53	100.0±0.0, 1.73	82.5±2.7, 3.03	91.7±6.1, 5.11	100.0±0.0, 4.26
Iterative, boundary view held fixed across rounds					
Fixed view (control)	100.0±0.0, 1.42	100.0±0.0, 2.36	82.5±7.6, 3.37	93.3±9.8, 5.49	100.0±0.0, 3.49
DRALoop: view updated or re-pointed at the unsolved residual					
Residual-updated statistics	100.0±0.0, 1.42	100.0±0.0, 2.60	84.2±2.0, 3.39	92.5±5.2, 5.39	100.0±0.0, 3.32
+ scheduled exploration	100.0±0.0, 1.42	100.0±0.0, 2.62	82.5±5.2, 3.29	94.2±3.8, 5.60	100.0±0.0, 3.44
Portfolio-aligned view	100.0±0.0, 1.42	100.0±0.0, 2.19	80.0±5.5, 3.16	95.8±4.9, 5.06	99.2±2.0, 3.14
+ model-chosen exploration	100.0±0.0, 1.42	100.0±0.0, 2.23	82.5±4.2, 3.25	99.2±2.0, 5.49	98.3±2.6, 3.28

Table 15: Budgeted counterfactual search with DeepSeek-R1. Each cell reports validity as mean ± standard deviation over six seeded runs, followed by mean normalized L_1 proximity among valid candidates. All budget-matched methods use K=3, B=12, and N=20 instances per run; the LLM rows allocate this budget as R=3, Q=4.

D EVALUATION PROTOCOL AND PROMPT TEMPLATES

D.1 DATASET PARTITIONS

For each dataset, we retain the provided train–test partition when a canonical split is available; otherwise we construct a stratified outer split. We treat the training portion as the *generation window* and the disjoint test portion as the sealed *reporting window*. The reporting window is not used for tree fitting, representation construction, search-time feedback, rule acceptance, or portfolio selection, and is opened only for final evaluation. Table 16 gives the resulting outer partitions and class prevalence.

Dataset	Total	Generation	Reporting	Gen. %	Rep. %	Gen. pos. %	Rep. pos. %
Online Shoppers	12,330	7,056	5,274	57.2	42.8	15.69	15.19
Bank Marketing	41,188	30,891	10,297	75.0	25.0	11.17	11.54
Adult	48,842	32,561	16,281	66.7	33.3	23.85	24.08
Credit Default	30,000	22,500	7,500	75.0	25.0	21.96	22.60
Phishing Websites	11,055	8,291	2,764	75.0	25.0	44.05	45.08

Table 16: Outer dataset partitions and positive-class prevalence.

Within each generation window, we form a balanced subset containing the positives and an equal-sized sample of negatives. We use 80% of this subset to fit the tree and 20% as an internal holdout for search-time statistics. Rule acceptance and portfolio selection also use the generation window.

D.2 PROMPT TEMPLATES

The experiments use fixed task prompts with marked variable fields. For the seven static representations in Table 1, the task instruction, objective, field whitelist, output schema, parser, and inference budget are identical; only `[TREE_REPRESENTATION]` changes. Dataset-specific fields such as `[TASK_NAME]`, `[TASK_DESCRIPTION]`, and the field lists are fixed within a dataset and do not depend on the representation. In the iterative conditions, the additional state-dependent fields shown below vary by round by design.

Rule generation. The prompt uses the same precision floor, support requirement, and rule budget as the evaluator. Here "held-out" refers to the internal holdout within the generation window.

```
<role>Detection analyst producing deployable enforcement rules for
[TASK_NAME].</role>
<objective>
A rule is DEPLOYABLE only if its precision on a held-out,
fully-labelled window is >= [TAU]% while flagging at least [MIN_HITS]
instances. LIFT IS NOT PRECISION: a single high-lift feature is almost
always low-precision on its own. Reach precision by AND-ing signals so the
joint set is rare AND genuinely positive.
</objective>
<selection_objective>
The downstream evaluator keeps at most [K] rules and scores the UNION of the
kept rules. Optimize the SET, not isolated rules. Prefer rules that each
individually clear the gate and jointly cover DISTINCT positives. Avoid
emitting near-duplicate variants of the same branch. Order rules from
highest expected marginal coverage to lowest.
</selection_objective>
<proposal_budget>
You may return up to [R] candidate rules. The first [K] should be your best
union-oriented set; any additional rules should be backup alternatives that
target different regions rather than near-duplicates.
</proposal_budget>
<task_context>
[TASK_DESCRIPTION]
</task_context>
<data>
[DATASET_SUMMARY_STATISTICS]
```

```
</data>
[TREE_REPRESENTATION]
<allowed_fields>
Numeric (gt/gte/lt/lte): [NUMERIC_FIELD_LIST]
Categorical (eq/neq/in/not_in; values EXACTLY as shown):
[CATEGORICAL_FIELD_LIST]
</allowed_fields>
<instructions>
[SET_CONSTRUCTION_INSTRUCTION]
[SHARED_PRINCIPLES]
</instructions>
<output_format>
Return ONLY a JSON object, no markdown:
{"rules":[{"description":"which DISTINCT sub-population",
           "conditions":[{"field":"<allowed>","operator":"<op>",
                          "value":<number or [list]>}]}]}
</output_format>
```

Budgeted counterfactual search. The prompt specifies the edit constraint and allowed feature domains.

```
<role>Counterfactual explanation generator for [TASK_NAME]. You identify
small input changes that alter a frozen model's prediction; you are not
claiming those changes cause the real-world outcome.</role>
<task_context>
[TASK_DESCRIPTION]
Requested model target: class [TARGET_CLASS], meaning [TARGET_NAME].
</task_context>
<situation>
A frozen model T currently predicts class [SOURCE_CLASS] for the instance
below. Propose the SMALLEST in-domain set of feature changes that makes T
predict target class [TARGET_CLASS] ([TARGET_NAME]). Change only the
permitted fields listed. Prefer few changes of small magnitude; do not
restate features you are not changing. A HARD BUDGET applies: a sparsity
budget of AT MOST [K] changed fields. Any proposal that exceeds either
budget is REJECTED even if it reaches the target -- prefer the CHEAPEST,
SPARSEST successful counterfactual.
</situation>
[PREVIOUS_ATTEMPT_FEEDBACK]
<model_representation>
This describes the frozen model T (the same representation used elsewhere in
this study). Use it to reason about which feature changes cross the model's
decision boundary.
[TREE_REPRESENTATION]
</model_representation>
<mutable_fields>
Numeric (you may set a new value only inside the listed generation-derived
range): [NUMERIC_FIELD_BOUNDS]
Categorical (set a new level, EXACTLY as one of the listed strings):
[CATEGORICAL_FIELD_LEVELS]
</mutable_fields>
<instance>
Current feature values (predicted class [SOURCE_CLASS]):
[INSTANCE_FEATURE_VALUES]
</instance>
<output_format>
Return ONLY a JSON object, no markdown:
{"edits":[{"field":"<mutable field>","value":<new number or "new level">}]}
</output_format>
```

Adaptive state inserted by the loop. For rule generation, the loop can append feedback from the preceding round:

```
FEEDBACK FROM LAST ROUND. You proposed [N_PROPOSED] rules; [N_KEPT] cleared
```

```
the gate and were kept. These did NOT clear it:
  - [CONDITIONS]  (generation-frame precision [P], holdout precision [P_HO],
                   support [N_HITS])
Iterate on this: a rule JUST BELOW the precision floor is close, so ADD one
condition to tighten it and resubmit; a rule with too FEW hits is too
narrow, so relax a threshold or drop a condition to broaden it; abandon
rules that were far off. Turn the near-misses into gate-clearers.
```

Here `[P]` is precision on the full generation window and `[P_HO]` is precision on its internal holdout. The reporting window is used only for final evaluation.

In the re-pointed conditions, the prompt also announces that the rendered representation has been recomputed on the currently uncovered positives:

```
You have already deployed [N_ACCEPTED] rules that cover [N_COVERED] of
[N_TOTAL] positives. Every statistic below is recomputed over ONLY the
[N_UNCOVERED] positives those rules STILL MISS. Your job this round is to
cover that residual -- propose rules that catch the uncovered positives,
not ones that re-cover what is already caught.
```

The fixed-view control uses the same rounds and feedback without changing the rendered representation. The residual-statistics control receives the residual counts while keeping the per-rule representation fixed. Counterfactual rounds include the preceding attempt and its model outcome.

Output parsing and inference settings. Both tasks require a single JSON object and no prose. Invalid JSON, fields outside the whitelist, or unsupported operators are discarded rather than repaired, with the same parser and validation logic used for every representation condition. No decoding parameters are sent. The negative subsample, internal split, fitted ensemble, and mined rules are seeded, while model generation itself is not. The output-token cap is $4{,}000$ for all three language models and the reasoning-effort hint is fixed at `medium` across conditions.