UVET: A Partitioning Approach for Continuous Treatment Assignment At Scale

Artem Betlei[1*] **Mariia Vladimirova**[1,2*] **Victor Girou**[1,3†] **Thibaud Rahier**[1]
[1]Criteo AI Lab [2]FairPlay joint team [3]ENS Paris-Saclay
{a.betlei, m.vladimirova, v.girou, t.rahier}@criteo.com

Abstract

Many services have a fixed budget to spend on their users, such as discounts in retail, doses in medicine, compute in cloud systems or bids in advertising, and must decide how much each user receives. Users respond differently, and each extra unit of spend usually buys less value than the one before. We introduce the **Unit Value Equalization Tree (UVET)**, a decision tree that splits users into a few groups and spends on each group until the next unit of value costs the same in all of them. When value and cost follow power laws in the treatment intensity, this rule gives the budget split in **closed form**, so the tree needs no integer program to assign treatments and scales to tens of millions of users. We also release **UVET-benchmark**, a randomized experiment with 86.7 million ad impressions, five ordered treatment levels, and both value and cost outcomes. On a public coupon-marketing dataset rescaled to have diminishing returns, UVET gains about 50% more value than the best baseline while overspending less, and it keeps the best score once overspending is penalized. On UVET-benchmark, where value responds weakly to the bid, the best UVET partition that stays within budget gains about three times as much value as the best baseline that does, and a paired test puts it ahead of six of the eight baselines. The dataset is available at https://huggingface.co/datasets/criteo/uvet-benchmark.

1 Introduction

Consider a retailer allocating promotional budget across millions of customers. Each dollar spent on customer i yields diminishing returns: the first discount drives a large response, but successive increases in generosity produce smaller and smaller lifts (Sethuraman et al., 2011; Shephard and Färe, 1974). The firm observes both a value outcome (revenue) and a cost outcome for each treatment intensity, and seeks a policy that maximizes total value subject to a global budget constraint. This problem (personalized, cost-aware assignment of an ordered treatment under diminishing returns) arises wherever resources must be allocated heterogeneously: ad bidding (Betlei et al., 2024), clinical dosing (Chen et al., 2016; Brock et al., 2021), pricing (Kilts Center for Marketing, University of Chicago Booth School of Business, 2026), and computational infrastructure, where data-centre electricity consumption is projected to double by 2030 (International Energy Agency, 2025) yet each additional unit of compute yields ever-smaller performance gains (Hoffmann et al., 2022; Schwartz et al., 2020).

Existing approaches split into two camps. *Two-stage methods* first estimate heterogeneous treatment effects (HTE) via causal forests (Athey et al., 2019; Wager and Athey, 2018) or causal trees (Athey and Imbens, 2016), then solve a knapsack-constrained allocation. Sun et al. (2021) show that the

*Equal contribution.
†The work was done during a master's internship at Criteo AI Lab.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

optimal budget-constrained policy thresholds on the ratio $\tau_v(x)/\tau_c(x)$; Ai et al. (2022) operationalize this via Lagrangian bisection on causal forest scores. Exact knapsack allocation requires an integer program at inference time (Tu et al., 2021; De Vos et al., 2026), and optimal prescriptive trees a combinatorial search at training time (Jo et al., 2021; Amram et al., 2022), which limits scalability. *Decision-focused methods* integrate prediction and optimization (Mandi et al., 2023; Elmachtoub and Grigas, 2022), yet recent work confirms that even state-of-the-art CATE estimators can be suboptimal for downstream decisions (Frauen et al., 2025; Casacuberta and Hardt, 2026). Existing decision-focused implementations still decouple user-space partitioning from cost-aware allocation (Betlei et al., 2024; Tu et al., 2021).

We take a different route. Under a power-law (elastic) relationship between treatment intensity and outcomes, an assumption backed by decades of evidence in economics, advertising, and pharmacology (Section 2.2), the constrained allocation across user cohorts admits a *closed-form solution* obtained by equalizing the marginal cost per unit value (mCUV) across cohorts. Exploiting structural assumptions to obtain closed-form allocations is a classical idea: in information theory, the water-filling principle yields an analytic power allocation by equalizing marginal channel capacity (Cover and Thomas, 2006); in production economics, Cobb-Douglas returns give closed-form input mixes proportional to factor elasticities (Varian, 1992). The same principle reappears in modern machine learning and operations research: under power-law neural scaling laws, the compute-optimal split between model parameters and training tokens admits a closed-form solution (Kaplan et al., 2020; Hoffmann et al., 2022), and in online resource allocation, dual-based methods exploit concavity to produce analytic per-step decisions without per-instance optimization (Balseiro et al., 2023). Our contribution is to bring the same principle, concavity-driven marginal equalization, to personalized treatment assignment and embed it directly into a tree-splitting criterion. Under the power-law assumption the per-cohort allocation becomes concave and separable and admits a closed-form marginal-equalization solution, which removes the per-inference integer program. The speedup comes from this added structure; the general multiple-choice knapsack remains NP-hard. The resulting algorithm, UVET (Unit Value Equalization Tree), simultaneously partitions the user space *and* solves the allocation, with no post-hoc optimization step.

Contributions

- **UVET (algorithm).** A greedy tree that scores each candidate split by the total value of its two children under the optimal iso-cost allocation of a per-region power-law model. For any fixed partition, the native rule equalizes marginal cost per unit value across buckets in closed form, with no per-inference integer program, in contrast to "estimate effects, then knapsack" pipelines (Tu et al., 2021; De Vos et al., 2026; Ai et al., 2022). On the semi-synthetic MT-LIFT$_{\text{scaled}}$ benchmark the Bayesian variant UVET$_{\text{bay}}$ gains the most value under LP (12.5%, against at most 8.2% for the baselines) at a 1.6% overspend, and it has the best budget-penalized score. On UVET-benchmark, where value reacts weakly to the bid, UVET$_{\text{uncl}}$ gains the most value of the methods that stay within budget under LP, and a paired bootstrap separates it from six of the eight baselines (Section 4).
- **UVET-benchmark (resource).** An 86.7M-impression public dataset from randomized industrial A/B tests with ordered treatment intensities, covariates, and dual value–cost outcomes. No existing public benchmark simultaneously provides these five ingredients (ordered treatment, individual covariates, experimental support, dual outcomes, and enough scale for heterogeneous subgroup discovery), yet diminishing-returns settings are widespread in advertising, compute scaling, and energy-constrained systems (Sethuraman et al., 2011; Kaplan et al., 2020; Strubell et al., 2019). UVET-benchmark is designed to close that gap (Table 4, Appendix B).

Related work, a taxonomy of baseline families (Table 3), and the benchmark landscape are in Appendices A and B.

2 Problem formulation

Let $\mathcal{X} \subset \mathbb{R}^d$ be a $d-$dimensional feature space, and $X \in \mathcal{X}$ be the random variable containing users *features*.

We assume we have at our disposal a collection of different treatments $\mathcal{T}$: in all generality this set might be anything from finite to continuous, but we do not make any further assumption on the structure of this set at this point. For simplicity, we will however consider that $\mathcal{T}$ always contains a *reference* treatment t_0, which is a default treatment that is assigned to each user outside of 'testing' periods (for example in digital advertising, the reference treatment corresponds to the way production operates by default; in medicine it is the standard dose). We denote $\mathbf{Y} = (Y^c, Y^v) \in \mathbb{R}^2$ the random *outcome vector* we observe after treatment assignment to users. The random variables Y^c and Y^v are respectively designating the cost and value at the user level.

Let us consider a fixed user u with features $x \in \mathcal{X}$ and outcome vector $\mathbf{y} \in \mathbb{R}^2$ (which are respectively realizations of the random variables X and $\mathbf{Y}$). For any treatment $t \in \mathcal{T}$, we denote $\mathbf{y}(t) = (y^c(t), y^v(t))$ the *potential outcome* vector (Rubin, 1974), corresponding to the outcome vector we would have observed **had treatment** t **been assigned to user** u.

2.1 Multi-treatment cost-constrained assignment

An optimal multi-treatment assignment (or policy) corresponds to a mapping $\psi : \mathcal{X} \to \mathcal{T}$ assigning treatments to users so that the *value of interest* Y^v is maximized (in expectation) while ensuring the cost Y^c is controlled. Typically, we ensure the expected cost of the policy is at most equal to the expected cost resulting from applying the reference treatment t_0 to each user. In the following, we denote $C = \mathbb{E}[Y^c(t_0)]$ our allowed budget in the multiple treatment allocation problem.

With this in hand, we can now formalize the general treatment assignment – or policy learning – problem at the user level:

$$\max_{\psi:\mathcal{X}\to\mathcal{T}} \mathbb{E}[Y^v(\psi(X))] \quad \text{s.t.} \quad \mathbb{E}[Y^c(\psi(X))] \leq C, \tag{1}$$

where the expectations are implicitly defined with respect to the joint distribution of variables X and $(\mathbf{Y}(t))_{t\in\mathcal{T}}$ underlying our considered setting.

Solving the treatment assignment problem This is a multiple-choice knapsack problem with stochastic rewards, which has been extensively studied in the past decades. It can be solved with linear programming (LP), which however does not scale easily: optimization solvers are not typically designed for large-scale datasets, such as those encountered in online advertising (Tu et al., 2021; Albert and Goldenberg, 2022; Zhou et al., 2023a).

Treatment effect estimation and bucketization Whatever the method which is used to solve (1), there remains the problem of estimating the effect of each treatment $t \in \mathcal{T}$ on each user. Provided we have access to data resulting from a randomized controlled trial, a good solution to balance bias and variance is to use bucket-level treatment effect estimation (instead of user-level), which requires a bucketization of the user space (Tu et al., 2021).

Reparametrization of the treatment assignment problem assuming a bucketization of the user space A partition of the user space is given by a function $\pi : \mathcal{X} \to \mathcal{B}$, mapping the user feature space $\mathcal{X}$ to a discrete set of buckets $\mathcal{B}$. Assuming a given partition function $\pi : \mathcal{X} \to \mathcal{B}$, treatment assignment problem (1) can be rewritten as

$$\max_{\phi:\mathcal{B}\to\mathcal{T}} \sum_{b\in\mathcal{B}} Y_b^v(\phi(b)) \quad s.t. \sum_{b\in\mathcal{B}} Y_b^c(\phi(b)) \leq C, \tag{2}$$

where for every bucket $b \in \mathcal{B}$ and treatment $t \in \mathcal{T}$,

$$\mathbf{Y}_b(t) = \mathbb{E}[\mathbf{Y}(t)|\pi(X) = b] \cdot \mathbb{P}(\pi(X) = b),$$

designates the (weighted) *bucket level* expected outcome vector.

2.2 Continuous treatment and elasticity assumptions

Continuous treatment set assumption In the following, we assume the treatment collection set $\mathcal{T}$ is of the form

$$\mathcal{T} = \{t_0(\alpha),\ \alpha \in A\}, \tag{3}$$

where $A \subset \mathbb{R}_+$ is an interval of the form $[\alpha_{\min}, \alpha_{\max}] \ni 1$, and $t_0(.)$ is a *treatment parametrization* such that the reference treatment is $t_0(1)$. An example of such a continuous treatment set is presented by Betlei et al. (2024), where the authors consider a set $\Pi = \{\alpha \cdot \pi_0\}_{\alpha \in I}$ of 'candidate policies' (which act as different treatments) in the context of bidding strategy design.

In practice treatments are bounded for two reasons. First, some hypotheses hold only locally, so solutions should not stray too far from the reference treatment. Second, production systems typically restrict large deviations from the reference to avoid unintended behavior.

Simplified notations For simplicity, we will denote $\mathbf{Y}(\alpha)$ the potential outcome variable corresponding to treatment $t(\alpha) \in \mathcal{T}$, *i.e.* $\mathbf{Y}(\alpha) = \mathbf{Y}(t(\alpha))$. We also designate the two components of $\mathbf{Y}$ by (c, v) instead of (Y^c, Y^v). In particular for a bucket $b \in \mathcal{B}$ we denote $c_b := Y_b^c$ and $v_b := Y_b^v$.

Diminishing returns assumption The diminishing returns principle, whereby marginal gains decrease with increasing input, is standard across economics, engineering, and machine learning (Gabaix, 2009; Shephard and Färe, 1974; Clauset et al., 2009; Kaplan et al., 2020); we refer the reader to Section 1 for motivating examples. We operationalize it below through a specific power-law (elastic) functional form.

Elasticity assumption Elastic (power-law) relationships are widely used to model the sensitivity of an outcome to a change in an input (Phillips, 2021; Zhao et al., 2019; Zhou et al., 2023a). This assumption appears across domains: advertising cost typically shows an elastic return curve (Yan et al., 2023; Zhou et al., 2023a); in pharmacology, the Hill (Emax) model describes concave dose–response relationships via a power-law-like saturating form (Goutelle et al., 2008); electricity demand responds to price with a constant elasticity commonly estimated in the range -0.2 to -0.5 (Labandeira et al., 2017); and in machine learning, scaling laws show that training loss decreases as a power law of compute (Kaplan et al., 2020), a relationship that persists across hardware configurations even as marginal returns diminish at scale (Fernandez et al., 2024; Lu, 2025). We choose to model the relationships between the treatment parameter α, cost c and value v as power-laws (elastic) inside any given partition $\pi : \mathcal{X} \to \mathcal{B}$. Formally, for any bucket $b \in \mathcal{B}$, we assume there exist parameters γ_b and η_b such that:

$$v_b(\alpha) = v_b(1)\alpha^{\gamma_b}, \quad c_b(\alpha) = c_b(1)\alpha^{\eta_b}. \tag{4}$$

This implies the following relationship between average value and cost inside bucket b:

$$\forall c_b \in [c_b(\alpha_{\min}), c_b(\alpha_{\max})], \quad v_b(c_b) = v_b(1)\left(\frac{c_b}{c_b(1)}\right)^{\frac{\gamma_b}{\eta_b}}.$$

For any $b \in \mathcal{B}$, γ_b and η_b have to satisfy $0 < \gamma_b < \eta_b$, in order for the function $c_b \mapsto v_b(c_b)$ to be (strictly) concave. This concavity is needed for the principle of *diminishing returns* to apply: the higher the cost allocated in bucket b, the smaller the effect on value of an additional marginal unit of cost.

Robustness and limitations of the power-law assumption Constant elasticity is stronger than concavity, and we use (4) as a local model on the range of randomized treatment levels ($[0.8, 1.2]$ in our data), without extrapolating beyond it. Theorem 1 (Appendix C.5) bounds the cost of this approximation: if the true log-response deviates from the fitted power law by at most δ in level and in local elasticity, and the oracle allocation is interior, the relative value loss is $O(\delta)$; Appendix H measures it on saturating, logarithmic, piecewise power-law, and non-monotone responses. The assumption enters partition learning and the mCUV allocator only, while the LP and ILP evaluators use empirical totals per treatment level. The assumption can still break in practice: for example, when a brand already dominates a search term, additional advertising cost yields nearly flat returns rather than a smooth power-law curve; or when users choose purely on price (e.g. generic USB-cable campaigns), advertising cost has little nonlinear effect on conversions. In such regimes the response may still be locally elastic, but the global power-law parametrization becomes unreliable.

3 The UVET approach

Usually, the criterion we want to maximize is not expressible in a closed-form way with respect to quantities that can be directly estimated from data. However the elasticity assumption from Sec-

tion 2.2 (on the value and cost outcomes) enables this criterion to be expressed directly from offline data estimates, while solving the treatment assignment problem implicitly at the same time. It therefore performs jointly both the *partitioning* (step 1) and *treatment assignment* (step 2) problems.

3.1 Treatment assignment problem

In this section we consider a given partition $\pi : \mathcal{X} \to \mathcal{B}$. Our chosen treatment assignment should respect total fixed budget C. Since (1) value is monotone increasing in cost under our power-law assumptions and (2) treatments are continuous, the optimum is achieved at equality. Thus, we rewrite the constraint in (2) as a strict iso-cost constraint: $\sum_b c_b(\alpha_b) = C$. Overall, our optimization problem is written as:

$$\max_{\phi:\mathcal{B}\to\mathcal{T}} \sum_{b\in\mathcal{B}} v_b(\phi(b)) \quad \text{s.t.} \quad \sum_{b\in\mathcal{B}} c_b(\phi(b)) = C, \tag{5}$$

In the following, for all $b \in \mathcal{B}$, we will denote $\alpha_b = \phi(b)$.

3.2 Marginal Cost equalization

In microeconomic theory, a market is said to be in equilibrium when supply and demand curves intersect, determining the equilibrium price and quantity, and in competitive markets each participant sets their marginal cost equal to this equilibrium price (Varian, 1992). Analogously, we can view our setting as a market where each bucket represents a player and the outcome value corresponds to the quantity. At equilibrium, the marginal cost per unit of value in each bucket should be equal to a single shadow price. We therefore characterize an iso-cost optimal treatment assignment as one for which the marginal cost per unit of value is equal across buckets to a fixed value μ (computed below), that is:

$$\forall b, b' \in \mathcal{B} \quad mCUV_b(\alpha_b) = mCUV_{b'}(\alpha_{b'}) := \mu, \tag{6}$$

where for all $b \in \mathcal{B}$, $mCUV_b(\alpha_b)$ is the marginal cost per unit value in bucket b, evaluated for $\alpha = \alpha_b$ with

$$\forall \alpha \in A, \quad mCUV_b(\alpha) = \frac{\partial c_b}{\partial v_b}(\alpha). \tag{7}$$

3.3 Optimal splitting criterion

A closed-form expression for the treatment parameter ensuring equalization of mCUVs We formulate the optimal policy in the following proposition, proved in Appendix C.1.

Proposition 1 (Optimality of closed-form α_b that equalizes mCUV)**.** *Fix a partition $\mathcal{B}$ and assume* (4) *holds in each bucket. Then choosing, for a given $\mu > 0$,*

$$\alpha_b^*(\mu) \;=\; \left(\frac{\mu}{\mathrm{mCUV}_b(1)}\right)^{\frac{1}{\eta_b - \gamma_b}} \tag{8}$$

ensures $\mathrm{mCUV}_b(\alpha_b^*(\mu)) = \mu$ *for all $b \in \mathcal{B}$.*

Determining the value of μ We have shown thanks to Proposition 1 how to choose the value of the treatment parameters α_b in each bucket to ensure all marginal costs per unit value are equal to a fixed value μ. This value is constrained by the fact that we want our allocation ϕ to be iso-cost with respect to our reference policy. This characterizes entirely the value of μ which is used in (6) according to the following implicit equation:

$$\sum_b c_b(1) \left(\frac{\mu}{mCUV_b(1)}\right)^{\frac{\eta_b}{\eta_b - \gamma_b}} = C. \tag{9}$$

There exists a unique value of μ satisfying (9) (see Proposition 3 in Appendix C.2), which can be comfortably found with an optimization solver (e.g. by dichotomy or Brent's method).

Optimality of the mCUV equalization approach. In the next proposition, we now formally state that the iso-cost mCUV equalization approach, consisting in using treatment parameters α_b^* given by (8) in each bucket b, where μ is implicitly defined in (9), is indeed optimal in terms of value maximization under cost constraint. This result is proven in Appendix C.1.

Proposition 2. *Fix a partition $\mathcal{B}$ and assume (4) holds in every bucket. Let $\mathbf{a}^*(\mu) = (\alpha_b^*(\mu))_{b\in\mathcal{B}}$ be given by (8), where μ satisfies (9). Then*

$$\mathbf{a}^*(\mu) \in \underset{(\alpha_b)_{b\in\mathcal{B}}}{\operatorname{argmax}} \left\{ \sum_{b\in\mathcal{B}} v_b(\alpha_b) \right\} \quad \textit{s.t.} \quad \sum_{b\in\mathcal{B}} c_b(\alpha_b) = C. \tag{10}$$

In particular, $\mathbf{a}^*(\mu)$ solves the central optimization problem (5).

Proposition 2 certifies the allocation for a *fixed* partition. The partition itself is grown greedily (Section 3.4) and carries no global optimality guarantee; Appendix C.6 discusses this trade-off against mixed-integer trees.

Closed-form splitting criterion Now, computing the value of μ from (9), then injecting the resulting values of α_b^* given by (8) in our optimization objective (5) yields the following closed-form criterion:

$$\sum_{b\in\mathcal{B}} v_b(\alpha_b^*), \tag{11}$$

which can be readily estimated using offline data estimates of the elasticity parameters for cost and value. The core idea of our UVET approach is to perform recursive splitting as in any decision tree algorithm using this criterion in order to determine the optimal successive splits. We give a detailed pseudocode of this algorithm in the next subsection. The optimal solution will be the policy ϕ^* such that for every bucket b, $\phi^*(b) = \alpha_b^*$ where the values of α_b^* are given by Proposition 1.

3.4 UVET method for partitioning

We assume that in any region of the user space, cost-value relationships can be modeled as power-law relationships as presented in (4). Our goal is to use offline data to find a sequence of thresholds performing binary split of the nested regions following the generic decision tree blueprint (Breiman, 1984). We provide pseudocode for our method in Algorithm 1.

Recursive tree Thorough exploration of all possible partitions is computationally prohibitive with large-scale data. Our `BuildTree` function circumvents this problem by adopting a greedy, recursive approach to partition exploration. We assume access to a `candidate_generator` function that produces candidate splits expressed as a tuple (feature, threshold): a percentile grid between the 25th and 75th percentiles of each feature, scanned with a unimodal search (Appendix F.6). For every candidate, a scalar root search gives μ and the closed form gives the children's α_b^*, so treatment levels are never searched. The splitting criterion is the one presented in (11) with only two buckets. At each split search, we select the (feature, threshold) binary split which maximizes this criterion (which ensures maximally increased value at iso-cost). Once the split is found, we recursively find next splits considering both resulting leaves as input nodes for the two subsequent applications of the `BuildTree` function. The algorithm stops when we reach a given `stopping_criterion`, defined from hyperparameters `max_depth`, `min_samples_split`, and `min_samples_leaf`, defined in Appendix F.1.

Estimating elasticity parameters The value and cost elasticity parameters are not known in advance and must be estimated for each region of interest. To do so, for each candidate (feature, threshold) split, we aggregate observed value and cost by treatment level α inside the 'left' and 'right' regions, and obtain $\hat{\gamma}_b, \hat{\eta}_b$ as the OLS slopes of $\log v_b(\alpha)$ and $\log c_b(\alpha)$ regressed on $\log \alpha$ (see Appendix C.4 for the consistency argument). This requires our dataset to contain at least two distinct (randomly assigned) treatment levels in each region, which is the case when using data from a randomized controlled trial or A/B test with two or more treatment policies. The reference quantities $v_b(1)$ and $c_b(1)$ in (4) are the observed value and cost totals of region b under the reference treatment α=1; they anchor the fitted curves, so $v_b(\alpha) = v_b(1)\alpha^{\hat{\gamma}_b}$ and $c_b(\alpha) = c_b(1)\alpha^{\hat{\eta}_b}$ reproduce the reference outcomes without any counterfactual estimation.

Bayesian elasticity regularization Log–log estimates of γ_b and η_b rest on one point per logged treatment level and can be noisy in small nodes. UVET_{bay} pulls each node estimate $\hat{\theta}_b$, $\theta \in \{\gamma, \eta\}$, toward the global (root-level) estimate $\hat{\theta}_{\text{global}}$:

$$\hat{\theta}_b^{\text{post}} = \frac{\lambda_0 \hat{\theta}_{\text{global}} + L \hat{\theta}_b}{\lambda_0 + L}, \qquad (12)$$

the posterior mean of a Normal–Normal model in which the prior is worth λ_0 points and the node regression its L points, L being the number of logged treatment levels (L=5 on both benchmarks). Since L is the same in every node, every node receives the same weight $L/(\lambda_0 + L)$ whatever its size, a fixed-weight shrinkage in the spirit of empirical-Bayes estimators (Efron, 2012); weighting each node by the precision of its own regression would adapt the weight to the node and is a natural extension. We choose λ_0 on validation rows, which gives λ_0=20 on $\text{MT-LIFT}_{\text{scaled}}$ and λ_0=10 on UVET-benchmark (Appendix G.6.2). Pulling every node toward the same value also shrinks the differences between cohorts by the factor $L/(\lambda_0 + L)$; Section 4 shows when this helps.

Algorithm 1: UVET Algorithm

Input: Train dataset $\mathbf{X}$ containing samples $x \in \mathbf{X}$ where $x = (x_i)_{i \in I}$, I being the set of *features*, value function v, cost function c, reference treatment α, `stopping_criterion`, `candidate_generator`

Output: Tree T

Function `BuildTree`$(\mathbf{X}, v, c, \alpha)$**:**
- **if** *`stopping_criterion` met* **then**
  - **return** leaf node $(\mathbf{X}, v)$ with predicted value
- $v_{\max} \leftarrow v$
- **for** (i, s) *from* *`candidate_generator`* **do**
  - `// Find the best split maximizing` $\sum_b v_b(\alpha_b^*)$
  - $\mathbf{X}^{\text{left}} \leftarrow \mathbf{X} \cap \mathbb{I}[x_i \leq s]$, $\mathbf{X}^{\text{right}} \leftarrow \mathbf{X} \setminus \mathbf{X}^{\text{left}}$;
  - For $b \in \{\text{left}, \text{right}\}$: $v_b(1), c_b(1)$ at α=1, γ_b, η_b by log–log OLS, $mCUV_b(1)$ by (15);
  - Compute μ satisfying (9) (equivalently (6));
  - Compute α_b^* for $b \in \{\text{left}, \text{right}\}$, see (8);
  - Compute $v_b^* = v_b(\alpha_b^*)$ for $b \in \{\text{left}, \text{right}\}$;
  - Compute $v^* = v_{\text{left}}^* + v_{\text{right}}^*$;
  - **if** $v^* > v_{\max}$ **then**
    - $v_{\max} \leftarrow v^*, \quad (i^*, s^*) \leftarrow (i, s)$;
- Create an internal node with split (i^*, s^*);
- Partition $\mathbf{X}$ into $\mathbf{X}^{\text{left}}$ and $\mathbf{X}^{\text{right}}$ based on (i^*, s^*);
- `// Recursively build sub-trees`
- $T_{\text{left}} \leftarrow$ `BuildTree`$(\mathbf{X}^{left}, v_{left}, c_{left}, \alpha_{left}^*)$;
- $T_{\text{right}} \leftarrow$ `BuildTree`$(\mathbf{X}^{right}, v_{right}, c_{right}, \alpha_{right}^*)$;
- Attach T_{left} and T_{right} to the internal node;
- **return** tree T

`// Start tree construction`

$T \leftarrow$ `BuildTree`$(\mathbf{X}, v, c, \alpha = 1)$;

return T

4 Experiments

We benchmark UVET on two public-scale RCT logs below ($\text{MT-LIFT}_{\text{scaled}}$ and *UVET-benchmark*). Beyond these tables, the appendices add a toy split where HTE-based partitioning disagrees with UVET's objective (Appendix D), power-law sweeps over depth, training size, budget, and wall time (Appendix F.5), and a **robustness** suite under elasticity misspecification and noise (Appendix H). **Ablations** vary λ_0 and leaf floors for UVET_{bay} (Appendix G.6.2) and the penalty weight of S_{comb} (Appendix G.6.1). Full seeds, hardware, experiment-matrix defaults, and reproduction notes are in Appendix F, and the code, with a script that reruns every experiment, is at https://github.com/criteo-research/uvet.

Evaluation metrics For partition $\mathcal{B}$, write totals $v(\phi_{\mathcal{B}}^*) = \sum_{b \in \mathcal{B}} v_b(\phi_{\mathcal{B}}^*(b))$ and $c(\phi_{\mathcal{B}}^*) = \sum_{b \in \mathcal{B}} c_b(\phi_{\mathcal{B}}^*(b))$ at the learned assignment, and $v(1), c(1)$ under the reference (Appendix G). Relative uplifts are

$$\Delta v_{\mathcal{B}} = \frac{v(\phi_{\mathcal{B}}^*)}{v(1)} - 1, \quad \Delta c_{\mathcal{B}} = \frac{c(\phi_{\mathcal{B}}^*)}{c(1)} - 1$$

(higher $\Delta v_{\mathcal{B}}$ is better; feasible policies target $\Delta c_{\mathcal{B}} \leq 0$). Tables 1 and 2 also report the budget-penalized score S_{comb} (Appendix G.6). They report uplifts under three downstream allocators: **LP**, which we treat as the primary comparator because the *same* linear-programming evaluator is applied to every learned partition, and which optimizes total value subject to a global cost budget by assigning each cohort a *mixture* over the observed randomized discrete policies, i.e. a probability vector over the logged treatment levels, so that the cohort's value and cost are the correspondingly weighted empirical totals per treatment level (no UVET-specific power-law map; Appendix G.4); **ILP**, which instead commits each cohort to a single policy; and **mCUV** equalization (Section 3.2). Definitions and the Qini/AUUC link are in Appendix G; allocator bias in Appendix G.5.

Baselines. Tables 1–2 compare UVET variants to a shared grid of forest, budget-aware, prescriptive, and CADR (De Vos et al., 2026) baselines; names, implementation details, and subsample caps are in Appendix F.4, taxonomy in Table 3. UVET_{cl} scores splits with α_b^* clipped to the logged treatment levels and $\text{UVET}_{\text{uncl}}$ with $\alpha_b^* \in [0.001, 1.999]$; the mCUV allocator clips α_b^* to the logged range in every table. Both tables group rows into three blocks: baselines, which use no UVET partition at any step; the UVET variants; and *UVET-derived merged partitions*. A merged partition intersects the leaves of two partitions learned by other methods, following the leaf-intersection step of Tu et al. (2021): Merge (UVET_{cl} + CF_{mse}) and Merge (UVET_{cl} + CF_{het}) intersect the clipped UVET partition with a CF_{mse} or CF_{het} partition, and on UVET-benchmark (Table 2) the baseline Merge (CF_{mse} + CF_{het}) applies the same step with no UVET input. Merged partitions have 8 to 14 cohorts, against at most 4 for a single depth-2 tree, which gives the LP and ILP allocators more room. Baselines mostly use the scalar `rev` proxy (Appendix F.3); UVET splits on dual (v, c) (Section 3).

4.1 MT-LIFT data

MT-LIFT[3] is a public Meituan coupon RCT (∼5.5M rows, 99 features, five coupon treatments, click/conversion) (Huang et al., 2024a) with the official split, which holds out the last week for testing and keeps the rest for training.

Preprocessing We treat coupons as ordered intensities $\{0.8, \ldots, 1.2\}$ (reference 1.0), label click as cost and conversion as value, and apply IPW (Zhou et al., 2023a; Yan et al., 2023). Aggregate log–log fits on the raw RCT outcomes violate the diminishing-returns condition $\gamma < \eta$ from Section 2: under our click-as-cost / conversion-as-value labeling, the global training elasticities satisfy $\hat{\gamma}_{\text{glob}} > \hat{\eta}_{\text{glob}}$, so evaluating UVET "out of the box" on those raw outcomes would *misalign* the benchmark with the paper's modeling target. **MT-LIFT$_{\text{scaled}}$** is therefore a semi-synthetic variant: we keep the real Meituan randomization, features, and outcomes, but apply a semi-synthetic per-policy log-multiplier (documented in Appendices I and I.1) that forces aggregate elasticities into the $\gamma < \eta$ regime in which UVET's closed-form allocation is well posed. The results below use $\text{MT-LIFT}_{\text{scaled}}$ (Table 1); Appendix I.1 contrasts the raw and isotonic variants.

Results UVET_{bay} buys the most value on $\text{MT-LIFT}_{\text{scaled}}$. Under LP it raises value by 12.5% and beats every other row on the same test resamples, the closest one, the CB ratio, by 4.3 points (Tables 1 and 14). It also spends 1.6% more than the reference on the test week, so our marking rule leaves it unmarked, and the best LP row within budget, Merge (UVET_{cl} + CF_{mse}), gains 1.9%. Pricing the overspend does not change the ranking, since UVET_{bay} keeps the highest S_{comb}. Its only difference from UVET_{cl}, which gains 1.8%, is the shrinkage of the leaf elasticities, so most of the gain comes from shrinkage, and its size depends on the prior strength (Appendix G.6.2). The strongest baseline also pays for its value with extra spend: the CB ratio overspends by 4.0%. With their own mCUV allocator, the two best rows within budget are UVET partitions, and mCUV keeps most of their training gain on the test week, while LP loses half of it or more (Appendix I).

On the raw MT-LIFT release the global elasticities violate $\gamma < \eta$, UVET finds no split, and causal forests with an LP allocator gain more value, at a cost overrun of 3 to 4% (Table 15). This case is easy to detect, since the global elasticities, fit on the whole training split, can be checked before growing any tree.

4.2 UVET-benchmark data

UVET-benchmark[4] is an 86.7M-impression industrial A/B log with five covariates, dual value/cost outcomes, and five ordered bid multipliers $\{0.8, \ldots, 1.2\}$ on the reference policy (Appendix E). The multiplier itself is continuous, and any $\alpha \in [0.5, 1.5]$ is deployable; the five levels are the levels randomized in the A/B test, and offline evaluation is restricted to them. The closed form returns intensities between logged treatment levels, e.g. $\alpha_b^* = 0.879$ for one UVET_{cl} leaf (Table 16).

Results UVET partitions again give the best policies within budget (Table 2). Under LP, $\text{UVET}_{\text{uncl}}$ raises value by 0.95% without a significant overspend, and it and UVET_{bay} are the only rows with a positive S_{comb}. The DR Policy Tree gains slightly more, 1.05%, but overspends by 0.38%, and

[3] https://github.com/MTDJDSP/MT-LIFT

[4] https://huggingface.co/datasets/criteo/uvet-benchmark

Table 1: MT-LIFT$_{\text{scaled}}$ test, depth D=2: $\Delta v_{\mathcal{B}}$, $\Delta c_{\mathcal{B}}$, and S_{comb} (%) under three allocators, mean $\pm$ std over the same B=1000 test resamples for every row (paired comparisons in Table 14). †: over budget, i.e. the 95% bootstrap interval of $\Delta c_{\mathcal{B}}$ lies above zero. **Bold** and underline: highest and second-highest $\Delta v_{\mathcal{B}}$ per allocator among rows without †, and highest and second-highest S_{comb}. S_{comb} prices overspend at $1/\varepsilon_{\text{glob}} = 1.58$; rows that train on `rev` use the shadow-price target of Appendix F.3.

	LP (primary)		ILP		mCUV eq. (native)		Budget-penalized
Method	$\Delta v_{\mathcal{B}}$ (%)	$\Delta c_{\mathcal{B}}$ (%)	$\Delta v_{\mathcal{B}}$ (%)	$\Delta c_{\mathcal{B}}$ (%)	$\Delta v_{\mathcal{B}}$ (%)	$\Delta c_{\mathcal{B}}$ (%)	S_{comb} (%)
Baselines (no UVET partition)							
CF$_{\text{mse}}$	0.0 $\pm$0.6	−3.0 $\pm$0.4	−2.9 $\pm$0.7	−4.9 $\pm$0.4	−0.4 $\pm$0.5	3.2 $\pm$0.2†	−1.46 $\pm$0.62
CF$_{\text{het}}$	4.5 $\pm$1.3	1.6 $\pm$0.6†	0.7 $\pm$1.5	−0.3 $\pm$0.7	3.8 $\pm$0.7	0.6 $\pm$0.3	1.24 $\pm$1.26
DR Policy Tree	−8.0 $\pm$1.1	−5.3 $\pm$0.5	−8.0 $\pm$1.2	−6.0 $\pm$0.6	−0.6 $\pm$0.3	−1.7 $\pm$0.2	−8.01 $\pm$1.13
CART-Rev	−4.6 $\pm$1.0	−3.6 $\pm$0.5	−8.3 $\pm$1.2	−3.7 $\pm$0.6	−2.4 $\pm$0.5	3.5 $\pm$0.3†	−6.45 $\pm$1.08
UMCT (CADR)	−5.9 $\pm$1.4	0.4 $\pm$0.7	−4.0 $\pm$1.1	−0.9 $\pm$0.5	−0.4 $\pm$0.4	0.0 $\pm$0.2	−5.35 $\pm$1.12
LBCF	−2.9 $\pm$0.8	−3.2 $\pm$0.4	−2.8 $\pm$0.7	−3.4 $\pm$0.5	−0.8 $\pm$0.4	2.9 $\pm$0.2†	−2.86 $\pm$0.73
CB ratio	8.2 $\pm$0.8	4.0 $\pm$0.5†	**4.6** $\pm$0.9	−0.7 $\pm$0.5	−2.3 $\pm$0.5	4.4 $\pm$0.3†	3.15 $\pm$0.80
UVET variants							
UVET$_{\text{cl}}$	1.8 $\pm$1.4	1.7 $\pm$0.7†	−1.9 $\pm$1.5	−2.1 $\pm$0.7	8.0 $\pm$0.7	1.6 $\pm$0.3†	−1.41 $\pm$1.29
UVET$_{\text{uncl}}$	1.6 $\pm$1.4	−0.6 $\pm$0.7	0.2 $\pm$1.4	−3.5 $\pm$0.7	**8.3** $\pm$0.6	−0.8 $\pm$0.3	0.86 $\pm$1.32
UVET$_{\text{bay}}$	12.5 $\pm$1.1	1.6 $\pm$0.5†	13.5 $\pm$1.1	4.7 $\pm$0.6†	5.5 $\pm$0.5	−2.0 $\pm$0.2	**7.98** $\pm$1.16
UVET-derived merged partitions							
Merge (UVET$_{\text{cl}}$ + CF$_{\text{mse}}$)	**1.9** $\pm$1.5	−2.5 $\pm$0.6	−0.9 $\pm$1.4	−3.8 $\pm$0.6	0.9 $\pm$0.6	3.6 $\pm$0.3†	0.50 $\pm$1.45
Merge (UVET$_{\text{cl}}$ + CF$_{\text{het}}$)	5.0 $\pm$1.3	1.5 $\pm$0.6†	3.9 $\pm$1.4	−0.3 $\pm$0.6	4.4 $\pm$0.7	0.6 $\pm$0.3†	3.16 $\pm$1.21

the paired bootstrap cannot tell the two apart. It cannot separate UVET$_{\text{uncl}}$ from the CF-only merge either, and it does separate it from the six other baselines (Table 17). When the `rev`-based baselines are retrained with cost priced at $\hat{\gamma}/\hat{\eta}$, the DR Policy Tree drops to zero and the CB ratio becomes the only baseline the test does not separate from UVET$_{\text{uncl}}$ (Table 18). The best row within budget is the merge of UVET$_{\text{cl}}$ with CF$_{\text{het}}$ under ILP and UVET$_{\text{uncl}}$ under mCUV. On the test split the baselines lose most of their small training gains, apart from the DR Policy Tree, while the UVET variants keep theirs (Appendix J).

The leaves show what UVET exploits (Table 16). Both partitions separate a cohort whose value hardly reacts to the bid ($\hat{\gamma}_b \leq 0.04$) from one that reacts several times more strongly ($\hat{\gamma}_b$ of 0.26 and 0.36), and the allocation moves budget from the first to the second at the same total cost. Shrinkage narrows this contrast, and on this dataset its effect on value is small: across λ_0 from 5 to 120, UVET$_{\text{bay}}$ gains between 0.3% and 0.7% under LP (Appendix G.6.2).

Table 2: UVET-benchmark test (21.7M rows), depth D=2: same layout and markers as Table 1, with $1/\varepsilon_{\text{glob}} = 7.31$. Paired comparisons are in Table 17; Table 18 retrains the `rev`-based rows with cost priced at $\hat{\gamma}/\hat{\eta}$. S_{comb} is negative for most rows because its penalty charges about 0.7 points of value for a 0.1% overrun, more than most rows gain.

	LP (primary)		ILP		mCUV eq. (native)		Budget-penalized
Method	$\Delta v_{\mathcal{B}}$ (%)	$\Delta c_{\mathcal{B}}$ (%)	$\Delta v_{\mathcal{B}}$ (%)	$\Delta c_{\mathcal{B}}$ (%)	$\Delta v_{\mathcal{B}}$ (%)	$\Delta c_{\mathcal{B}}$ (%)	S_{comb} (%)
Baselines (no UVET partition)							
CF$_{\text{mse}}$	−0.02 $\pm$0.37	0.31 $\pm$0.20	−0.70 $\pm$0.40	−0.54 $\pm$0.22	0.47 $\pm$0.26	0.06 $\pm$0.13	−1.53 $\pm$0.62
CF$_{\text{het}}$	−0.14 $\pm$0.33	0.22 $\pm$0.19	0.35 $\pm$0.33	−0.76 $\pm$0.16	0.15 $\pm$0.07	0.04 $\pm$0.03	−0.73 $\pm$0.58
Merge (CF$_{\text{mse}}$ + CF$_{\text{het}}$)	0.28 $\pm$0.38	0.36 $\pm$0.22	0.72 $\pm$0.41	−0.14 $\pm$0.23	0.32 $\pm$0.16	0.10 $\pm$0.07	−0.97 $\pm$0.87
DR Policy Tree	1.05 $\pm$0.41	0.38 $\pm$0.18†	0.89 $\pm$0.31	−0.29 $\pm$0.11	0.19 $\pm$0.06	−0.03 $\pm$0.05	−0.44 $\pm$0.62
CART-Rev	0.10 $\pm$0.32	0.22 $\pm$0.14	−0.16 $\pm$0.30	−2.38 $\pm$0.12	0.19 $\pm$0.14	0.06 $\pm$0.06	−0.84 $\pm$0.46
UMCT (CADR)	0.00 $\pm$0.29	−0.17 $\pm$0.13	−0.17 $\pm$0.24	−0.45 $\pm$0.11	0.04 $\pm$0.14	−0.06 $\pm$0.07	−0.11 $\pm$0.25
LBCF	−0.16 $\pm$0.27	0.18 $\pm$0.16	−0.42 $\pm$0.37	−0.68 $\pm$0.22	0.17 $\pm$0.19	0.05 $\pm$0.10	−0.99 $\pm$0.47
CB ratio	−0.15 $\pm$0.40	0.28 $\pm$0.21	0.00 $\pm$0.00	0.00 $\pm$0.00	0.42 $\pm$0.19	−0.09 $\pm$0.08	−1.12 $\pm$0.64
UVET variants							
UVET$_{\text{cl}}$	0.54 $\pm$0.43	0.24 $\pm$0.21	0.53 $\pm$0.35	−0.27 $\pm$0.15	0.94 $\pm$0.22	0.23 $\pm$0.12†	−0.38 $\pm$0.64
UVET$_{\text{uncl}}$	**0.95** $\pm$0.31	0.13 $\pm$0.12	0.76 $\pm$0.33	−0.23 $\pm$0.14	**0.77** $\pm$0.16	0.15 $\pm$0.09	**0.33** $\pm$0.43
UVET$_{\text{bay}}$	0.61 $\pm$0.42	0.13 $\pm$0.20	0.77 $\pm$0.42	−0.62 $\pm$0.21	0.60 $\pm$0.22	0.09 $\pm$0.12	0.09 $\pm$0.55
UVET-derived merged partitions							
Merge (UVET$_{\text{cl}}$ + CF$_{\text{mse}}$)	0.29 $\pm$0.35	0.05 $\pm$0.18	0.22 $\pm$0.40	−0.07 $\pm$0.22	0.58 $\pm$0.18	0.01 $\pm$0.09	−0.31 $\pm$0.68
Merge (UVET$_{\text{cl}}$ + CF$_{\text{het}}$)	0.78 $\pm$0.42	0.26 $\pm$0.21	**1.04** $\pm$0.43	0.18 $\pm$0.22	0.32 $\pm$0.13	0.03 $\pm$0.08	−0.85 $\pm$1.23

5 Discussion

Size of the gains. The gains on UVET-benchmark are an order of magnitude smaller than on MT-LIFT$_{\text{scaled}}$, and the data explain why. On UVET-benchmark the bid multiplier mostly changes cost: from α=0.8 to α=1.2, total cost rises by 57% and total value by only 6% (Appendix E), so $\hat{\gamma}/\hat{\eta}$ is 0.14, against 0.63 on MT-LIFT$_{\text{scaled}}$. Moving budget between cohorts at a fixed total cost then has little room. With leaf value elasticities of at most 0.36, even raising a cohort to the highest logged multiplier lifts its value by at most $1.2^{0.36} - 1 \approx 6.8\%$, and other cohorts have to be lowered to pay for it. Gains of about one percent are what this kind of data allows, for UVET and for the baselines alike.

Cost on the test split. LP and mCUV spend exactly the reference budget on the training split, and ILP at most that amount. On the test split the same allocations drift off budget. On UVET-benchmark the test rows are a random sample of the same two weeks, the drift is sampling noise, and it stays below 0.4% for every row. On MT-LIFT$_{\text{scaled}}$ the test week comes after the training weeks, so the drift also includes the shift between the two periods. It reaches 1.6% for the LP allocation of UVET$_{\text{bay}}$ and 4.7% for its ILP allocation. mCUV overspends by 3 to 4.5% on four of the seven baseline partitions, whose cohorts likely follow a power law in α less closely than UVET leaves do.

Wall time. UVET needs only closed-form steps and a scalar root search per candidate split (Algorithm 1), and our Python implementation fits in two to three times the time of the EconML causal forests (Tables 9 and 11), whose tree learner is compiled.

6 Conclusion

UVET is a greedy tree that scores splits by the marginal cost of a unit of value and assigns each cohort its treatment in closed form under a power-law model. On MT-LIFT$_{\text{scaled}}$, a version of MT-LIFT rescaled into the diminishing-returns regime, UVET$_{\text{bay}}$ raises LP value by 12.5%, against at most 8.2% for the baselines, and its 1.6% overspend still leaves it with the best budget-penalized score (Table 1). On UVET-benchmark, where value reacts weakly to the bid, all gains are about 1% or less, and under each allocator the largest gain within budget comes from a UVET-based partition (Table 2). We release UVET-benchmark so that cost-aware assignment methods can be compared on a large randomized log. The causal forests are built to estimate treatment effects and still trail the best UVET variant on both datasets, in line with the argument of Casacuberta and Hardt (2026) that good allocations do not need accurate effect estimates. UVET suits ordered treatments with diminishing returns, hard budgets and populations of 10^7 units or more, where the policy has to be cheap to score and easy to audit, as with bid multipliers. For unordered treatments, small samples, or budgets loose enough to allow per-unit optimization, a causal forest followed by a knapsack solver is a reasonable alternative (Appendix K.1).

Limitations UVET needs randomized data with at least two treatment levels in every candidate region (Kohavi et al., 2013; Xu et al., 2015), and it assumes that value grows more slowly than cost, $\gamma_b < \eta_b$. The leaf elasticities come from log–log fits on a handful of treatment levels and are noisy in small leaves. Clipping and shrinkage help, but the gain of UVET$_{\text{bay}}$ on MT-LIFT$_{\text{scaled}}$ depends on how strongly we shrink, and validation picks the prior strength with little margin (Appendix G.6.2). The splits are greedy, so the partition has no global optimality guarantee; mixed-integer trees (Jo et al., 2021; Amram et al., 2022) have one but do not scale to tens of millions of rows. On synthetic data, trees deeper than three levels overspend by 4 to 6% on held-out rows (Table 8), and the learned policy stays 4 to 15% below an oracle that knows the true groups, whether or not the responses follow a power law (Appendix H). Our largest gain is on a semi-synthetic variant built to satisfy the assumption. Appendix C.6 discusses sample complexity, identification without randomization, and fairness.

Acknowledgments and Disclosure of Funding

This work was partially supported by the French National Research Agency (ANR) through grants ANR-20-CE23-0007 and ANR-23-CE23-0002 and through the PEPR IA FOUNDRY project

(ANR-23-PEIA-0003). It was also supported by Hi! PARIS and the ANR/France 2030 program (ANR-23-IACL-0005).

References

Ai, M., Li, B., Gong, H., Yu, Q., Xue, S., Zhang, Y., Zhang, Y., and Jiang, P. (2022). LBCF: A large-scale budget-constrained causal forest algorithm. In *Proceedings of the ACM Web Conference 2022*.

Albert, J. and Goldenberg, D. (2022). E-commerce promotions personalization via online multiple-choice knapsack with uplift modeling. In *ACM International Conference on Information & Knowledge Management*.

Amram, M., Dunn, J., and Zhuo, Y. D. (2022). Optimal policy trees. *Machine Learning*, 111(7):2741–2768.

Athey, S. and Imbens, G. (2016). Recursive partitioning for heterogeneous causal effects. *National Academy of Sciences*, 113(27):7353–7360.

Athey, S., Tibshirani, J., and Wager, S. (2019). Generalized random forests. *The Annals of Statistics*.

Balseiro, S. R., Lu, H., and Mirrokni, V. (2023). The best of many worlds: Dual mirror descent for online allocation problems. *Operations Research*, 71(1):101–119.

Battocchi, K., Dillon, E., Hei, M., Lewis, G., and Oka, P., Oprescu, M., and Syrgkanis, V. (2019). EconML: A Python Package for ML-Based Heterogeneous Treatment Effects Estimation. https://github.com/py-why/EconML. Version 0.x.

Bertsimas, D. and Dunn, J. (2017). Optimal classification trees. *Machine Learning*, 106:1039–1082.

Betlei, A., Vladimirova, M., Sebbar, M., Urien, N., Rahier, T., and Heymann, B. (2024). Maximizing the success probability of policy allocations in online systems. *AAAI Conference on Artificial Intelligence*.

Bica, I., Jordon, J., and van der Schaar, M. (2020). Estimating the effects of continuous-valued interventions using generative adversarial networks. In *Advances in Neural Information Processing Systems*, volume 33.

Bockel-Rickermann, C., Vanderschueren, T., Verdonck, T., and Verbeke, W. (2024). Sources of gain: Decomposing performance in conditional average dose response estimation. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*.

Breiman, L. (1984). Classification and regression trees. *The Wadsworth & Brooks/Cole*.

Brock, K., Homer, V., Soul, G., Potter, C., Chiuzan, C., and Lee, S. (2021). Is more better? an analysis of toxicity and response outcomes from dose-finding clinical trials in cancer. *BMC Cancer*, 21(1):777.

Brock, K. and Potter, C. (2019). Dose-level toxicity and efficacy outcomes from dose-finding clinical trials in oncology. University of Birmingham dataset.

Casacuberta, S. and Hardt, M. (2026). Good allocations from bad estimates. *arXiv preprint arXiv:2601.05597*.

Chen, G., Zeng, D., and Kosorok, M. R. (2016). Personalized dose finding using outcome weighted learning. *Journal of the American Statistical Association*, 111(516):1509–1521.

Clauset, A., Shalizi, C. R., and Newman, M. E. (2009). Power-law distributions in empirical data. *SIAM review*, 51(4):661–703.

Cover, T. M. and Thomas, J. A. (2006). *Elements of Information Theory*. Wiley-Interscience, 2nd edition.

De Vos, S., Bockel-Rickermann, C., Lessmann, S., and Verbeke, W. (2026). Uplift modeling with continuous treatments: A predict-then-optimize approach. *European Journal of Operational Research*, 330(1):230–244.

Diemert, E., Betlei, A., Renaudin, C., and Massih-Reza, A. (2018). A large-scale benchmark for uplift modeling. *AdKDD and TargetAd Workshop, KDD*.

Du, S., Lee, J., and Ghaffarizadeh, F. (2019). Improve user retention with causal learning. In *ACM SIGKDD Workshop on Causal Discovery*.

Efron, B. (2012). *Large-Scale Inference: Empirical Bayes Methods for Estimation, Testing, and Prediction*. Cambridge University Press.

Elmachtoub, A. N. and Grigas, P. (2022). Smart "predict, then optimize". *Management Science*, 68(1):9–26.

Fernandez, J., Wehrstedt, L., Shamis, L., Elhoushi, M., Saladi, K., Bisk, Y., Strubell, E., and Kahn, J. (2024). Hardware scaling trends and diminishing returns in large-scale distributed training. *arXiv preprint arXiv:2411.13055*.

Frauen, D., Melnychuk, V., and Feuerriegel, S. (2025). Treatment effect estimation for optimal decision-making. In *Advances in Neural Information Processing Systems*, volume 38.

Gabaix, X. (2009). Power laws in economics and finance. *Annu. Rev. Econ.*, 1(1):255–294.

Goutelle, S., Maurin, M., Rougier, F., Barbaut, X., Bourguignon, L., Ducher, M., and Maire, P. (2008). The Hill equation: a review of its capabilities in pharmacological modelling. *Fundamental & Clinical Pharmacology*, 22(6):633–648.

Gubela, R. M. and Lessmann, S. (2020). Interpretable multiple treatment revenue uplift modeling. In *AMCIS 2020 Proceedings*.

Gubela, R. M., Lessmann, S., and Stöcker, B. (2024). Multiple treatment modeling for target marketing campaigns: A large-scale benchmark study. *Information Systems Frontiers*, 26(3):875–898.

Hillstrom, K. (2008). The minethatdata e-mail analytics and data mining challenge. Dataset description page.

Hirano, K. and Imbens, G. W. (2004). The propensity score with continuous treatments. *Applied Bayesian Modeling and Causal Inference from Incomplete-Data Perspectives*, pages 73–84.

Hoffmann, J., Borgeaud, S., Mensch, A., Buchatskaya, E., Cai, T., Rutherford, E., Casas, D. d. L., Hendricks, L. A., Welbl, J., Clark, A., et al. (2022). Training compute-optimal large language models. *Advances in Neural Information Processing Systems*, 35:30016–30030.

Huang, Y., Wang, S., Gao, M., Wei, X., Li, C., Luo, C., Zhu, Y., Xiao, X., and Luo, Y. (2024a). Entire chain uplift modeling with context-enhanced learning for intelligent marketing. In *Companion Proceedings of the ACM on Web Conference 2024*, pages 226–234.

Huang, Y., Wang, S., Gao, M., Wei, X., Li, C., Luo, C., Zhu, Y., Xiao, X., and Luo, Y. (2024b). MT-LIFT: The real-world production dataset from e-commerce platform Meituan. Dataset page.

International Energy Agency (2025). Energy and AI. IEA Report.

International Warfarin Pharmacogenetics Consortium (2009). Estimation of the warfarin dose with clinical and pharmacogenetic data. *New England Journal of Medicine*, 360(8):753–764.

Jo, N., Aghaei, S., Gómez, A., and Vayanos, P. (2021). Learning optimal prescriptive trees from observational data. *arXiv preprint arXiv:2108.13628*.

Kamran, F., Makar, M., and Wiens, J. (2024). Learning to rank for optimal treatment allocation under resource constraints. In *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*, volume 238 of *Proceedings of Machine Learning Research*, pages 3727–3735. PMLR.

Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., Gray, S., Radford, A., Wu, J., and Amodei, D. (2020). Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*.

Kennedy, E. H. (2023). Towards optimal doubly robust estimation of heterogeneous causal effects. *Electronic Journal of Statistics*, 17(2):3008–3049.

Kilts Center for Marketing, University of Chicago Booth School of Business (2026). Dominick's dataset. Dataset page. Accessed 2026-04-21.

Kohavi, R., Deng, A., Frasca, B., Walker, T., Xu, Y., and Pohlmann, N. (2013). Online controlled experiments at large scale. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Künzel, S. R., Sekhon, J. S., Bickel, P. J., and Yu, B. (2019). Metalearners for estimating heterogeneous treatment effects using machine learning. *National Academy of Sciences*, 116(10):4156–4165.

Labandeira, X., Labeaga, J. M., and López-Otero, X. (2017). A meta-analysis on the price elasticity of energy demand. *Energy Policy*, 102:549–568.

Liu, D., Tang, X., Qiao, Y., Liu, M., Sun, Z., He, X., and Ming, Z. (2024). Benchmarking for deep uplift modeling in online marketing. In *Proceedings of the ACM Web Conference*.

Lu, C.-P. (2025). The race to efficiency: A new perspective on AI scaling laws. *arXiv preprint arXiv:2501.02156*.

Makhijani, R., Chakrabarti, S., Struble, D., and Liu, Y. (2019). Lore: a large-scale offer recommendation engine with eligibility and capacity constraints. In *Proceedings of the 13th ACM Conference on Recommender Systems*, pages 160–168.

Mandi, J., Kotary, J., Berden, S., Mulamba, M., Bucarey, V., Guns, T., and Fioretto, F. (2023). Decision-focused learning: Foundations, state of the art, benchmark and future opportunities. *arXiv preprint arXiv:2307.13565*.

Nie, L., Ye, M., Liu, Q., and Nicolae, D. (2021). VCNet and functional targeted regularization for learning causal effects of continuous treatments. In *International Conference on Learning Representations*.

Olaya, D., Coussement, K., and Verbeke, W. (2020). A survey and benchmarking study of multi-treatment uplift modeling. *Data Mining and Knowledge Discovery*, 34(2):273–308.

Open Data Science (ODS.ai) (2019). X5 retail hero: Uplift modeling for promotional campaign. Competition page.

Patriksson, M. (2008). A survey on the continuous nonlinear resource allocation problem. *European Journal of Operational Research*, 185(1):1–46.

Pellatt, D. F. (2022). PAC-Bayesian treatment allocation under budget constraints. *arXiv preprint arXiv:2212.09007*.

Phillips, R. L. (2021). *Pricing and revenue optimization*. Stanford university press.

Radcliffe, N. J. and Surry, P. D. (2007). Using control groups to target on predicted lift: Building and assessing uplift models. In *Direct Marketing Analytics Journal*, volume 1, pages 14–21.

Rubin, D. B. (1974). Estimating causal effects of treatments in randomized and nonrandomized studies. *Journal of educational Psychology*, 66(5):688.

Rzepakowski, P. and Jaroszewicz, S. (2012). Decision trees for uplift modeling with single and multiple treatments. *Knowledge and Information Systems*, 32:303–327.

Schwab, P., Linber, L., Bauer, S., Buhmann, J. M., and Karlen, W. (2020). Learning counterfactual representations for estimating individual dose-response curves. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 5612–5619.

Schwartz, R., Dodge, J., Smith, N. A., and Etzioni, O. (2020). Green AI. *Communications of the ACM*, 63(12):54–63.

scikit-uplift (2026a). Lenta uplift modeling dataset documentation. Documentation page. Accessed 2026-04-21.

scikit-uplift (2026b). Megafon uplift competition dataset documentation. Documentation page. Accessed 2026-04-21.

Sethuraman, R., Tellis, G. J., and Briesch, R. A. (2011). How well does advertising work? generalizations from meta-analysis of brand advertising elasticities. *Journal of Marketing Research*, 48(3):457–471.

Shephard, R. W. and Färe, R. (1974). The law of diminishing returns. In *Production Theory: Proceedings of an International Seminar Held at the University at Karlsruhe May–July 1973*, pages 287–318. Springer.

Sołtys, M., Jaroszewicz, S., and Rzepakowski, P. (2015). Ensemble methods for uplift modeling. *Data Mining and Knowledge Discovery*, 29(6):1531–1559.

Strubell, E., Ganesh, A., and McCallum, A. (2019). Energy and policy considerations for deep learning in NLP. In *Annual Meeting of the Association for Computational Linguistics*, pages 3645–3650.

Sun, H., Munro, E., Kalashnov, G., Du, S., and Wager, S. (2021). Treatment allocation under uncertain costs. *arXiv preprint arXiv:2103.11066*.

Tu, Y., Basu, K., DiCiccio, C., Bansal, R., Nandy, P., Jaikumar, P., and Chatterjee, S. (2021). Personalized treatment selection using causal heterogeneity. In *ACM Web Conference*.

Varian, H. R. (1992). *Microeconomic Analysis*. W. W. Norton & Company, 3rd edition.

Wager, S. and Athey, S. (2018). Estimation and inference of heterogeneous treatment effects using random forests. *Journal of the American Statistical Association*, 113(523):1228–1242.

Wu, H., Tan, S., Li, W., Garrard, M., Obeng, A., Dimmery, D., Singh, S., Wang, H., Jiang, D., and Bakshy, E. (2022). Interpretable personalized experimentation. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 4173–4183.

Xu, Y., Chen, N., Fernandez, A., Sinno, O., and Bhasin, A. (2015). From infrastructure to culture: A/B testing challenges in large-scale social networks. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Yan, Z., Wang, S., Zhou, G., Lin, J., and Jiang, P. (2023). An end-to-end framework for marketing effectiveness optimization under budget constraint. *arXiv preprint arXiv:2302.04477*.

Zhang, W., Yuan, S., Wang, J., and Shen, X. (2014). Real-time bidding benchmarking with ipinyou dataset. *CoRR*, abs/1407.7073.

Zhao, K., Hua, J., Yan, L., Zhang, Q., Xu, H., and Yang, C. (2019). A unified framework for marketing budget allocation. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1820–1830.

Zhou, H., Li, S., Jiang, G., Zheng, J., and Wang, D. (2023a). Direct heterogeneous causal learning for resource allocation problems in marketing. *AAAI Conference on Artificial Intelligence*.

Zhou, Z., Athey, S., and Wager, S. (2023b). Offline multi-action policy learning: Generalization and optimization. *Operations Research*, 71(1):148–183.

A Related Work

We organize the literature into five streams, highlighting where UVET sits relative to each.

Heuristic partitioning methods Before model-driven approaches, practitioners typically segment users via hand-crafted rules: quantile-based splits on a chosen feature (Betlei et al., 2024), iso-cost buckets (fixing equal cost per group), or feature-similarity clustering (Makhijani et al., 2019). Du et al. (2019) and Tu et al. (2021) discuss such heuristics in advertising and coupon targeting. These methods ignore treatment-effect heterogeneity entirely and thus cannot exploit differential response; however, they remain widely deployed due to their simplicity and interpretability. UVET inherits the tree structure familiar to practitioners while replacing heuristic splits with a criterion that directly optimizes downstream allocation quality.

Heterogeneous treatment effect estimation Causal trees (Athey and Imbens, 2016) and causal forests (Athey et al., 2019; Wager and Athey, 2018) partition the covariate space to estimate conditional average treatment effects (CATE). Meta-learners (S-, T-, and X-learner (Künzel et al., 2019)) and doubly robust methods (Kennedy, 2023) provide flexible CATE estimators. When treatment is continuous, generalized propensity score methods (Hirano and Imbens, 2004) and neural dose-response estimators such as DRNet (Schwab et al., 2020), SCIGAN (Bica et al., 2020), and VCNet (Nie et al., 2021) learn the full individualized dose-response curve nonparametrically. All of these methods target prediction accuracy for $\tau(x)$ or $m(t, x)$ but do not solve the downstream allocation; a separate optimization step is required to map effect estimates to treatment assignments. UVET sidesteps individualized dose-response estimation: the power-law assumption reduces the response surface to two elasticity parameters per cohort, and the splitting criterion directly optimizes allocation quality.

Budget-constrained treatment assignment Sun et al. (2021) characterize the optimal budget-constrained policy as thresholding on $\tau_v(x)/\tau_c(x)$, establishing the theoretical foundation for cost-aware assignment. Ai et al. (2022) operationalize this via Lagrangian bisection on causal forest scores (LBCF), deployed at scale in e-commerce. Tu et al. (2021) cast the problem as a stochastic knapsack after bucketing users with causal trees. All these methods are two-stage: estimate effects first, then solve allocation. Casacuberta and Hardt (2026) show that this two-stage paradigm is wasteful in sample complexity: coarse CATE estimates already suffice for near-optimal allocations, requiring $O(M/\epsilon)$ rather than $O(M/\epsilon^2)$ samples. Their result shows that precise effect estimation is not necessary for good treatment assignment. When the per-unit objective is concave and separable, classical results in operations research show the allocation reduces to equalizing marginal returns, solvable in linear time (Patriksson, 2008). UVET exploits this structure: the power-law assumption makes the per-cohort problem concave-separable, so mCUV equalization yields a closed form up to a scalar root search for μ, with no integer program at inference time. By embedding this allocation into the tree-splitting criterion, UVET integrates estimation and optimization in a single pass.

Prescriptive and policy trees Optimal prescriptive trees (Bertsimas and Dunn, 2017; Jo et al., 2021) use mixed-integer optimization (MIO) to find globally optimal partitions with treatment assignments; they come with strong guarantees but become intractable for $> 10^4$ observations, and optimal policy trees (Amram et al., 2022) scale further through coordinate-descent search. Depth-limited policy trees (Zhou et al., 2023b) search exactly over shallow trees to optimize doubly-robust welfare, and Wu et al. (2022), who grow one binary causal tree per treatment and merge them by intersecting the resulting partitions $\mathcal{B}^{(j)}$ across treatments $j = 1, \ldots, J$ to obtain a final partition $\{B^{(1)}_{i_1} \cap \cdots \cap B^{(J)}_{i_J} \neq \emptyset\}$. Kamran et al. (2024) propose a tree-based method that directly optimizes CATE rankings for treatment allocation under resource constraints, showing that ranking-oriented splitting criteria can outperform accuracy-oriented ones. UVET shares this decision-focused philosophy but targets a different setting: continuous treatment with dual value–cost outcomes, where the power-law structure yields a closed-form allocation per cohort rather than a ranking.

Decision-focused learning Rather than separating prediction from optimization, decision-focused learning (DFL) (Mandi et al., 2023; Elmachtoub and Grigas, 2022) trains models to minimize downstream decision loss. Frauen et al. (2025) prove that state-of-the-art CATE estimators, while optimal for estimation accuracy, can be suboptimal for downstream decision-making, and propose retargeting the learning objective to balance both. De Vos et al. (2026) take a predict-then-optimize route to continuous treatment assignment, with ILP-based allocation. Zhou et al. (2023a) and Betlei et al. (2024) reformulate the optimization criteria to integrate allocation into the learning phase. UVET can be viewed as a domain-specific DFL method: the mCUV splitting criterion is a loss function

Table 3: Method families vs. assignment desiderata (abbreviated columns). *End-to-end*: training targets a partition directly tied to the downstream welfare map (our LP/ILP/mCUV evaluation protocol is orthogonal to this column). *Large-scale*: families commonly deployed at 10^5–10^7+ rows with standard tooling, often with inner subsampling; qualitative only. Middle rows include the baselines of Tables 1 and 2 (Section 4, Appendix F.4).

Method	Problem	Treatment type	Applicable to Large-Scale Data	Incorporate Constraints	End-to-end	Open-source and Reproducible
Causal Tree (CT) (Athey and Imbens, 2016)	HTE	Binary	✓	✗	✗	✓
Causal Forest (CF) (Athey et al., 2019; Wager and Athey, 2018)	HTE	Multiple	✓	✗	✗	✓
MTrees (Tu et al., 2021)	HTE for TA	Multiple	✓	✗	✗	✗
LBCF (Ai et al., 2022)	HTE for TA	Multiple	✓	✓	✗	✗
Distill-HTE (merged CTs) (Wu et al., 2022)	HTE for TA	Multiple	✓	✗	✗	✗
CB ratio prioritization (Sun et al., 2021)	HTE for TA	Multiple	✓	✓	✗	✓
DR policy tree (honest tree on DR scores) (Battocchi et al., 2019)	TA	Multiple	✓	✗	✓	✓
UMCT (CADR) (De Vos et al., 2026)	TA	Continuous	✓	✓	✗	✓
Optimal Prescriptive Trees (Jo et al., 2021)	TA	Multiple	✗	✓	✓	✗
Optimal Policy Trees (Amram et al., 2022)	TA	Any	✓	✗	✓	✗
CAIPWL (Zhou et al., 2023b)	TA	Multiple	✗	✗	✓	✓
UVET (ours)	TA	Continuous or discretized	✓	✓	✓	✓

derived from the closed-form allocation, so the partitioning directly optimizes decision quality. The difference from generic DFL is that UVET's power-law assumption eliminates the need for differentiable optimization layers or ILP solvers, making it particularly suited for large-scale settings ($> 10^7$ observations).

A.1 Baseline Methods Comparison

Table 3 situates UVET among representative HTE estimators, HTE-for-assignment pipelines, and prescriptive-tree methods; the preceding paragraphs in Section A give the narrative, while the middle block lists families that appear in the main score tables (Section 4, Appendix F.4). We additionally time CAIPWL on a fixed synthetic draw of $3 \cdot 10^5$ rows (Table 9) without adding it to Tables 1 and 2. UVET remains the only row that combines continuous (or discretized) ordered treatment, explicit budget handling in the split objective, $> 10^7$ scalability in our released pipeline, and end-to-end partitioning with closed-form per-cohort allocation.

B Benchmark Landscape and Positioning of *UVET-benchmark*

Ordered interventions with diminishing returns are common across marketing, medicine, and energy allocation (Section 1), yet no public benchmark simultaneously provides five ingredients needed to evaluate personalized cost-aware assignment: (i) individual covariates, (ii) ordered or continuous treatment, (iii) randomized support, (iv) dual value–cost outcomes, and (v) sufficient scale for subgroup discovery. Drop any one and the evaluation is incomplete: without dual outcomes one cannot test budget feasibility; without ordered treatment one cannot estimate dose-response curves; without randomization one must rely on unverifiable identification assumptions.

Recent work confirms that this gap persists. Bockel-Rickermann et al. (2024) decompose estimator performance on four popular dose-response benchmarks and show that confounding, the challenge most estimators target, is not actually present in any of them; the benchmarks are hard for other, often undisclosed, reasons. Liu et al. (2024) audit 13 deep uplift models on two industrial datasets and find that differences between methods are smaller than reported, partly because evaluation protocols lack standardization. De Vos et al. (2026) study continuous-treatment uplift and still rely on semi-synthetic evaluation, while real-world multiple-treatment targeting studies with continuous outcomes remain proprietary (Gubela and Lessmann, 2020; Gubela et al., 2024). The problem is not new: Sołtys et al. (2015) flagged the scarcity of public uplift datasets with true control groups over a decade ago, and Olaya et al. (2020) documented that commonly used datasets were never designed for individual treatment-effect estimation. Yet the gap remains unresolved. Current benchmarks steer research toward binary treatment or synthetic dose-response settings, even where the applied problems are broader.

Table 4 surveys the main public datasets adjacent to our setting. The large-scale display-ad uplift benchmark of Diemert et al. (2018) is randomized and large but binary-treatment. Hillstrom (Hill-

Dataset	Setting	Treatment structure	Outcomes	Gap w.r.t. cost-aware ordered-treatment eval.
Diemert et al. (Diemert et al., 2018)	display advertising	binary treat/control	visit + conversion	Only two treatment levels; no ordered-dose evaluation.
Hillstrom (Hillstrom, 2008)	e-mail marketing	2 e-mail campaigns + control	visit + conversion + spend	Categorical treatments, not ordered intensity; sparse conversion (Sołtys et al., 2015).
Lenta (scikit-uplift, 2026a)	retail targeting	binary contact/control	single behavioral target	Binary treatment, single outcome.
X5 RetailHero (Open Data Science (ODS.ai), 2019)	retail targeting	binary communication/control	purchase target	Binary treatment, single outcome.
MegaFon (scikit-uplift, 2026b)	telecom uplift	binary treatment/control	single target	Synthetic; no ordered-dose semantics.
MT-LIFT (Huang et al., 2024a,b)	coupon recommendation	5 coupon labels	click + conversion	Coupon identities, not calibrated scalar dose.
Dominick's (Kilts Center for Marketing, University of Chicago Booth School of Business, 2026)	pricing	randomized price variation	sales / quantity	Store–week level, not individual assignment.
iPinYou (Zhang et al., 2014)	real-time bidding	bid values in auction logs	click + conversion + bids	Observational, not randomized.
`dosefindingdata` (Brock and Potter, 2019; Brock et al., 2021)	oncology dose finding	ordered dose levels	toxicity + efficacy	Aggregated by dose/trial; no patient covariates.
Warfarin / IWPC (International Warfarin Pharmacogenetics Consortium, 2009; Chen et al., 2016)	personalized dosing	continuous dose	INR-related endpoint	Observational; single outcome.
Retailer campaigns (Gubela and Lessmann, 2020; Gubela et al., 2024)	target marketing	multiple discounts	continuous outcomes	Proprietary; not a public benchmark.
UVET-benchmark	online advertising	ordered / continuous policy scaling	value + cost	All five ingredients: randomized, individual, ordered, dual outcomes, large scale.

Table 4: Public datasets near our setting; each existing dataset lacks at least one ingredient for large-scale cost-aware ordered-treatment evaluation (Section B).

strom, 2008) has three policies that represent campaign categories, not ordered intensities, and prior work evaluates visits because conversions are sparse (Sołtys et al., 2015). Lenta (scikit-uplift, 2026a), X5 (Open Data Science (ODS.ai), 2019), and MegaFon (scikit-uplift, 2026b) provide binary treatment and a single outcome. MT-LIFT (Huang et al., 2024a,b) comes closest: it is randomized, large-scale, and exposes five coupon treatments with click and conversion labels. Its released treatment, however, indexes coupon identities rather than a calibrated scalar intensity, so an ordered-treatment interpretation requires a preprocessing step. In pricing, Dominick's (Kilts Center for Marketing, University of Chicago Booth School of Business, 2026) offers randomized price variation at store–week granularity, not individual level. iPinYou (Zhang et al., 2014) logs bids, clicks, and conversions from real auctions but without randomized assignment. In medicine, the Birmingham dosefindingdata (Brock and Potter, 2019; Brock et al., 2021) contains ordered dose levels with joint toxicity/efficacy outcomes, yet aggregated at the dose/trial level with no patient covariates. Open warfarin data (International Warfarin Pharmacogenetics Consortium, 2009; Chen et al., 2016) provide patient-level covariates and continuous dose but are observational and expose a single endpoint.

UVET-benchmark fills this gap by jointly providing randomized support, individual-level covariates, an ordered continuous treatment parameter, dual value–cost outcomes, and 86.7 M impressions, enough to test tree-based partitioning for heterogeneous subgroup discovery under the regime that is common in practice but absent from existing public benchmarks.

Relation to limitations stated in prior work Each piece of the five-ingredient gap has been flagged independently in prior work (binary-only datasets (Sołtys et al., 2015), non-causal benchmarks (Olaya et al., 2020), proprietary multi-treatment data (Gubela and Lessmann, 2020; Gubela et al., 2024), flawed continuous-dose benchmarks (Bockel-Rickermann et al., 2024), reliance on semi-synthetic evaluation (De Vos et al., 2026)). *UVET-benchmark* addresses their conjunction.

C Theoretical Properties of UVET

C.1 Leaf-level treatment rule and KKT optimality

Proof of Proposition 1. We restate the power-law assumption: for $\alpha \in A$,

$$c_b(\alpha) = c_b(1)\,\alpha^{\eta_b}, \qquad v_b(\alpha) = v_b(1)\,\alpha^{\gamma_b}. \tag{13}$$

Eliminating α gives the following relationship between value and cost inside bucket b:

$$v_b\big(c_b\big) \;=\; v_b(1)\left(\tfrac{c_b}{c_b(1)}\right)^{\gamma_b/\eta_b},$$

where (with a slight abuse of notation) we now treat v_b as a function of the realized cost. Writing $\epsilon_b := \gamma_b/\eta_b$, the cost-value relationship is increasing and strictly concave when $\epsilon_b \in (0,1)$, i.e. $0 < \gamma_b < \eta_b$, which we assume throughout.

Cost of Unit Value (CUV). For $b \in \mathcal{B}$ and $\alpha \in A$, we define the (average) cost of unit value as

$$CUV_b(\alpha) = \frac{c_b(\alpha)}{v_b(\alpha)} \;=\; CUV_b(1)\,\alpha^{\eta_b-\gamma_b}, \tag{14}$$

where $c_b(1)$ and $v_b(1)$ are the observed cost and value totals of bucket b under the reference treatment α=1, so $CUV_b(1) = c_b(1)/v_b(1)$ is read directly from the data, without counterfactual estimation of treatment effects.

Marginal Cost of Unit Value (mCUV). We define the marginal cost of unit value as $mCUV_b(\alpha) = \partial c_b(\alpha)/\partial v_b(\alpha)$, which under (13) simplifies to

$$mCUV_b(\alpha) \;=\; \frac{c_b(1)\,\eta_b}{v_b(1)\,\gamma_b}\,\alpha^{\eta_b-\gamma_b} \;=\; \frac{CUV_b(1)}{\epsilon_b}\,\alpha^{\eta_b-\gamma_b}. \tag{15}$$

In particular, $mCUV_b(1) = CUV_b(1)/\epsilon_b$, and computing $mCUV_b(\alpha)$ requires only the reference quantities $c_b(1), v_b(1)$ together with elasticity estimates $\hat{\gamma}_b, \hat{\eta}_b$ obtained by log-log OLS (Appendix C.4).

Equalization. We seek a treatment assignment $(\alpha_b)_{b\in\mathcal{B}}$ such that the marginal cost of unit value is constant across buckets. Denoting this common value by $\mu > 0$, we require, for all $b \in \mathcal{B}$,

$$mCUV_b(\alpha_b) = \mu, \tag{16}$$

together with the iso-cost constraint

$$\sum_{b\in\mathcal{B}} c_b(\alpha_b) \;=\; C \;=\; \sum_{b\in\mathcal{B}} c_b(1), \tag{17}$$

where the second equality follows from the definition $C = \sum_b c_b(1)$ as the total reference cost. Substituting (16) into (13) via $\alpha_b = (\mu/mCUV_b(1))^{1/(\eta_b-\gamma_b)}$ yields the equivalent implicit equation for μ (see Proposition 3 for existence and uniqueness):

$$\sum_{b\in\mathcal{B}} c_b(1)\left(\tfrac{\mu}{mCUV_b(1)}\right)^{1/(1-\epsilon_b)} \;=\; C.$$

Closed form for α_b^*. Solving (16) for α_b using (15):

$$\frac{CUV_b(1)}{\epsilon_b}\,(\alpha_b)^{\eta_b-\gamma_b} = \mu \quad\Longrightarrow\quad \alpha_b^*(\mu) \;=\; \left(\tfrac{\mu\,\epsilon_b}{CUV_b(1)}\right)^{1/(\eta_b-\gamma_b)} \;=\; \left(\tfrac{\mu}{mCUV_b(1)}\right)^{1/(\eta_b-\gamma_b)},$$

which matches (8) in the main text and concludes the proof. □

Proof of Proposition 2. We verify KKT conditions step by step.

Step 1 (form the Lagrangian). Define

$$\mathcal{L}\big((\alpha_b)_{b\in\mathcal{B}},\lambda\big) = \sum_{b\in\mathcal{B}} v_b(\alpha_b) \;+\; \lambda\left(C - \sum_{b\in\mathcal{B}} c_b(\alpha_b)\right),$$

Step 2 (first-order stationarity). For each bucket b,

$$\frac{\partial \mathcal{L}}{\partial \alpha_b} = \frac{\partial v_b}{\partial \alpha}(\alpha_b) - \lambda \frac{\partial c_b}{\partial \alpha}(\alpha_b) = 0$$
$$\iff \quad \lambda = \frac{\frac{\partial v_b}{\partial \alpha}(\alpha_b)}{\frac{\partial c_b}{\partial \alpha}(\alpha_b)} = \frac{1}{\mathrm{mCUV}_b(\alpha_b)}.$$

Thus stationarity implies $\mathrm{mCUV}_b(\alpha_b) = 1/\lambda$ is constant across all buckets. Let $\mu := 1/\lambda$. Then the KKT stationarity conditions are equivalent to

$$\forall b \in \mathcal{B}, \quad \mathrm{mCUV}_b(\alpha_b) = \mu,$$

i.e., marginal cost per unit value equalization (Eq. (6)).

Step 3 (solve stationarity using Proposition 1). By Proposition 1, the unique α_b satisfying $\mathrm{mCUV}_b(\alpha_b) = \mu$ is exactly $\alpha_b^*(\mu)$ in (8).

Step 4 (primal feasibility). Choosing μ such that (9) holds ensures the constraint $\sum_b c_b(\alpha_b) = C$ is satisfied.

Step 5 (global optimality). Under $\gamma_b < \eta_b$, the induced mapping from cost to value inside each bucket is strictly concave (see Proposition 5 below), and the objective in (10) is concave in the cost allocation with a linear constraint. Therefore, any feasible point satisfying the KKT conditions is globally optimal. Hence $\alpha^*(\mu)$ solves (10). □

C.2 Uniqueness of μ and monotonicity properties

Proposition 3 (Existence and uniqueness of μ (implicit in Eq. (9))). *Assume $\eta_b > \gamma_b$ for all $b \in \mathcal{B}$ and $c_b(1) > 0$. Then the equation $\sum_b c_b(\alpha_b^*(\mu)) = C$ admits a unique solution $\mu > 0$.*

Proof. Using $c_b(\alpha) = c_b(1)\alpha^{\eta_b}$ and (8),

$$c_b(\alpha_b^*(\mu)) = c_b(1) \left(\frac{\mu}{\mathrm{mCUV}_b(1)} \right)^{\frac{\eta_b}{\eta_b - \gamma_b}}.$$

Define

$$F(\mu) := \sum_{b \in \mathcal{B}} c_b(1) \left(\frac{\mu}{\mathrm{mCUV}_b(1)} \right)^{\frac{\eta_b}{\eta_b - \gamma_b}}.$$

Step 1 (continuity). Each term is continuous in $\mu > 0$, hence F is continuous on $(0, \infty)$.

Step 2 (strict monotonicity). For each b, the exponent $\frac{\eta_b}{\eta_b - \gamma_b} > 0$ since $\eta_b > \gamma_b$. Thus each term is strictly increasing in μ, hence F is strictly increasing.

Step 3 (limits). As $\mu \downarrow 0$, each term tends to 0, so $F(\mu) \to 0$. As $\mu \uparrow \infty$, each term diverges to ∞, so $F(\mu) \to \infty$.

Step 4 (existence and uniqueness). By continuity and intermediate value theorem, there exists μ such that $F(\mu) = C$. By strict monotonicity, this solution is unique. □

Proposition 4 (Monotonicity of α_b^* in μ and in mCUV). *For any bucket b with $\eta_b > \gamma_b$, the map $\mu \mapsto \alpha_b^*(\mu)$ is strictly increasing and the map $\mathrm{mCUV}_b(1) \mapsto \alpha_b^*(\mu)$ is strictly decreasing.*

Proof. From (8),

$$\alpha_b^*(\mu) = (\mu)^{\frac{1}{\eta_b - \gamma_b}} \left(\mathrm{mCUV}_b(1)\right)^{-\frac{1}{\eta_b - \gamma_b}}.$$

Since $\eta_b - \gamma_b > 0$, the exponent $\frac{1}{\eta_b - \gamma_b}$ is positive, implying $\alpha_b^*(\mu)$ increases with μ and decreases with $\mathrm{mCUV}_b(1)$. □

C.3 Concavity (diminishing returns) and its consequence

Proposition 5 (Concavity of value as a function of cost in each bucket)**.** *Assume* (4) *with* $\gamma_b < \eta_b$ *and* $c_b(1), v_b(1) > 0$*. Then* v_b *as a function of* c_b *is strictly concave on* $(c_b(\alpha_{\min}), c_b(\alpha_{\max}))$*.*

Proof. We eliminate α. From $c_b(\alpha) = c_b(1)\alpha^{\eta_b}$ we get

$$\alpha = \left(\frac{c_b}{c_b(1)}\right)^{\frac{1}{\eta_b}}.$$

Plugging into $v_b(\alpha) = v_b(1)\alpha^{\gamma_b}$ yields

$$v_b(c_b) = v_b(1)\left(\frac{c_b}{c_b(1)}\right)^{\frac{\gamma_b}{\eta_b}} = K\, c_b^{\epsilon_b}, \quad \text{where} \quad K := \frac{v_b(1)}{c_b(1)^{\epsilon_b}},\ \ \epsilon_b := \frac{\gamma_b}{\eta_b}.$$

Since $0 < \gamma_b < \eta_b$, we have $\epsilon_b \in (0, 1)$. Compute derivatives:

$$\frac{dv_b}{dc_b} = K\epsilon_b\, c_b^{\epsilon_b - 1}, \qquad \frac{d^2 v_b}{dc_b^2} = K\epsilon_b(\epsilon_b - 1)\, c_b^{\epsilon_b - 2}.$$

Because $K > 0$, $\epsilon_b > 0$, and $\epsilon_b - 1 < 0$, we have $\frac{d^2 v_b}{dc_b^2} < 0$ for all $c_b > 0$, proving strict concavity. □

C.4 Consistency of elasticity estimation (log–log regression)

Proposition 6 (Consistency of bucket-level elasticity estimates)**.** *Fix a bucket* b *and suppose data in* b *are generated by*

$$v_{bi} = v_b(1)\alpha_{bi}^{\gamma_b}\exp(\xi_{bi}), \qquad c_{bi} = c_b(1)\alpha_{bi}^{\eta_b}\exp(\zeta_{bi}),$$

where ξ_{bi}, ζ_{bi} *are i.i.d. with* $\mathbb{E}[\xi_{bi}] = \mathbb{E}[\zeta_{bi}] = 0$ *and finite second moments, independent of* α_{bi}*, and where* α_{bi} *takes at least two distinct values with non-zero probability in bucket* b *(as ensured by randomized assignment). Let* $\hat{\gamma}_b$ *and* $\hat{\eta}_b$ *be obtained by ordinary least squares in the log–log regressions*

$$\log v_{bi} = \beta_v + \gamma\, \log \alpha_{bi} + \varepsilon_{bi}, \qquad \log c_{bi} = \beta_c + \eta\, \log \alpha_{bi} + \nu_{bi}.$$

Then, as the number of observations N_b *in bucket* b *grows, with the share of every treatment level bounded away from zero,* $\hat{\gamma}_b \to \gamma_b$ *and* $\hat{\eta}_b \to \eta_b$ *in probability.*

Proof. We prove the value case; the cost case is identical.

Step 1 (linear model after log transform). Taking logs yields

$$\log v_{bi} = \log v_b(1) + \gamma_b \log \alpha_{bi} + \xi_{bi}.$$

This is a standard linear regression with regressor $x_i := \log \alpha_{bi}$ and noise ξ_{bi}.

Step 2 (exogeneity). By assumption, ξ_{bi} is independent of α_{bi}, hence $\mathbb{E}[\xi_{bi}\, x_i] = 0$.

Step 3 (non-degenerate design). Because α_{bi} takes at least two distinct values with non-zero probability in bucket b, we have $\mathrm{Var}(x_i) > 0$.

Step 4 (OLS consistency). Under finite second moments, exogeneity, and $\mathrm{Var}(x_i) > 0$, the OLS slope estimator is consistent by standard LLN/continuous mapping arguments for linear regression. Therefore $\hat{\gamma}_b \to \gamma_b$ in probability. □

Remark (why this matches the implementation) In the algorithm, elasticities are estimated by linear regression on the logarithm of aggregated bucket-level outcomes across treatment policies. The same argument applies to the aggregated setting when each policy contains enough samples, since sample averages converge to their expectations and the regression noise remains mean-zero under randomized assignment.

C.5 Local Robustness to Elasticity Misspecification in Log-Space

In this section we formalize a local robustness guarantee: if the true bucket-level response functions deviate from the power-law model by at most δ in log-space (both in levels and log-slopes), then the allocation induced by the power-law solution remains near-optimal, with allocation error and value loss scaling as $O(\delta)$.

Misspecification model Fix a partition $\mathcal{B}$ and nominal (power-law) bucket functions

$$v_b^0(\alpha) = v_b(1)\alpha^{\gamma_b}, \qquad c_b^0(\alpha) = c_b(1)\alpha^{\eta_b}, \qquad \gamma_b < \eta_b.$$

Let $\tilde{v}_b(\alpha)$ and $\tilde{c}_b(\alpha)$ denote the *true* value and cost functions used to generate data. We assume a uniform *log-space* deviation bound: for all buckets $b \in \mathcal{B}$ and all $\alpha \in [\alpha_{\min}, \alpha_{\max}]$,

$$\left|\log \tilde{v}_b(\alpha) - \log v_b^0(\alpha)\right| \le \delta, \qquad \left|\log \tilde{c}_b(\alpha) - \log c_b^0(\alpha)\right| \le \delta, \tag{18}$$

$$\left|\frac{\partial}{\partial \log \alpha} \log \tilde{v}_b(\alpha) - \gamma_b\right| \le \delta, \qquad \left|\frac{\partial}{\partial \log \alpha} \log \tilde{c}_b(\alpha) - \eta_b\right| \le \delta. \tag{19}$$

The level bounds (18) control multiplicative misspecification, while the slope bounds (19) ensure that *instantaneous elasticities* remain within δ of the nominal exponents.

Policies Let $\alpha^{*,0} = (\alpha_b^{*,0})_{b\in\mathcal{B}}$ denote the power-law allocation computed by Eqs. (8)–(10) (i.e., $\mathrm{mCUV}_b^0(\alpha_b^{*,0}) = \mu^0$ for all b, and $\sum_b \tilde{c}_b(\alpha_b^{*,0}) = C$ enforced by choosing μ^0 accordingly). UVET itself calibrates μ^0 with the nominal costs through (9); under (18) its true spend then lies in $[e^{-\delta}C, e^{\delta}C]$, so the theorem describes UVET up to this $O(\delta)$ deviation from the budget. Let $\alpha^\star = (\alpha_b^\star)_{b\in\mathcal{B}}$ denote the oracle allocation that solves the true constrained problem

$$\max_{\alpha_b \in [\alpha_{\min}, \alpha_{\max}]} \sum_b \tilde{v}_b(\alpha_b) \quad \text{s.t.} \quad \sum_b \tilde{c}_b(\alpha_b) = C.$$

Theorem 1 (Local robustness bound in log-space)**.** *Assume (18)–(19) hold and define $\kappa_b := \eta_b - \gamma_b$ and $\kappa_{\min} := \min_{b\in\mathcal{B}} \kappa_b > 0$. Assume further that the oracle solution is interior (i.e., $\alpha_b^\star \in (\alpha_{\min}, \alpha_{\max})$ for all b), so KKT stationarity holds with equality, and that each true $\widetilde{\mathrm{mCUV}}_b$ is strictly increasing in α on $[\alpha_{\min}, \alpha_{\max}]$.*

Then there exist constants $K_\alpha, K_V > 0$ depending only on $\{\gamma_b, \eta_b\}_{b\in\mathcal{B}}$, $\alpha_{\min}, \alpha_{\max}$, and $|\mathcal{B}|$ (but not on δ), such that for all sufficiently small δ:

$$\max_{b\in\mathcal{B}} \left|\log \alpha_b^{*,0} - \log \alpha_b^\star\right| \le \frac{K_\alpha}{\kappa_{\min}}\,\delta, \tag{20}$$

$$0 \le \sum_b \tilde{v}_b(\alpha_b^\star) - \sum_b \tilde{v}_b(\alpha_b^{*,0}) \le K_V\,\delta \cdot \sum_b \tilde{v}_b(\alpha_b^\star). \tag{21}$$

In particular, the relative value loss under the true functions is $O(\delta)$.

Proof. We verify the bound step by step.

Step 1 (rewrite true functions as multiplicative perturbations). From (18), define perturbations $\varepsilon_{v,b}(\alpha)$ and $\varepsilon_{c,b}(\alpha)$ such that

$$\tilde{v}_b(\alpha) = v_b^0(\alpha)\exp(\varepsilon_{v,b}(\alpha)), \qquad \tilde{c}_b(\alpha) = c_b^0(\alpha)\exp(\varepsilon_{c,b}(\alpha)),$$

with $|\varepsilon_{v,b}(\alpha)| \le \delta$ and $|\varepsilon_{c,b}(\alpha)| \le \delta$ for all b, α.

Step 2 (control true elasticities). By (19), define log-slope deviations

$$\Delta_{v,b}(\alpha) := \frac{\partial}{\partial \log \alpha} \log \tilde{v}_b(\alpha) - \gamma_b, \qquad \Delta_{c,b}(\alpha) := \frac{\partial}{\partial \log \alpha} \log \tilde{c}_b(\alpha) - \eta_b,$$

so that $|\Delta_{v,b}(\alpha)| \le \delta$ and $|\Delta_{c,b}(\alpha)| \le \delta$.

Step 3 (control the true mCUV relative to the nominal one). Recall $\mathrm{mCUV}_b(\alpha) = \frac{\partial c_b/\partial \alpha}{\partial v_b/\partial \alpha}$. Using the identity

$$\frac{\partial \tilde{v}_b}{\partial \alpha} = \tilde{v}_b(\alpha) \cdot \frac{1}{\alpha} \cdot \frac{\partial}{\partial \log \alpha} \log \tilde{v}_b(\alpha),$$

(and analogously for $\tilde{c}_b$), we obtain

$$\widetilde{\mathrm{mCUV}}_b(\alpha) = \frac{\frac{\partial \tilde{c}_b}{\partial \alpha}}{\frac{\partial \tilde{v}_b}{\partial \alpha}} = \frac{\tilde{c}_b(\alpha)}{\tilde{v}_b(\alpha)} \cdot \frac{\frac{\partial}{\partial \log \alpha} \log \tilde{c}_b(\alpha)}{\frac{\partial}{\partial \log \alpha} \log \tilde{v}_b(\alpha)}$$
$$= \underbrace{\frac{c_b^0(\alpha)}{v_b^0(\alpha)} \cdot \frac{\eta_b}{\gamma_b}}_{=\mathrm{mCUV}_b^0(\alpha)} \cdot \exp\big(\varepsilon_{c,b}(\alpha) - \varepsilon_{v,b}(\alpha)\big) \cdot \frac{\eta_b + \Delta_{c,b}(\alpha)}{\gamma_b + \Delta_{v,b}(\alpha)}.$$

For sufficiently small δ (e.g., $\delta \le \min_b \gamma_b/2$), the ratio $\frac{\eta_b+\Delta_{c,b}}{\gamma_b+\Delta_{v,b}}$ is well-defined and satisfies

$$\left| \log \frac{\eta_b + \Delta_{c,b}(\alpha)}{\gamma_b + \Delta_{v,b}(\alpha)} - \log \frac{\eta_b}{\gamma_b} \right| \le C_1\, \delta$$

for a constant C_1 depending only on $\min_b \gamma_b$ and $\max_b \eta_b$. Together with $|\varepsilon_{c,b} - \varepsilon_{v,b}| \le 2\delta$, we conclude there exists $C_2 > 0$ such that

$$\big|\log \widetilde{\mathrm{mCUV}}_b(\alpha) - \log \mathrm{mCUV}_b^0(\alpha)\big| \le C_2\, \delta \qquad \forall b, \alpha. \tag{22}$$

Step 4 (KKT equalization for the true optimum). Since the oracle solution is interior, KKT stationarity implies there exists $\tilde{\mu} > 0$ such that

$$\widetilde{\mathrm{mCUV}}_b(\alpha_b^\star) = \tilde{\mu} \qquad \forall b \in \mathcal{B}. \tag{23}$$

Similarly, the nominal allocation α^0 is computed by equalizing the nominal mCUV: there exists $\mu^0 > 0$ such that

$$\mathrm{mCUV}_b^0(\alpha_b^{*,0}) = \mu^0 \qquad \forall b \in \mathcal{B}. \tag{24}$$

Step 5 (compare μ^0 and $\tilde{\mu}$ via feasibility). Define the total-cost maps under the two allocations:

$$F^\star(\tilde{\mu}) := \sum_b \tilde{c}_b(\alpha_b^\star(\tilde{\mu})), \qquad F^0(\mu^0) := \sum_b \tilde{c}_b(\alpha_b^{*,0}(\mu^0)),$$

where $\alpha_b^\star(\cdot)$ denotes the inverse of (23) and $\alpha_b^{*,0}(\cdot)$ the inverse of (24). Both maps are strictly increasing in their argument on the interior because $\tilde{c}_b$ is strictly increasing in α and $\widetilde{\mathrm{mCUV}}_b$ is strictly increasing in α by assumption. By construction both satisfy the same constraint $F^\star(\tilde{\mu}) = C$ and $F^0(\mu^0) = C$. Therefore, it suffices to show that the maps $F^\star$ and F^0 are close when their arguments are close.

Using (22) and the explicit nominal form

$$\log \mathrm{mCUV}_b^0(\alpha) = \log \mathrm{mCUV}_b^0(1) + \kappa_b \log \alpha,$$

standard inverse-function stability yields that the inverse mappings satisfy

$$\big|\log \alpha_b^{*,0}(\mu) - \log \alpha_b^\star(\mu)\big| \le \frac{C_3}{\kappa_b}\, \delta \tag{25}$$

uniformly over μ in the feasible range, for some constant $C_3 > 0$ depending only on the bounded interval $[\alpha_{\min}, \alpha_{\max}]$ and the nominal parameters. Applying (25) at the respective solutions μ^0 and $\tilde{\mu}$, and using monotonicity of the cost maps, we obtain

$$|\log \mu^0 - \log \tilde{\mu}| \le C_4\, \delta \tag{26}$$

for a constant $C_4 > 0$ (depending only on $\mathcal{B}$, α-bounds, and nominal parameters).

Step 6 (allocation error bound). Combine (24), (23), (22), and (26):

$$\log \mathrm{mCUV}_b^0(\alpha_b^{*,0}) = \log \mu^0,$$
$$\log \mathrm{mCUV}_b^0(\alpha_b^\star) = \log \widetilde{\mathrm{mCUV}}_b(\alpha_b^\star) + O(\delta) = \log \tilde{\mu} + O(\delta).$$

Subtracting and using $\log \mathrm{mCUV}_b^0(\alpha) = \log \mathrm{mCUV}_b^0(1) + \kappa_b \log \alpha$ gives

$$\kappa_b\big(\log \alpha_b^{*,0} - \log \alpha_b^\star\big) = \big(\log \mu^0 - \log \tilde{\mu}\big) + O(\delta) = O(\delta).$$

Thus

$$\left|\log \alpha_b^{*,0} - \log \alpha_b^\star\right| \le \frac{K_\alpha}{\kappa_b}\,\delta \le \frac{K_\alpha}{\kappa_{\min}}\,\delta,$$

which proves (20).

Step 7 (value loss bound). By (19), the function $\log \tilde{v}_b(\alpha)$ is Lipschitz in $\log \alpha$ with constant at most $\gamma_b + \delta$ on $[\alpha_{\min}, \alpha_{\max}]$:

$$\left|\log \tilde{v}_b(\alpha) - \log \tilde{v}_b(\alpha')\right| \le (\gamma_b + \delta)\left|\log \alpha - \log \alpha'\right|.$$

Applying this with $\alpha = \alpha_b^{*,0}$ and $\alpha' = \alpha_b^\star$ and using (20) yields

$$\left|\log \tilde{v}_b(\alpha_b^{*,0}) - \log \tilde{v}_b(\alpha_b^\star)\right| \le (\gamma_b + \delta) \cdot \frac{K_\alpha}{\kappa_{\min}}\,\delta \le C_5\,\delta,$$

for a constant C_5 independent of δ (for δ small). Exponentiating gives

$$\tilde{v}_b(\alpha_b^{*,0}) \ge \tilde{v}_b(\alpha_b^\star)\, e^{-C_5\delta} \ge \tilde{v}_b(\alpha_b^\star)\,(1 - C_5\delta),$$

where the last inequality uses $e^{-x} \ge 1 - x$ for $x \ge 0$. Summing over buckets:

$$\sum_b \tilde{v}_b(\alpha_b^{*,0}) \ge (1 - C_5\delta) \sum_b \tilde{v}_b(\alpha_b^\star),$$

which rearranges to (21) with $K_V := C_5$. □

Discussion Theorem 1 is consistent with the small value gaps of Appendix H, although it does not bound the cost violations reported there: when the true response curves are close to power laws in log-space (both in level and instantaneous elasticity), the induced mCUV functions are uniformly close, implying that (i) the equal-mCUV allocation changes only by $O(\delta)$ in log-treatment space, and (ii) the resulting value loss is $O(\delta)$ relative to the oracle optimum under the true functions.

C.6 Discussion of theoretical scope and open questions

We collect several remarks on the theoretical scope of UVET that may guide future work.

Greedy splitting vs. global optimality UVET uses recursive greedy partitioning (Algorithm 1), which does not guarantee a globally optimal tree. The partitioning baselines we compare against (causal forests, DR Policy Tree, CART-Rev, LBCF, the CB ratio) also split greedily; Proposition 2 covers only the allocation within a fixed partition. Mixed-integer optimization (MIO) formulations (Bertsimas and Dunn, 2017; Jo et al., 2021) find the optimal tree but scale as $O(2^{|\mathcal{B}|})$, becoming impractical beyond about 10^4 observations. UVET's greedy strategy with closed-form per-cohort allocation runs in $O(n \cdot d \cdot T \cdot D)$ where n is the number of samples, d the number of features, T the number of treatment levels, and D the tree depth. On UVET-benchmark (43.4M training rows), a depth-3 tree builds in under 10 minutes on a single CPU, orders of magnitude faster than MIO. On the same hardware, Table 9 reports a *fixed-n synthetic* micro-benchmark ($n_{\text{train}} = n_{\text{test}} = 3{\cdot}10^5$) where wall times for the partitioners of Tables 1 and 2 (partition fit plus LP/ILP/mCUV evaluation with a single bootstrap replicate for timing) span roughly 0.08–1.0 s once every baseline uses the full training draw, with CAIPWL (Zhou et al., 2023b) at ∼0.99 s because its cross-fitted doubly-robust stage fits a propensity model plus per-policy outcome regressors on all units; causal forests and UVET remain below 0.3 s, the UVET-derived merge (intersection of two depth-2 partitions) sits near 0.55 s, and LBCF together with the CB ratio baseline cluster near 0.9 s once pairwise (LBCF) or dual global (CB) forests run on all units. This table answers "who is cheap *per fit* at identical n?"; it is not rank-preserving for absolute wall times on full real datasets, where I/O and sample size dominate (Appendix F.6). We have not measured the greedy gap, since mixed-integer trees do not run at our scale.

Sample complexity UVET's allocation quality depends on accurate elasticity estimation via log-log regression. With T treatment levels and N_b observations in bucket b, the OLS estimators $\hat{\gamma}_b, \hat{\eta}_b$ converge at rate $O(1/\sqrt{N_b})$ under standard regularity conditions. The minimum leaf size therefore controls the bias-variance tradeoff: too small, and the elasticity estimates are noisy; too large, and the tree cannot capture heterogeneity. We set the minimum leaf size to $n/2^{D+2}$ (where D is depth),

so each leaf holds at least $n/2^{D+2}$ samples, a quarter of the average occupancy at depth D. Pellatt (2022) provide PAC-Bayes bounds for tree-based allocation that could be adapted to UVET's power-law setting. Relatedly, Casacuberta and Hardt (2026) show that treatment allocation generically requires fewer samples than effect estimation ($O(M/\epsilon)$ vs. $O(M/\epsilon^2)$), suggesting that UVET's coarse two-parameter model per cohort may be statistically sufficient; formalizing this connection is future work.

Unconfoundedness assumption UVET assumes access to randomized (A/B test) data so that the log-log regression identifies causal elasticities. This is stated informally in the problem formulation but deserves emphasis: without randomization, the OLS estimators of γ_b and η_b are biased by confounding. Extending UVET to observational data, for example via inverse propensity weighting of the log-log regression or instrumental variables when treatment assignment is endogenous, is a natural direction. Sun et al. (2021) formalize the unconfoundedness requirement for budget-constrained policies; their framework applies directly to our setting.

Fairness considerations The mCUV equalization principle has an implicit fairness property: at the optimum, every cohort receives the same marginal return per unit of cost (Proposition 1). Explicit fairness constraints, such as requiring that treatment intensity does not correlate with protected attributes, can be incorporated by adding linear constraints to the Lagrangian, though this may break the closed-form solution. Jo et al. (2021) handle fairness constraints in tree-based allocation and De Vos et al. (2026) in predict-then-optimize allocation; adapting their approach to UVET is straightforward but increases computational cost.

D Synthetic example when UVET outperforms HTE-based approaches

Synthetic setup We consider a single dimensional feature variable X, with $x \sim \mathcal{U}(0,1)$. We introduce two thresholds $t_1, t_2 \in (0,1)$ which define three buckets $\{b_1, b_2, b_3\}$. We consider our data results of applying a randomized control trial with two treatments: $\alpha_0 = 1$ and $\alpha_1 = 1.5$. We design the elasticity (power-law) parameters in each bucket so that the only proper split that should be learned from a value optimization perspective is threshold t_2.

Formally, we set $\{t_1, t_2\} = \{0.1, 0.5\}$, and fix the following values for the power-law parameters and reference (cost, value) per bucket:

$$\begin{array}{llll} v_{b_1}(1) = 6.5, & c_{b_1}(1) = 0.4, & \gamma_{b_1} = 1, & \eta_{b_1} = 1.5 \\ v_{b_2}(1) = 1.63, & c_{b_2}(1) = 0.1, & \gamma_{b_2} = 1, & \eta_{b_2} = 1.5 \\ v_{b_3}(1) = 1.19, & c_{b_3}(1) = 0.1, & \gamma_{b_3} = 1, & \eta_{b_3} = 1.5 \end{array}$$

The marginal cost of unit value ($mCUV$s), optimal μ, values of α^* all have a closed-form expressions with respect to those parameters. In this setting they are equal to:

$$\begin{array}{ll} mCUV(b_1) \approx mCUV(b_2) \approx 0.092, & mCUV(b_3) \approx 0.126 \\ \mu \approx 0.096, & \alpha^*(b_1) \approx \alpha^*(b_2) \approx 1.08, \quad \alpha^*(b_3) \approx 0.58 \end{array}$$

Intuition behind the setup We therefore see that the theoretical optimal treatment parameter α^* are the same for buckets b_1 and b_2, and therefore the same for the whole region $\{x < 0.5\}$, while it is different for bucket b_3 *i.e.* for the region $\{x > 0.5\}$. This implies that the single theoretically optimal split should be made at threshold $x = t_2$. However, the value and cost quantities are significantly larger in bucket b_1 which makes the causal forests below, fit on the scalar proxy y, prefer threshold t_1 because it better separates heterogeneous treatment effects. This split is not the best one for treatment assignment here, so UVET, which scores splits by allocated value, outperforms them on this example. The closed-form values above are unclipped, as in UVET$_{\text{uncl}}$; $\alpha^*(b_3)$ lies below the logged levels.

Data We generate 600,000 points with these parameters, 200,000 per bucket with each treatment level on half of them, and split them at random into training and test halves of 300,000 rows. For the outcome of CF we use $y = v - \frac{\eta}{\gamma} c$.

Results. Figure 1 plots the UVET criterion and the closed-form intensities of the two children against the threshold of a single split. Both causal forests split at $t_1 = 0.1$, while UVET splits at $t_2 = 0.5$, where its criterion peaks.

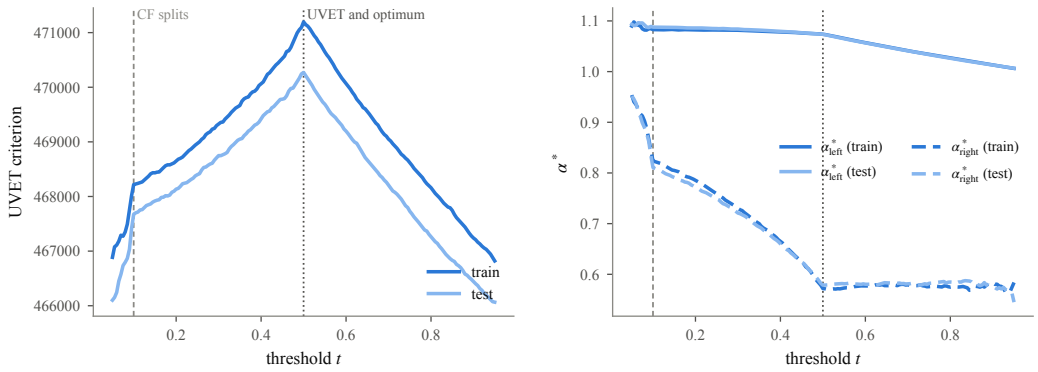


Figure 1: Synthetic example, one split of x on the training (dark) and test (light) split: UVET criterion (left) and closed-form intensities of the two children (right) against the threshold t. Dotted line: $t_2 = 0.5$, the split UVET chooses; dashed line: $t_1 = 0.1$, the split both causal forests choose.

Table 5 reports $\Delta v_{\mathcal{B}}$ and $\Delta c_{\mathcal{B}}$ under mCUV equalization, with intensities in $[0.001, 1.999]$ since $\alpha^*(b_3)$ lies below the logged levels; test values use the full test split. UVET reaches two to three times the value uplift of the forests on both splits. Cost is exactly on budget on the training split and moves by less than 0.1% on the test split, which holds slightly different numbers of rows per bucket and treatment level.

Table 5: Synthetic example (mCUV equalization): train and test relative uplifts $\Delta v_{\mathcal{B}}$ and $\Delta c_{\mathcal{B}}$ (%) for CF_{het}, CF_{mse}, and $\text{UVET}_{\text{uncl}}$, each with one split; test values on the full test split. Highest $\Delta v_{\mathcal{B}}$ per split in bold.

Split	Metric	CF_{het}	CF_{mse}	$\text{UVET}_{\text{uncl}}$
Train	$\Delta v_{\mathcal{B}}$ (%)	0.39	0.39	**1.02**
	$\Delta c_{\mathcal{B}}$ (%)	0.00	0.00	0.00
Test	$\Delta v_{\mathcal{B}}$ (%)	0.49	0.49	**0.94**
	$\Delta c_{\mathcal{B}}$ (%)	0.07	0.07	−0.09

E UVET-benchmark Dataset Documentation

We introduce *UVET-benchmark* under `CC BY-NC-SA 4.0` license, a public dataset collected by an online advertising platform that conducted a two-week randomized A/B test with five different treatments. The dataset is designed to support research on offline policy learning and treatment assignment under cost constraints.

Overview and intended task The dataset contains 86.7M samples and each sample represents an ad impression[5] for which a multi-dimensional context $x \in \mathbb{R}^5$ is observed, along with a randomly assigned treatment parameter α and observed outcomes (v, c). Particular care has been taken to guarantee that each sample (x, α, v, c) is independent. No sequential or panel structure is assumed by default.

The learning goal is to infer a policy $\psi : \mathcal{X} \to A$ that assigns a (continuous) treatment parameter and improves expected value compared to the reference treatment $\alpha = 1$, while controlling cost.

The *UVET-benchmark* dataset is not suitable for estimating real monetary ROI.

[5] An impression corresponds to a single ad auction (e.g., one userpageviewtime tuple). Filtering excluded invalid auctions (no bid, no auction winner, missing telemetry).

Treatments A treatment with parameter α corresponds to uniformly multiplying the bid values of the reference treatment t_0. Therefore, the treatment family can be written as $\mathcal{T} = \{\alpha \cdot t_0,\ \alpha \in A\}$ where $A = [0.5, 1.5]$.

The logged treatment parameters are $\{0.8, 0.9, 1.0, 1.1, 1.2\}$, where $\alpha = 1.0$ is the reference treatment. In our offline evaluation, the LP and ILP allocators assign the logged treatment levels, and the mCUV allocator assigns a continuous α clipped to the logged range. Because treatment assignment is randomized, the logged data support unbiased offline evaluation of treatment effects under standard causal assumptions (no interference, consistency).

In the dataset, the treatment is stored in a single column (`policy`, `float64` type).

Dataset details (schema) The dataset contains the following fields:

- **Features:** five continuous features `f0`–`f4`, with no missing values.
- **Treatment:** continuous `policy` (the treatment parameter α).
- **Outcomes:** `value` and `spend` (cost).

The dataset card on Hugging Face lists every column with its type and description, the splits, the treatment levels, and the license.

Splits The rows of the two-week test are split at random: 75% form the training file, in which a binary flag `is_val` marks a random third of the rows as the validation subset used in the paper experiments, and 25% form the test file.

Privacy and anonymization The original dataset does not include any data that directly identifies a user, such as names, postal addresses, or email addresses in plain text, so the original data is pseudonymized. Continuous features are manually selected from online interaction signals, perturbed with noise, and standardized, so that re-identification is not possible because original scales are not provided. The `value` and `cost` outcomes are scaled and perturbed to preserve business confidentiality. All released outcomes preserve ordering and relative comparisons but not absolute magnitudes. No interpretation in real currency units is possible.

Treatment levels, covariates, and outcomes Table 6 reports, for each logged treatment level, its share of rows and the totals and per-row means of value and cost; Table 7 summarizes the five covariates and the two outcomes on the training split. Every row carries positive value, and cost is zero on fewer than 0.01% of rows; the shares of rows per treatment level agree across train, validation, and test to within 0.1 points. The four non-reference levels received equal shares of the test traffic and the reference level a larger share. Rows are won impressions, so among the non-reference levels the share of rows rises from 8.5% at α=0.8 to 11.3% at α=1.2 while value per row falls and cost per row rises, consistent with higher bids winning more, and more expensive, auctions. The reference level holds 60.2% of rows, with value and cost per row about six times lower than at the other levels. Every estimator in this paper (log–log elasticity fits, LP, ILP, and mCUV) works from per-level *totals* within a cohort, which compares levels at equal traffic if the reference outcomes are scaled to the traffic of one level. The totals are consistent with that scaling: they rise smoothly across all five treatment levels, the reference included, from 957 to 1014 in value and from 211.0k to 330.5k in cost on the training split, which gives global elasticities $\hat{\gamma} = 0.15$ and $\hat{\eta} = 1.11$ ($1/\varepsilon_{\text{glob}} = 7.31$). Per-row means are not comparable across treatment levels and should not be used to fit elasticities on this dataset. The covariates are standardized and right-skewed, with a long upper tail; outcomes are heavy-tailed, with the 99th percentile of cost about 14 times its mean.

Known limitations. We report the following limitations:

- Outcomes (`value`, `cost`) are scaled and perturbed, limiting absolute interpretability.
- Although treatments are modeled as continuous, only five discrete treatment parameters are observed in the logged data.
- Offline evaluation assumes independence of samples and stability of treatment effects across time.
- The power-law elasticity assumption may not hold uniformly across all contexts.

Table 6: *UVET-benchmark* per logged treatment level: share of rows in the train and test splits, value and cost totals on train (cost in thousands), and per-row means on train. Outcomes are in the released (scaled and perturbed) units.

α	train rows (%)	test rows (%)	value total	cost total (k)	value / row	cost / row
0.8	8.5	8.5	957	211.0	2.6×10^{-4}	0.06
0.9	9.2	9.2	973	240.3	2.4×10^{-4}	0.06
1.0	60.2	60.2	994	271.2	3.8×10^{-5}	0.01
1.1	10.7	10.7	1009	299.8	2.2×10^{-4}	0.06
1.2	11.3	11.4	1014	330.5	2.1×10^{-4}	0.07

Table 7: *UVET-benchmark* training split (43.4M rows): summary statistics of the covariates and outcomes. Quantiles of value and cost are over positive entries (cost is zero on fewer than 0.01% of rows).

column	mean	std	min	q_{25}	median	q_{75}	q_{99}	max
`f0`	-3.1×10^{-4}	1.00	-0.50	-0.45	-0.33	0.04	4.12	48.68
`f1`	-2.6×10^{-3}	1.00	-0.90	-0.65	-0.35	0.25	4.06	11.28
`f2`	-1.2×10^{-3}	0.99	-0.20	-0.20	-0.17	-0.08	3.02	266.14
`f3`	-4.6×10^{-4}	0.99	-0.32	-0.29	-0.22	-0.05	3.37	124.15
`f4`	6.7×10^{-4}	1.00	-0.54	-0.44	-0.32	-1.0×10^{-3}	4.68	48.76
`value`	1.1×10^{-4}	9.4×10^{-4}	9.0×10^{-10}	3.2×10^{-6}	1.5×10^{-5}	6.2×10^{-5}	1.6×10^{-3}	2.37
`cost`	0.03	0.12	0	1.1×10^{-3}	5.1×10^{-3}	0.02	0.43	5.41

F Reproducibility Details

This appendix documents the experimental protocols, hyperparameters, and computational budgets sufficient to reproduce every number in the main paper. The code is released at https://github.com/criteo-research/uvet under the Apache 2.0 license, with a guide that maps every table and figure to the command that produces it; this appendix describes the procedures rather than file names.

Bayesian λ_0 sweep and validation choice (Appendix G.6.2). Two scripts in the code repository reproduce this appendix: one refits UVET_{bay} for every λ_0 and leaf floor and scores it on the shared test resamples, and one fits each λ_0 on training rows and scores it on the validation rows. Each writes its results to a new dated folder.

F.1 UVET algorithm hyperparameters

The UVET tree-builder takes the following hyperparameters (Algorithm 1):

- **Maximum depth** D: total number of binary splits along any root-to-leaf path. Tables 1 and 2 fit every partitioner at $D{=}2$ (Section 4); Appendix F.5 varies D on synthetic data.
- **Minimum samples per leaf**: fixed to $\lfloor |\mathbf{X}|/2^{D+2} \rfloor$, ensuring an average leaf occupancy of at least four times the smallest admissible leaf. This rule scales with dataset size and depth, so no per-dataset tuning is needed.
- **Minimum samples to split**: fixed to $\lfloor |\mathbf{X}|/2^{D+1} \rfloor$ (twice the leaf bound).
- **Treatment bounds** $[\alpha_{\min}, \alpha_{\max}]$: clipping interval applied to the leaf-level closed-form allocation (8). For each dataset we set these to the minimum and maximum treatment levels observed in the randomized policies (clipped variant, UVET_{cl}) or, for the unclipped variant $\text{UVET}_{\text{uncl}}$, to $[0.001, 1.999]$ when scoring splits, which leaves the split criterion effectively unconstrained. In every table, the mCUV allocator evaluates all partitions, $\text{UVET}_{\text{uncl}}$ included, with α_b^* clipped to the logged range. The Bayesian variant UVET_{bay} uses the clipped interval and additionally regularizes elasticity estimates as described in Appendix K.
- **Elasticity estimator**: log-log OLS of bucket-aggregated value and cost on $\log \alpha$ across the observed policies (Section 3); the slopes are not shrunk except in UVET_{bay} (eq. (12)).

- **Splitting criterion**: marginal cost equalization gain (Section 3); candidate splits are axis-aligned thresholds on a percentile grid between the 25th and 75th percentiles of each feature with a 1-percentile step (≈ 50 candidates), scanned with the unimodal search of Appendix F.6.

We never tune these knobs on test data. The synthetic depth sweep in Appendix F.5 shows diminishing returns beyond depth 3.

F.2 Example on building a tree with cost constraint

Cost constraint. To allow leaves at different depths (i.e. to support an odd number of buckets), we must ensure conservation of the total cost C used to estimate the splits.

Example Consider a tree with a parent $\{1\}$, first-level children $\{2, 3\}$ and leaves $\{4, 5, 6, 7\}$. In particular, $2 \leftarrow 1 \rightarrow 3$, and both 2 and 3 have child leaves $4 \leftarrow 2 \rightarrow 5$, $6 \leftarrow 3 \rightarrow 7$. We should have: $c_4(\alpha_4^*) + c_5(\alpha_5^*) = C_2$, $c_2(\alpha_2^*) + c_3(\alpha_3^*) = C_1$, and $c_3(\alpha_3^*) + c_4(\alpha_4^*) + c_5(\alpha_5^*) = C_1$. As a result we should have: $C_2 = c_2(\alpha_2^*)$. Each split therefore uses the parent's optimal cost as its constraint; the root uses the total cost of the samples under the reference policy ($\alpha = 1$).

F.3 Evaluation protocol (`rev` construction)

Scalar outcome construction (Causal Forest, CBR). Causal Forest baselines (CF_{mse}, CF_{het}) require a single regression outcome Y. We materialize it as the column `rev` in the preprocessing pipeline. On MT-LIFT and UVET-benchmark, $\texttt{rev}_i = v_i - \hat{\lambda}\, c_i$ with $\hat{\lambda} = (\hat{\gamma}/\hat{\eta})\, \hat{V}_{\text{ref}}/\hat{C}_{\text{ref}}$, where $\hat{\gamma}, \hat{\eta}$ are the *global* log–log elasticities and $\hat{V}_{\text{ref}}, \hat{C}_{\text{ref}}$ the reference-treatment value and cost totals of the training split (one price shared by all units). Under the global power law, $\hat{\lambda}$ is the marginal value of a unit of cost at the reference allocation, $dV/dC = (\gamma/\eta)\, V/C$, i.e. the Lagrange multiplier of the budget constraint in (1) at the reference, so `rev` prices cost in units of value. On UVET-benchmark cost totals are about 270 times value totals and $\hat{\lambda} = 5.0 \times 10^{-4}$. Table 18 also reports the `rev`-based baselines with the price set to $\hat{\gamma}/\hat{\eta}$ alone (0.137 on UVET-benchmark), which ignores the relative scale of value and cost and makes `rev` nearly proportional to $-c$ there. For the CB ratio (CBR) baseline, one forest estimates heterogeneous "uplift" in `rev` while a second estimates uplift in raw cost c, and units are ranked by the ratio of the two predicted uplifts. The synthetic benchmark loader uses a different algebraic convention for `rev` (historical compatibility with the toy split); all methods on that split still read the same `rev` column.

Scope. Train/validation/test splits, the fixed depth D=2, bootstrap confidence intervals (B=1000 shared test resamples for Tables 1 and 2), rules for discarding degenerate candidate splits, and the three-allocator evaluation stack are documented once in Appendix F.4 together with the experiment matrix; the LP/ILP/mCUV allocators are described in Appendix G.4.

F.4 Experiment matrix, seeds, and protocol

Datasets. We use three datasets. (i) *Synthetic* power-law and robustness generators (Section H): bucket-level value/cost responses follow known closed-form laws with multiplicative log-normal noise; covariates and latent groups are sampled from fixed distributions and seeded. (ii) *MT-LIFT* (Huang et al., 2024b): public marketing experiment with five coupon treatments recoded as ordered intensities $\{0.8, \ldots, 1.2\}$, ∼5.5M instances, click and conversion outcomes (Appendix I). (iii) *UVET-benchmark* (this paper): industrial A/B test logs, 86.7M impressions, five covariates, ordered treatment grid, dual value/cost outcomes (Appendix J).

Train/validation/test splits. For UVET-benchmark we use the released split, identical across methods: the rows of the two-week test are divided at random into train (43.4M rows), validation (21.7M rows) and test (21.7M rows) (Appendix E). For MT-LIFT we use the official train and test partitions from the public release. For synthetic data we sample independent draws for train and test with seed offsets $\{42, 1042\}$ respectively.

Model selection, bootstrap, and table conventions. Tables 1 and 2 fit every tree-based partitioner once at depth $D{=}2$ on the training split and report metrics on the held-out test split; on MT-LIFT the official test split is used, identically for every method. The budget-penalized score of Appendix G.6 averages LP and ILP only (**mCUV is excluded**), so it does not favour UVET's native allocator. For every real-data table cell, we bootstrap the test units with replacement at the unit level (policy assignment preserved) and report the mean $\pm$ one standard deviation across replicates; on UVET-benchmark we additionally bootstrap the validation split when forming the depth objective so the selected D is stable across replicates. Tables 1 and 2 use $B{=}1000$: every method is refit once under the protocol above and evaluated with the train-fitted allocations on the same resamples (replicate i drawn with seed $20260801{+}i$), so differences between rows can be bootstrapped in pairs (Table 17). At $B{=}10$ the bootstrap mean on UVET-benchmark carries a Monte Carlo error of about ± 0.11 points, the same order as the differences between rows, and interval endpoints stabilize by $B{\approx}1000$; a check at $B{=}2000$ moves no paired interval endpoint by more than 0.04 points. A dagger in Tables 1 and 2 marks a $\Delta c_{\mathcal{B}}$ whose 95% percentile interval over the $B{=}1000$ resamples lies above zero, so overruns within resampling noise are left unmarked. Bold entries mark, per allocator, the highest mean $\Delta v_{\mathcal{B}}$ among rows without a dagger, underlined entries the second highest, and the same two marks go to the highest and second-highest S_{comb}.

Degenerate candidate splits. If a candidate split produces a child with fewer than the minimum-leaf threshold or with identically zero value mass under the reference policy, we discard the split and continue scanning; if no admissible split exists, the node becomes a leaf. This rule is shared across all tree-based methods.

Method grid. Tables 1 and 2 use the same methods on both benchmarks: *CF$_{mse}$* and *CF$_{het}$* (Battocchi et al., 2019; Wager and Athey, 2018) on the cost-priced proxy outcome (Appendix F.3); the UVET-derived merges Merge (UVET$_{\text{cl}}$ + CF$_{\text{mse}}$) and Merge (UVET$_{\text{cl}}$ + CF$_{\text{het}}$) and the baseline Merge (CF$_{\text{mse}}$ + CF$_{\text{het}}$); *LBCF* (Ai et al., 2022); the *CB ratio* ranking baseline of Sun et al. (2021); and three UVET variants (*UVET*$_{\text{cl}}$, *UVET*$_{\text{uncl}}$, *UVET*$_{\text{bay}}$). The main text reports scores under this grid (Tables 1–2); here we give the exact grid and implementation details.

Splitting objectives and implementation details. Tables 1 (MT-LIFT$_{\text{scaled}}$) and 2 (*UVET-benchmark* test) evaluate one shared grid: the core list above plus DR policy tree, CART-Rev, and UMCT (CADR) on *both* benchmarks.

- **Causal forests (CF$_{\text{mse}}$, CF$_{\text{het}}$)** (Battocchi et al., 2019; Wager and Athey, 2018). We call `CausalForest` from EconML with scalar $Y{=}$`rev`, discrete policy index T, and honest splitting (Appendix F.3). The implementation targets local solutions $\theta(x)$ to linear-moment equations $\mathbb{E}[J\theta(X) - A \mid X{=}x]{=}0$ with design matrices (J, A) supplied by the library for the chosen score. **CF$_{\text{mse}}$** (`criterion="mse"`) scores a candidate split by maximizing $\sum_{\text{child}} w_{\text{child}}\, \theta_{\text{child}}^{\top}\, \mathbb{E}[J \mid X{\in}\text{child}]\, \theta_{\text{child}}$, which matches weighted prediction error for the best linear effect model in each child when $|\mathcal{T}|{=}2$; with more than two policies EconML replaces raw θ-contrasts by stabilized pseudo-outcomes $\rho_{\text{child}} := J_{\text{parent}}^{-1}\mathbb{E}[A - J\theta_{\text{parent}} \mid X{\in}\text{child}]$ and maximizes $\sum_{\text{child}} w_{\text{child}}\, \rho_{\text{child}}^{\top}\, \mathbb{E}[J \mid X{\in}\text{child}]\, \rho_{\text{child}}$, down-weighting ill-conditioned Jacobians. **CF$_{\text{het}}$** (`criterion="het"`) instead maximizes $\sum_{\text{child}} w_{\text{child}}\, \rho_{\text{child}}^{\top}\rho_{\text{child}}$, prioritizing heterogeneity of those pseudo-effects across children (approximating ideal $\|\theta_{\text{left}} - \theta_{\text{right}}\|_2^2$ weighting). Minimum split/leaf sizes follow the same $|\text{fit sample}|/2^{D+1}$ and $|\text{fit sample}|/2^{D+2}$ scaling as UVET (Appendix F.1).
- **Merged partitions.** A merged partition takes the leaf labels of two partitions already fitted by other methods and intersects them, so each cohort is a pair (leaf of the first, leaf of the second) observed in training; test units whose pair never occurs in training are assigned to the first cohort. No cohorts are merged after the intersection. Merge (UVET$_{\text{cl}}$ + CF$_{\text{mse}}$) and Merge (UVET$_{\text{cl}}$ + CF$_{\text{het}}$) intersect a trained UVET$_{\text{cl}}$ partition with a CF$_{\text{mse}}$ or CF$_{\text{het}}$ partition; Merge (CF$_{\text{mse}}$ + CF$_{\text{het}}$) intersects the two CF partitions and serves as the UVET-free control. At depth $D{=}2$ the merges have 8 to 10 cohorts on UVET-benchmark, against 3 for UVET$_{\text{cl}}$ and 4 for UVET$_{\text{uncl}}$.
- **LBCF** (Ai et al., 2022). We run the large-scale budget-constrained causal forest on the same $(X, T,$ `rev`$)$ tabular view as CF, fit on a uniform subsample of 100,000 training rows

(seed 42) whenever the training set is larger; its forest defines cohort labels for the same downstream LP/ILP/mCUV evaluation.

- **CB ratio** (Sun et al., 2021). Two causal forests, fit on the same 100,000-row subsample rule as LBCF, estimate heterogeneous uplift in `rev` and in raw cost c; the partition/ranking stage uses the ratio of predicted uplifts (Appendix F.3), then cohort labels feed the common allocator protocol.
- **UVET$_{\text{cl}}$, UVET$_{\text{uncl}}$, UVET$_{\text{bay}}$**. Splits maximize the iso-cost marginal surplus in (11) using joint (v, c) within-node statistics (Section 3, Algorithm 1); clipping and Bayesian elasticity shrinkage are documented in Appendix F.1.
- **DR Policy Tree** (`DRPolicyTree` in EconML (Battocchi et al., 2019), honest policy tree on doubly robust scores): scalar outcome Y=`rev` and discrete policy index T, matching CF$_{\text{mse}}$/CF$_{\text{het}}$ inputs (Appendix F.3). Whenever $|\mathcal{X}_{\text{train}}| > 50{,}000$, nuisance and tree stages are fit on a uniform subsample of 50,000 training rows (seed 42); the fitted tree then assigns leaves for *all* train, validation, and test units via `apply` on full covariates. This cap binds on both MT-LIFT (4.6M training rows) and *UVET-benchmark* (43.4M training rows).
- **CART-Rev**: a `sklearn DecisionTreeRegressor` of depth D predicts `rev` from covariates X *only* (treatment never enters the impurity), so the partition captures heterogeneity in the cost-priced proxy orthogonal to how policies enter CF/DR scores. Minimum split/leaf thresholds use the same $|\text{fit sample}|/2^{D+1}$ and $|\text{fit sample}|/2^{D+2}$ scaling as CF/UVET at the actual fit size. On *UVET-benchmark*, the reproduction script for Table 2 sets `max_fit_samples=50,000` for parity with DR/UMCT at tens-of-millions scale. On MT-LIFT the default configuration leaves `max_fit_samples` unset for CART-Rev, so the tree can be fit on the entire public training partition (still far smaller than *UVET-benchmark*).
- **UMCT (CADR)** (De Vos et al., 2026): histogram-gradient-boosting generalized propensity score, GPS-weighted T-learner regressions for value and cost on the fit subsample, uplift score as predicted value at the highest policy index minus value at the reference index, then a depth-D `DecisionTreeRegressor` on that uplift vector with `apply` to all rows. The implementation caps CADR fitting and the uplift tree at 50,000 training rows (seed 42) whenever the training corpus exceeds that threshold, for both benchmarks. Only dataset-specific preprocessing differs between benchmarks (Appendices I, J).

Beyond the methods of Tables 1 and 2, the released code further registers OPL-style welfare maximization, E3IR-style monotone uplift variants, UMCT-ILP (De Vos et al., 2026), and GoDoTree-inspired doubly robust trees; these are omitted from Tables 1 and 2 when they (a) consistently rank outside the leading methods under LP allocation on our largest sets, or (b) exceed practical CPU budgets at full UVET-benchmark scale, but they remain available for ablations. CAIPWL (Zhou et al., 2023b) is timed in Table 9 on the same synthetic draw but omitted from Tables 1 and 2 for the same scaling reasons (cross-fitted propensity and per-policy outcome models on large n).

Allocators. Every learned partition is evaluated under three allocators: (i) marginal-cost equalization (mCUV, native to UVET); (ii) linear programming (LP) over fractional per-cohort assignments; (iii) mixed-integer linear programming (**ILP**) that assigns each cohort a single discrete treatment policy (Appendix G.4). All three are solved on the training split under the same total-cost budget and evaluated on the test split (Appendix G). **Main-text tables report $(\Delta v_{\mathcal{B}}, \Delta c_{\mathcal{B}})$ under LP, ILP, and mCUV in parallel** (Tables 1, 2); LP remains the primary comparator for allocator-agnostic partition quality, while Appendix G.4 describes the allocators and Appendix G.5 discusses allocator-induced bias.

Random seeds. All experiments fix the global seed for NumPy, Python's `random`, and the underlying tree learners. Synthetic sweeps and robustness studies use seed offsets $42 + r$ for replicate $r \in \{0, \ldots, n_{\text{rep}} - 1\}$ in the training generator and $1042 + r$ in the test generator. Bootstrap replicate i of Tables 1 and 2 uses seed $20260801 + i$.

F.5 Synthetic sweeps and runtime characterization

We complement the robustness suite with three lightweight sweeps on the correctly specified power-law generator described in Section H (default settings: four latent groups, log-normal multiplicative

Table 8: Synthetic power-law DGP (Section H): depth D, fit wall time (s), oracle *value* gap (%), cost violation (%), and leaf count ($n_{\text{train}} = 4000$, $n_{\text{test}} = 8000$, seed 42). Gap is total oracle value minus learned value, as a percentage of oracle value (Appendix H.2.1).

depth D	wall (s)	value gap (%)	cost viol. (%)	leaves
0	0.006	17.1	0.0	1
1	0.066	15.3	−0.5	2
2	0.178	13.2	1.0	4
3	0.363	13.6	0.1	8
4	0.576	12.2	6.1	11
5	0.670	13.2	4.1	11

Table 9: Synthetic loader wall-clock at $n_{\text{train}} = n_{\text{test}} = 3{\cdot}10^5$ (seconds): total, partition fit, one LP/ILP/mCUV timing replicate, and leaf count. A subset of the methods of Tables 1 and 2 plus CAIPWL. Not representative of full-corpus jobs (Appendix F.6).

Method	total (s)	partition (s)	eval (s)	leaves
No split	0.06	0.00	0.06	1
CF_{mse}	0.22	0.17	0.05	4
CF_{het}	0.16	0.11	0.05	4
Merge (UVET_{cl} + CF_{mse})	0.45	0.40	0.05	7
LBCF	0.41	0.36	0.05	3
CB ratio	0.41	0.35	0.05	2
CAIPWL	0.99	0.88	0.11	1
UVET_{cl}	0.45	0.36	0.08	4
$\text{UVET}_{\text{uncl}}$	0.33	0.26	0.07	4
UVET_{bay}	0.28	0.23	0.06	4

noise, $K{=}5$ treatment policies uniformly randomized over $\{0.8, 0.9, 1.0, 1.1, 1.2\}$, $n_{\text{test}} = 8000$, seed 42). Each sweep varies one factor at a time:

- **Tree depth.** Fix $n_{\text{train}} = 4000$ and vary $D \in \{0, \ldots, 5\}$; record wall time, oracle value gap vs. the latent-group oracle, cost violation, and the number of learned leaves.
- **Budget multiplier.** Fix $D = 3$ and vary the target total cost as a multiplier $m \in \{0.88, 0.95, 1.00, 1.05, 1.12\}$ of the realized total cost at $\alpha{=}1$ on the evaluation draw. The tree and its intensities are fit once at $m{=}1$ and kept fixed; only the budget that the oracle and the cost violation refer to changes.

The value gap shrinks from 17.1% at $D{=}0$ to 12–14% from $D{=}2$ on, and the learned intensities stay within 1% of the budget up to $D{=}3$. Deeper trees have smaller leaves, noisier elasticities and test overspends of 4 to 6%. In the budget sweep the cost violation is $1/m - 1$ up to sampling noise, since the policy does not change, and the gap grows with m because the oracle uses the extra budget while the fixed policy does not. We also report wall-clock times for a subset of the partitioning methods of Tables 1 and 2 on a common synthetic split ($n_{\text{train}} = n_{\text{test}} = 3 \times 10^5$, single CPU worker, identical seed): we measure partition fit time, evaluation time (one bootstrap replicate of mCUV/LP/ILP for timing only), and total wall time. At $n = 3 \times 10^5$ the UVET variants take 0.3 to 0.45 s, about twice the causal forests and about as long as LBCF and the CB ratio.

Under misspecification, the qualitative patterns in Appendix H are the authoritative robustness summary.

F.6 Compute resources

All experiments in this paper run on commodity CPU hardware. We summarize the compute budget below; no GPU is required.

Hardware. The $B{=}1000$ runs behind Tables 1 and 2 and Appendices G.6.2, I and J ran on a laptop with an Apple M5 CPU (10 cores) and 24 GB of RAM. The synthetic and robustness experiments

Table 10: Same setup as Table 8 at D=3, with the tree and intensities fit at m=1 and kept fixed: cost-target multiplier m vs. oracle value gap (%), cost violation (%), wall time (s).

target multiplier	value gap (%)	cost viol. (%)	wall (s)
0.88	7.2	13.7	0.388
0.95	11.2	5.3	0.367
1.00	13.6	0.1	0.367
1.05	15.6	−4.7	0.366
1.12	18.0	−10.7	0.366

Table 11: UVET-benchmark subsample: partition fit + leaf-assignment wall time (s), $n_{\text{train}} = 2{\times}10^6$, $n_{\text{test}} = 2{\times}10^5$, D=2, one CPU.

Method	wall (s)
CF_{mse}	1.65
CF_{het}	1.40
UVET_{cl}	5.00

ran on a Linux workstation with an Intel Xeon-class CPU (8 physical cores, about 32 GB of RAM). Every fit is single-threaded.

Wall-clock per run. Table 9 is deliberately run on the synthetic loader at $n_{\text{train}} = n_{\text{test}} = 3{\times}10^5$ so every baseline sees the same draw and hardware; relative ordering among these methods is meaningful, but comparing these seconds to production jobs on 86.7M rows is not. Per-method wall times on that micro-benchmark range from 0.06 s (no split) to 0.99 s (CAIPWL) for one fit-and-evaluate cycle among the listed methods; a repeat of the benchmark moves individual times by up to 25%. On MT-LIFT (4.6M training rows), fits take between 1 s and 12 min depending on the method (DR Policy Tree is the slowest), and the B=1000 shared-resample bootstrap of all 14 rows takes about 2 min. On UVET-benchmark (43.4M training rows), fits take between 5 s and 7.5 min per method, and the B=1000 bootstrap takes about 5 min per method; a depth-3 UVET fit completes in under 10 min on a single CPU.

UVET-benchmark absolute training times vs. key baselines (fixed n). Table 11 fixes a *single* contiguous subsample of $n_{\text{train}} = 2{\times}10^6$ rows from the released UVET-benchmark training split (same five numeric features and treatment policies as the full benchmark) and reports wall time for *partition fit plus assigning group labels* on that train draw and a $2{\times}10^5$-row test head, on one CPU core of the UVET-benchmark node described in the **Hardware** paragraph above. The goal is an apples-to-apples comparison on real schema and covariate scale, not to replace end-to-end job times at 43.4M train rows (those aggregate budgets are already stated above). At this n, the causal forests take 1.4 to 1.7 s and UVET_{cl} 5.0 s. When CF_{mse}/CF_{het} rows are faster than UVET_{cl} in Table 11, much of the gap is *implementation* rather than asymptotics: the EconML (Battocchi et al., 2019) causal-forest backend builds on a highly optimized GRF-style C++/Cython tree learner for a scalar outcome, whereas the released UVET reference path relies on pandas sorts, per-split log–log elasticity fits, and SciPy root-finding (Appendix F.6). A production UVET engine could narrow this wall-clock difference without changing the splitting objective. As n_{train} grows toward the full 43.4M-row training corpus, I/O and repeated scoring passes dominate wall time, so the main text points readers to the controlled synthetic micro-benchmark (Table 9) for the *ordering* of the baselines at fixed n.

Per split search, tree passes, and inference. Training a depth-D tree performs one *root-to-leaf* sequence of greedy decisions. At each internal node, UVET evaluates every numeric feature: for feature j it sorts the node table once on x_j (one $\mathcal{O}(n_{\text{node}} \log n_{\text{node}})$ pass), builds a percentile grid of candidate thresholds between the 25th and 75th percentiles with step `thr_step`=1 (roughly 50 candidates), precomputes `searchsorted` split indices for all candidates, then runs a unimodal search over the index (a few $\mathcal{O}(n_{\text{node}})$ evaluations of the iso-cost criterion via vectorized `bincount` over treatment policies). On the same $2{\times}10^6$-row subsample, one full five-feature split search at the root takes ≈ 2.1 s, while a single-feature threshold search takes ≈ 0.38 s—consistent with "one heavy

sort plus a handful of linear scans" per node. Child nodes repeat on disjoint subsets, so total training work scales with the sum of node sizes rather than D independent full scans of all n. *Inference* on held-out users is a depth-D comparison along the learned path ($\mathcal{O}(D)$ per row); the dominant cost at deployment is therefore the downstream allocator (mCUV closed form vs. LP/ILP), not tree traversal.

Engineering choices enabling 86.7M samples. (i) **Flat tabular storage:** gzip-compressed CSV ingested into columnar in-memory frames once per job (random access over features is required for sorting-based splits). (ii) **Discrete treatment coding:** policies are factorized to small integers so `bincount` aggregates each candidate split without Python loops per policy. (iii) **Global sufficient statistics:** a single global $(\hat{\gamma}, \hat{\eta}, \hat{V}_{\mathrm{ref}}, \hat{C}_{\mathrm{ref}})$ on the train split defines the scalar `rev` targets for forest baselines once, avoiding repeated full-pass recomputation inside every baseline. (iv) **Explicit sample caps where needed:** on both benchmarks, LBCF and the Sun et al. (2021) ratio baseline use subsampled fits at default caps (documented with the baseline entries), which prevents accidental $\mathcal{O}(n^2)$ behavior on tens of millions of rows. Table 9 instead times them *without* those caps at $n_{\mathrm{train}} = 3{\cdot}10^5$ so the wall-clock comparison matches causal forests and UVET on the same draw. (v) **Allocator choice:** UVET's native mCUV equalization is closed form per bucket after elasticities are fit, so training never invokes a per-example MILP; LP/ILP are reserved for *evaluation* of every method under a common allocator protocol.

Total compute. The robustness suite (14 scenarios $\times$ up to 5 replicates) completes in under 30 min on a single CPU. The synthetic sweeps (depth, training size, budget, runtime characterization) add another $\sim$15 min. Real-data experiments (MT-LIFT in three preprocessing variants and UVET-benchmark with all baselines) consume the bulk of total compute: under 200 CPU-hours overall, including pilot/dev runs not reported in the paper.

End-to-end reproduction time. On the laptop above, one script in the code repository reruns every experiment in about five hours: the toy example, synthetic sweeps and robustness suite take a few minutes, the MT-LIFT runs behind Tables 1 and 15 about 50 minutes, the UVET-benchmark runs behind Tables 2 and 18 about two hours, and the λ_0 sweep and validation of Appendix G.6.2 about two hours. The UVET-benchmark runs need the released dataset and at least 24 GB of RAM.

G Evaluation Metrics and Protocol

This section provides a self-contained discussion of the evaluation metrics used throughout the paper, their relationship to standard uplift metrics, and the rationale behind our three-allocator evaluation protocol.

G.1 Primary metrics: relative value and cost uplift

Our primary metrics are the relative value uplift $\Delta v_{\mathcal{B}}$ and relative cost change $\Delta c_{\mathcal{B}}$ defined in Section 4. These directly measure the quantity a decision-maker cares about: how much additional value does the learned policy generate, and at what cost relative to the status quo? A method is preferable if it achieves higher $\Delta v_{\mathcal{B}}$ while maintaining $\Delta c_{\mathcal{B}} \leq 0$ (cost neutrality or savings).

G.2 Visual comparison in the $(\Delta v_{\mathcal{B}}, \Delta c_{\mathcal{B}})$ plane

Figures 2–4 place every row of Tables 1 and 2 in the $(\Delta v_{\mathcal{B}}, \Delta c_{\mathcal{B}})$ test plane, one figure per allocator, at the same means and ± 1 bootstrap standard deviations ($B{=}1000$). The bars are per coordinate, not joint regions. Points are numbered in table order, and the shaded half-plane is over budget.

G.3 Relationship to standard uplift metrics

The uplift modeling literature predominantly evaluates methods using the Qini coefficient (Radcliffe and Surry, 2007) and the area under the uplift curve (AUUC) (Rzepakowski and Jaroszewicz, 2012). These metrics assess how well a model *ranks* individuals by their treatment effect: the uplift curve plots cumulative incremental gain as a function of the fraction of the population targeted, and Qini/AUUC summarize this curve with a single number.

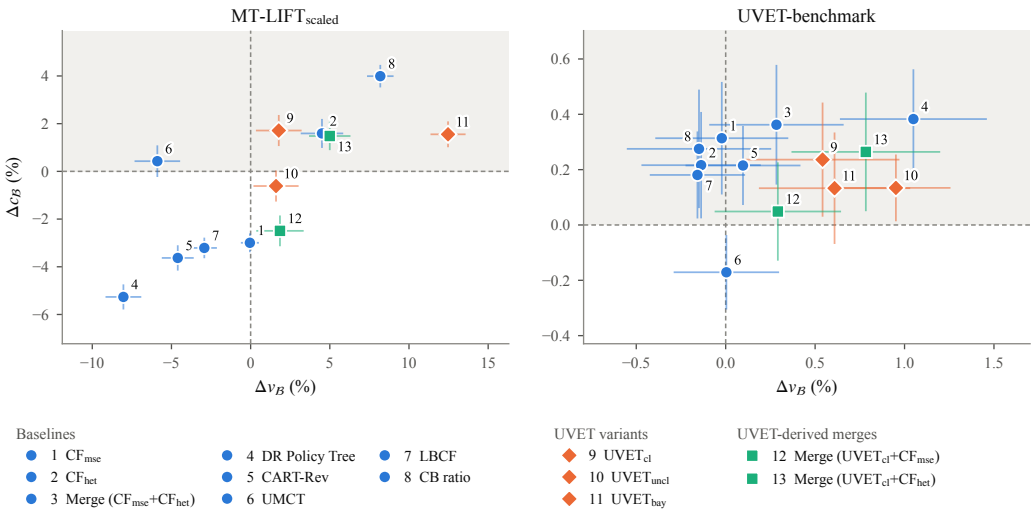


Figure 2: LP allocator, test split, D=2: mean $(\Delta v_{\mathcal{B}}, \Delta c_{\mathcal{B}})$ of each row with ± 1 bootstrap std per axis; shaded: $\Delta c_{\mathcal{B}} > 0$. Row 3 appears on UVET-benchmark only. On MT-LIFT$_{\text{scaled}}$, UVET$_{\text{bay}}$ (11) gains the most value by a wide margin, at a 1.6% overspend. On UVET-benchmark, UVET$_{\text{uncl}}$ (10) gains the most value of the rows within budget; the DR Policy Tree (4) gains slightly more and overspends, and most rows sit slightly over budget.

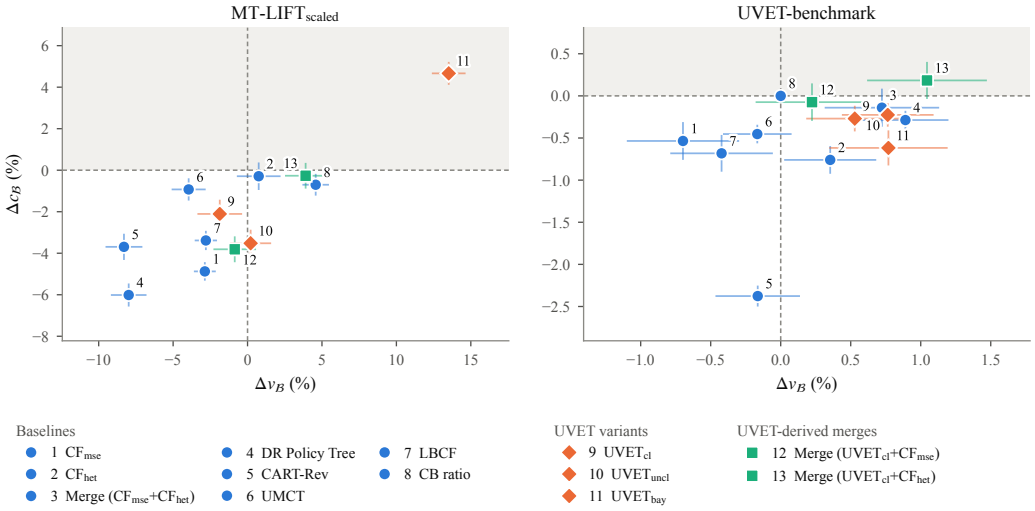


Figure 3: ILP allocator: same layout as Figure 2. On MT-LIFT$_{\text{scaled}}$, UVET$_{\text{bay}}$ (11) again gains the most value and is the only row over budget, by 4.7%. On UVET-benchmark, the merge of UVET$_{\text{cl}}$ with CF$_{\text{het}}$ (13) gains the most value.

These metrics are not directly applicable to our setting for three reasons:

1. **Binary vs. continuous treatment.** Qini and AUUC are defined for binary (treat/no-treat) decisions. In our setting, treatment is a continuous intensity parameter $\alpha \in [\alpha_{\min}, \alpha_{\max}]$, and the decision is how much to treat each cohort. Extending Qini to continuous treatment would require marginalizing over treatment levels, losing the operational meaning of the curve.

2. **Ranking vs. allocation.** Uplift curves evaluate whether a model correctly ranks individuals by decreasing treatment effect. Under a budget constraint with heterogeneous costs,

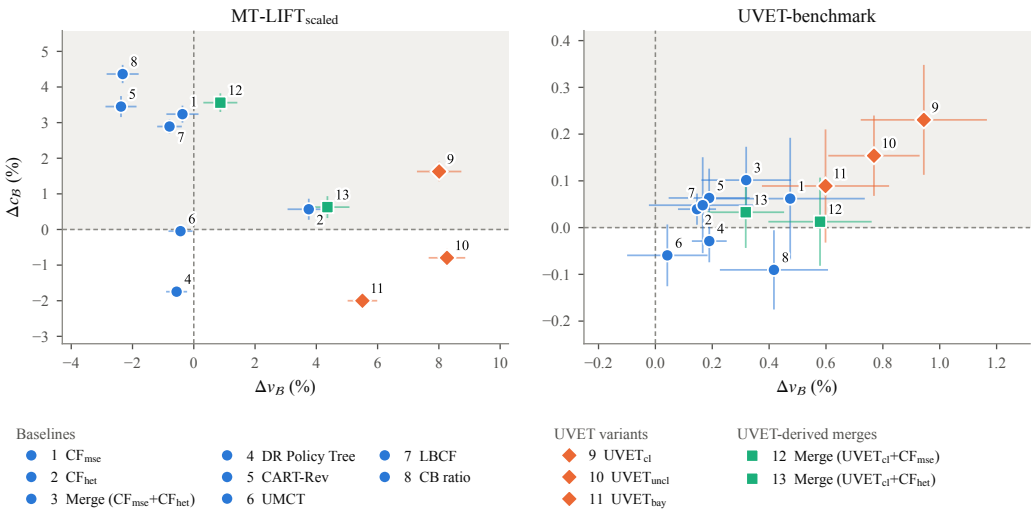


Figure 4: mCUV equalization: same layout as Figure 2. On MT-LIFT$_{\text{scaled}}$, four baseline partitions land in the upper-left quadrant (less value at higher cost), while UVET$_{\text{uncl}}$ (10) and UVET$_{\text{bay}}$ (11) gain value within budget.

however, the optimal targeting rule does not simply rank by treatment effect; it ranks by the cost-adjusted ratio $\tau_v(x)/\tau_c(x)$ (Sun et al., 2021). Standard Qini ignores costs entirely.

3. **Dual outcomes.** UVET jointly optimizes value and cost. Qini and AUUC are defined with respect to a single outcome, and there is no consensus on how to extend them to the dual-outcome, budget-constrained setting.

Our $\Delta v_{\mathcal{B}}$ / $\Delta c_{\mathcal{B}}$ metrics directly evaluate what the decision-maker optimizes: total value under a cost constraint. This is consistent with the policy-value evaluation used in the budget-constrained allocation literature (Sun et al., 2021; Ai et al., 2022; Tu et al., 2021; De Vos et al., 2026).

G.4 Three-allocator evaluation protocol

To disentangle the quality of user-space partitioning from the choice of downstream allocator, we evaluate every partitioning method under three allocation strategies:

- **Linear Programming (LP)**: a stochastic relaxation where each cohort receives a distribution over the discrete treatment policies observed in the data. LP does not assume a power-law form and is our primary comparator across methods.
- **Integer Linear Programming (ILP)**: assigns each cohort exactly one discrete policy via a MILP. Strongest fully deterministic discrete policy; useful when stochastic mixing is not deployable. We write **ILP** throughout the paper; the supplementary implementation historically exposes the same routine as `mi_opt`.
- **mCUV equalization** (continuous, closed-form): the allocator native to UVET under the power-law model. Reports the performance of the full modeling pipeline, not allocator-agnostic partition quality alone.

By cross-evaluating partitions and allocators, we can separate partition quality (LP/ILP) from the additional lift obtained when the native mCUV rule is trusted.

G.5 Allocator bias and the role of LP as primary metric

An important subtlety of the three-allocator protocol is that the mCUV allocator is not a neutral evaluator of partition quality. Because the UVET splitting criterion directly maximizes the mCUV-allocated value gain (Equation 11), partitions produced by UVET are *designed* to make the mCUV allocator perform well. Other partitioning methods (causal forests, policy and regression trees)

optimize different objectives (CATE heterogeneity, revenue prediction) and produce partitions that are not aligned with power-law equalization. Evaluating all methods under mCUV therefore gives UVET a structural advantage that conflates partition quality with allocator fit.

LP and ILP, by contrast, are allocator-agnostic: they optimize over the observed discrete treatment set using the empirical group-level value and cost means, without assuming any parametric form. A partition that achieves high $\Delta v_{\mathcal{B}}$ under LP does so because it creates groups with genuinely different optimal treatments, regardless of whether the underlying response follows a power law.

For this reason, we designate **LP as the primary evaluation metric** throughout the paper. mCUV results are reported as a secondary metric that quantifies the additional gain from pairing a UVET partition with its native closed-form allocator—a legitimate operational advantage, but one that should not be confused with partition quality.

G.6 Budget-penalized summary score

Tables in the main paper report $(\Delta v_{\mathcal{B}}, \Delta c_{\mathcal{B}})$ as percentages (metrics $\times 100$) for readability; the underlying definitions in Section 4 remain dimensionless relative uplifts. We also report a single *budget-penalized* scalar that condenses the value–cost trade-off into one number and is consistent with the inequality budget constraint $\sum_b c_b \leq C$ in eq. (1): allocating below the reference budget ($\Delta c_{\mathcal{B}} \leq 0$) is feasible and is therefore not penalized, while cost overrun ($\Delta c_{\mathcal{B}} > 0$) is mapped through a nonnegative, convex penalty and priced in value units at rate $1/\varepsilon_{\text{glob}}$.

Let $\varepsilon_{\text{glob}} := \gamma_{\text{glob}}/\eta_{\text{glob}}$ denote the ratio of global value to cost elasticities estimated from aggregate log–log regressions on the training split. Fix a curvature parameter $\kappa > 0$ (we use $\kappa{=}1$ in all experiments: for tiny violations the map below is first-order equivalent to the linear hinge $\max(0, \Delta c)$, while larger cost overruns incur increasingly severe marginal penalties). For each allocator $a \in \{\text{ILP}, \text{LP}\}$ and penalty coefficient $\lambda \geq 0$, define

$$S_\lambda^{(a)} := \Delta v_{\mathcal{B}}^{(a)} - \frac{\lambda}{\varepsilon_{\text{glob}}} \Big(\exp\big(\kappa \max(0, \Delta c_{\mathcal{B}}^{(a)})\big) - 1 \Big), \tag{27}$$

where $(\Delta v_{\mathcal{B}}^{(a)}, \Delta c_{\mathcal{B}}^{(a)})$ are the test-set uplifts under allocator a. For $\lambda = 1$ and infinitesimal cost overrun, the bracket behaves like $\kappa \max(0, \Delta c)$, so one point of overrun costs $1/\varepsilon_{\text{glob}}$ points of value, a factor $1/\varepsilon_{\text{glob}}^2$ above the global marginal rate of value per unit of cost, which makes the penalty deliberately conservative; for larger positive Δc the exponential term prices violations more aggressively than a linear hinge. The combined allocator-agnostic score averages LP and ILP at the same λ,

$$S_\lambda^{(\text{comb})} := \tfrac{1}{2}\big(S_\lambda^{(\text{ILP})} + S_\lambda^{(\text{LP})}\big), \tag{28}$$

excluding mCUV from the average: including mCUV would reward partitions specifically aligned with UVET's native allocator and would favour UVET-shaped partitions. Tables 1 and 2 report the shorthand $S_{\text{comb}} := S_1^{(\text{comb})}$.

G.6.1 Multi-λ sensitivity sweep

Tables 12–13 report $S_\lambda^{(\text{comb})}$ for $\lambda \in \{0.5, 1, 2\}$ on both real datasets. Smaller λ shrinks the multiplier in front of the nonnegative cost-overrun bracket in (27); $\lambda = 1$ is the setting of Tables 1 and 2 ($S_{\text{comb}} = S_1^{(\text{comb})}$), and $\lambda = 2$ doubles that multiplier (a conservative pricing that strongly favors staying at or below the reference cost). Methods within 1σ of the top score are shown in bold.

When the three λ columns coincide. Recall (27): the cost-overrun bracket vanishes whenever $\Delta c_{\mathcal{B}}^{(a)} \leq 0$. Since $S_\lambda^{(\text{comb})}$ is the average of the LP and ILP scores at the same λ (eq. 28), a row is flat across $\lambda \in \{0.5, 1, 2\}$ if **both** LP and ILP yield $\Delta c_{\mathcal{B}} \leq 0$ on every bootstrap resample for that method—then $S_\lambda^{(\text{comb})}$ reduces to $\frac{1}{2}(\Delta v_{\mathcal{B}}^{(\text{LP})} + \Delta v_{\mathcal{B}}^{(\text{ILP})})$ and cannot depend on λ. Rows that *do* change with λ (e.g., UVET_{cl} and CF_{het} on $\text{MT-LIFT}_{\text{scaled}}$) are those where at least one allocator incurs a positive $\Delta c_{\mathcal{B}}$ on some bootstrap draws, so cost overrun is being priced at different strengths; with (27) the marginal penalty grows faster than linearly in Δc once overrun is non-negligible.

Penalizing budget slack. A symmetric penalty on $|\Delta c|$ treats feasible slack ($\Delta c < 0$) like a mistake and pushes scores toward policies that hug the reference cost regardless of value. Under (27), allocating below the reference leaves the penalty term at zero, so UVET partitions that are conservative but feasible under LP/ILP retain their full Δv in $S_\lambda^{(\mathrm{comb})}$. That choice matches the inequality budget in eq. (1). Penalizing negative Δc symmetrically is appropriate only if "cost closer to the reference" is itself an operational goal (e.g., avoiding large unused budgets in a fixed fiscal envelope); it is not required for ranking partitions on value subject to not exceeding cost.

Table 12: MT-LIFT$_{\text{scaled}}$: $S_\lambda^{(\mathrm{comb})}$ (%) for $\lambda \in \{0.5, 1, 2\}$, D=2 (eqs. 27–28); mean ± std over the B=1000 shared resamples of Table 1; $1/\varepsilon_{\text{glob}} = 1.58$.

Method	$S_{\lambda=0.5}^{(\mathrm{comb})}$	$S_{\lambda=1}^{(\mathrm{comb})}$	$S_{\lambda=2}^{(\mathrm{comb})}$
CF$_{\text{mse}}$	-1.46 ±0.62	-1.46 ±0.62	-1.46 ±0.62
CF$_{\text{het}}$	1.94 ±1.28	1.24 ±1.26	-0.14 ±1.46
DR Policy Tree	-8.01 ±1.13	-8.01 ±1.13	-8.01 ±1.13
CART-Rev	-6.45 ±1.08	-6.45 ±1.08	-6.45 ±1.08
UMCT (CADR)	-5.14 ±1.14	-5.35 ±1.12	-5.78 ±1.20
LBCF	-2.86 ±0.73	-2.86 ±0.73	-2.86 ±0.73
CB ratio	4.77 ±0.79	3.15 ±0.80	-0.09 ±0.96
UVET$_{\text{cl}}$	-0.73 ±1.34	-1.41 ±1.29	-2.78 ±1.34
UVET$_{\text{uncl}}$	0.88 ±1.34	0.86 ±1.32	0.82 ±1.30
UVET$_{\text{bay}}$	**10.48** ±1.05	**7.98** ±1.16	**2.97** ±1.73
Merge (UVET$_{\text{cl}}$ + CF$_{\text{mse}}$)	0.50 ±1.45	0.50 ±1.45	0.50 ±1.45
Merge (UVET$_{\text{cl}}$ + CF$_{\text{het}}$)	3.80 ±1.23	3.16 ±1.21	**1.87** ±1.42

Table 13: UVET-benchmark: $S_\lambda^{(\mathrm{comb})}$ (%) for $\lambda \in \{0.5, 1, 2\}$, D=2; mean ± std over the B=1000 shared resamples of Table 2; $1/\varepsilon_{\text{glob}} = 7.31$. Rows grouped as in Table 2.

Method	$S_{\lambda=0.5}^{(\mathrm{comb})}$	$S_{\lambda=1}^{(\mathrm{comb})}$	$S_{\lambda=2}^{(\mathrm{comb})}$
CF$_{\text{mse}}$	-0.94 ±0.34	-1.53 ±0.62	-2.69 ±1.30
CF$_{\text{het}}$	-0.31 ±0.32	-0.73 ±0.58	-1.57 ±1.18
Merge (CF$_{\text{mse}}$ + CF$_{\text{het}}$)	-0.23 ±0.45	-0.97 ±0.87	-2.45 ±1.85
DR Policy Tree	0.27 ±0.38	-0.44 ±0.62	-1.84 ±1.23
CART-Rev	-0.43 ±0.30	-0.84 ±0.46	-1.64 ±0.90
UMCT (CADR)	-0.09 ±0.25	**-0.11** ±0.25	-0.13 ±0.28
LBCF	-0.64 ±0.29	-0.99 ±0.47	-1.68 ±0.94
CB ratio	-0.60 ±0.31	-1.12 ±0.64	-2.16 ±1.35
UVET$_{\text{cl}}$	0.08 ±0.38	-0.38 ±0.64	-1.30 ±1.29
UVET$_{\text{uncl}}$	**0.59** ±0.30	**0.33** ±0.43	-0.21 ±0.81
UVET$_{\text{bay}}$	**0.39** ±0.39	**0.09** ±0.55	-0.51 ±1.07
Merge (UVET$_{\text{cl}}$ + CF$_{\text{mse}}$)	-0.03 ±0.39	-0.31 ±0.68	-0.89 ±1.41
Merge (UVET$_{\text{cl}}$ + CF$_{\text{het}}$)	0.03 ±0.60	-0.85 ±1.23	-2.61 ±2.60

On MT-LIFT$_{\text{scaled}}$, UVET$_{\text{bay}}$ has the highest score at every λ. Its lead shrinks at $\lambda = 2$, where Merge (UVET$_{\text{cl}}$ + CF$_{\text{het}}$) comes within one std of it: the ILP allocation of UVET$_{\text{bay}}$ overspends by 4.7% on the test week, and the penalty grows faster than linearly in the overspend. On UVET-benchmark, where $1/\varepsilon_{\text{glob}}$ is $4.6\times$ larger, UVET$_{\text{uncl}}$ and UVET$_{\text{bay}}$ lead at $\lambda = 0.5$ and $\lambda = 1$, and at $\lambda = 1$ they are the only rows with a positive mean. At $\lambda = 2$ no row scores above the reference policy, whose score is zero, so no entry is bold; UMCT, which stays below the reference cost, comes closest.

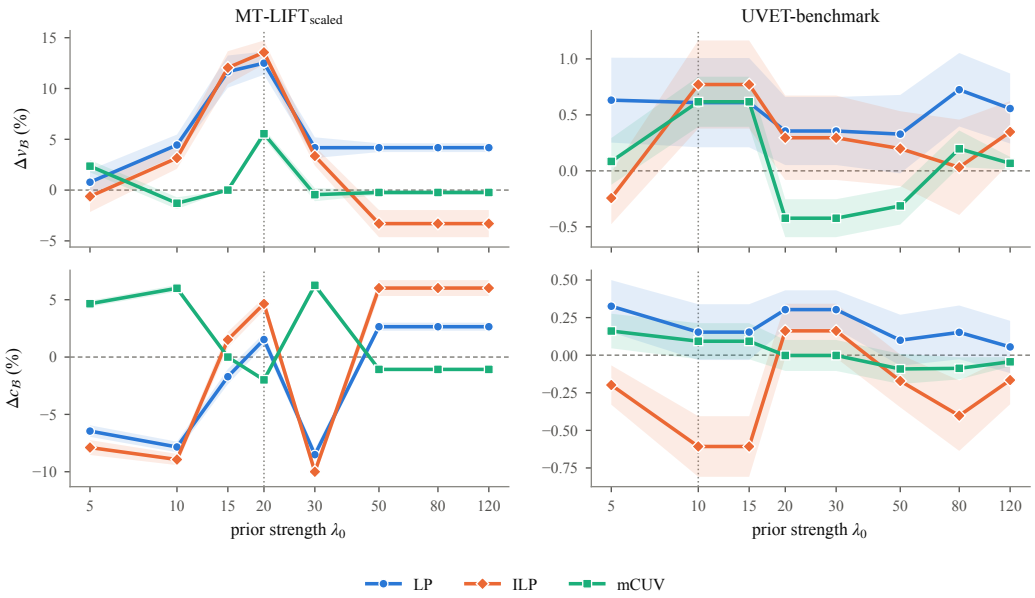


Figure 5: UVET$_{\text{bay}}$ test uplifts against the prior strength λ_0 (log scale), default minimum leaf size, D=2: mean $\pm$1 bootstrap std over B=200 shared resamples, one line per allocator. Dotted line: the value chosen on validation and used in Tables 1 and 2.

G.6.2 Sensitivity of UVET$_{\text{bay}}$ to λ_0 and the minimum leaf size

The prior strength λ_0 in (12) sets how far node elasticities are pulled toward the root estimate; it is unrelated to the penalty weight λ of $S_\lambda^{(\text{comb})}$ (Appendix G.6.1). We refit UVET$_{\text{bay}}$ at D=2 for $\lambda_0 \in \{5, 10, 15, 20, 30, 50, 80, 120\}$ with the default minimum leaf size, and at λ_0=30 with leaf floors from 0.5 to 1.5 times the default $\lfloor |\mathcal{X}^{\text{tr}}|/2^{D+2} \rfloor$. Every configuration is scored on the first B=200 of the shared test resamples of Tables 1–2, so the points at the values used in those tables (λ_0=20 on MT-LIFT$_{\text{scaled}}$, λ_0=10 on UVET-benchmark) reproduce their UVET$_{\text{bay}}$ rows replicate by replicate (Figure 5).

The leaf floor has no effect over this range. On both datasets every floor returns the same partition as the default, so the depth-2 splits never reach it.

The prior strength matters on MT-LIFT$_{\text{scaled}}$. Different values of λ_0 select different splits, and LP value is not monotone in λ_0: 0.8% at λ_0=5, 4.4% at λ_0=10, 11.7% within budget at λ_0=15, 12.5% with a 1.5% overspend at λ_0=20, and about 4.2% from λ_0=30 on. The large gain of Table 1 therefore needs λ_0 between 15 and 20. On UVET-benchmark the prior strength matters little: LP value stays between 0.33% and 0.72% over the whole range, with no clear trend (0.61% to 0.63% for $\lambda_0 \leq 15$, 0.36% at $\lambda_0 \in \{20, 30\}$, 0.72% at λ_0=80).

Choosing λ_0 without the test split. Tables 1 and 2 use the value of λ_0 chosen without the test split. We fit each value on training rows only, score it on held-out validation rows, and keep the highest validation S_{comb}. UVET-benchmark uses its released validation rows (`is_val`); MT-LIFT has no validation split, so we hold out a random third of its training rows. The chosen value is then refit on the full training split. On MT-LIFT$_{\text{scaled}}$, validation S_{comb} lies between 18.7 and 20.2 for $\lambda_0 \in \{10, 15, 20, 30\}$ and collapses from 50 on; it selects λ_0=20. Validation separates these four values far less than the test split does, where λ_0=10 and λ_0=30 fall to about 4%, so the MT-LIFT$_{\text{scaled}}$ gain of UVET$_{\text{bay}}$ rests on a choice that validation makes with little margin. On UVET-benchmark validation selects λ_0=10 (tied with λ_0=15, which returns the same partition), which weights each node estimate by 5/15. Validation uplifts are nearly twice the test uplifts on MT-LIFT$_{\text{scaled}}$, since the held-out rows come from the training weeks, so the validation split serves to rank values of λ_0, and its level overstates the test gain.

G.7 Additional metrics in robustness experiments

For the synthetic robustness experiments (Section H), we additionally report the *oracle value gap* (total oracle value minus learned value, normalized by oracle value) and cost-constraint violation, defined in Appendix H.2.1. These are synthetic-only diagnostics: unlike the main paper's $\Delta v_{\mathcal{B}}$ uplift vs. a logged reference policy, the oracle here uses the simulator's true response functions. Cost-constraint violation is the signed relative deviation from the target budget (eq. (36)).

H Robustness to Elasticity Misspecification

Our framework relies on a bucket-wise parametric assumption on treatment responses,

$$v_b(\alpha) = v_b(1)\alpha^{\gamma_b}, \qquad c_b(\alpha) = c_b(1)\alpha^{\eta_b}, \qquad \gamma_b < \eta_b, \tag{29}$$

which enables closed-form marginal-cost equalization and an efficient split criterion. While this assumption is reasonable in many practical settings, it is inevitably an approximation. In this section we design a systematic suite of *synthetic robustness experiments* to quantify the sensitivity of the method to misspecification. All experiments are fully controlled and reproducible: data are generated from known ground-truth response functions; the learner is the same UVET tree builder and mCUV allocation used in the main experiments (i.e., the model assumes power laws when fitting); and the resulting policy is evaluated against an oracle computed under the true data-generating process. Hyperparameter values and random seeds for replication are summarized in §H.11.

Throughout this section, we emphasize *oracle value gaps*, *constraint satisfaction*, and *partition stability* rather than absolute performance, with the goal of isolating failure modes induced specifically by model misspecification.

H.1 Common Experimental Backbone

All robustness experiments share the same backbone.

Features and true groups. We sample covariates $x \in \mathbb{R}^d$ from a fixed distribution (e.g., multivariate Gaussian or log-normal). A set of G latent groups $g(x) \in \{1, \dots, G\}$ is defined via known axis-aligned rules. These groups serve as the ground-truth segmentation that an oracle tree would recover.

Treatments. Unless otherwise noted, each unit is assigned a treatment level α drawn uniformly from a finite grid (default $\{0.8, 0.9, 1.0, 1.1, 1.2\}$). The treatment-support experiment varies only the cardinality and spacing of this grid. Continuous α is supported by the simulator but not used in the sweeps reported below.

Outcomes. Given group g and treatment α, cost and value are generated as

$$c = c_g(\alpha)\,\varepsilon^c, \qquad v = v_g(\alpha)\,\varepsilon^v, \tag{30}$$

where $\varepsilon^c, \varepsilon^v$ are multiplicative noise terms (log-normal unless otherwise noted).

Evaluation protocol. We fit elasticities and learn splits using the proposed algorithm, read off the leaf-level allocations produced during tree construction, and evaluate the induced policy on an independent test set drawn from the same response simulator. As a benchmark, we numerically solve the true constrained welfare problem with *one* treatment level per latent group $g \in \{1, \dots, G\}$ (full knowledge of heterogeneity), subject to the same total-cost budget (total cost at uniform α=1 on the evaluation sample).

H.2 Metrics

We report (i) value at matched total cost, (ii) oracle value gap relative to the simulator's constrained optimum, (iii) cost-constraint violations, and (iv) stability and recovery of the learned partition. Formal definitions of the oracle value gap, cost violation, and partition metrics appear in Appendices H.2.1, H.2.2, and H.2.3.

H.2.1 Oracle value gap under elasticity misspecification

To evaluate robustness to violations of the power-law elasticity assumption, we measure the *oracle value gap*: the shortfall in total value of the learned policy relative to an oracle that has access to the true treatment–response functions used to generate the data. This gap is defined at the policy level.

Let i index units on the evaluation sample and let $g_i \in \{1, \dots, G\}$ denote the true latent group. Write $v_g(\alpha)$ and $c_g(\alpha)$ for the *true* value and cost functions, which may deviate from the assumed power-law form.

Oracle policy. The oracle chooses one treatment level α_g^{oracle} per latent group by solving the constrained problem

$$\max_{\{\alpha_g\}_{g=1}^G} \quad \sum_{i=1}^n v_{g_i}(\alpha_{g_i}) \tag{31}$$

$$\text{s.t.} \quad \sum_{i=1}^n c_{g_i}(\alpha_{g_i}) = C, \tag{32}$$

$$\alpha_{\min} \le \alpha_g \le \alpha_{\max}, \quad g = 1, \dots, G,$$

where n is the evaluation sample size and C is the target total cost; by default, C equals the realized total cost at uniform $\alpha{=}1$ on the same draw. This problem is solved numerically with full knowledge of (v_g, c_g).

Learned policy. The learned policy assigns a treatment α_i^* to each unit according to the leaf of the fitted UVET tree and the leaf-level allocation stored during recursive partitioning (power-law elasticities are used only to build the tree and those allocations).

Oracle value gap. Let Gap_{orc} denote the difference in total value between the oracle policy and the learned policy on the same evaluation units:

$$\text{Gap}_{\text{orc}} \;=\; \sum_{i=1}^n v_{g_i}(\alpha_{g_i}^{\text{oracle}}) \;-\; \sum_{i=1}^n v_{g_i}(\alpha_i^*). \tag{33}$$

We report the normalized gap as a percentage of oracle value,

$$\text{Gap}_{\text{orc}}^{(\%)} \;=\; 100 \times \frac{\sum_{i=1}^n v_{g_i}(\alpha_{g_i}^{\text{oracle}}) - \sum_{i=1}^n v_{g_i}(\alpha_i^*)}{\sum_{i=1}^n v_{g_i}(\alpha_{g_i}^{\text{oracle}})}. \tag{34}$$

Interpretation. This benchmark compares a full-information optimum on the true coarse types to the tree-based policy returned by UVET on the same population and matched total-cost budget. The gap therefore aggregates both segmentation error (leaves need not align with latent groups) and allocation error within leaves; it is not decomposed in the experiments below. Because the oracle always uses G latent groups while the fitted tree may use fewer leaves at moderate depth, Gap_{orc} can remain positive even when the global DGP matches the power-law family on each group.

H.2.2 Cost-Constraint Violations

In addition to the oracle value gap, we evaluate the extent to which the learned policy satisfies the global cost constraint under elasticity misspecification.

Let $\{\alpha_b^*\}$ denote the treatment levels selected by the learned policy, and let $c_b(\alpha)$ be the true cost function for bucket b. The realized total cost of the learned policy is

$$C^{\text{learned}} \;=\; \sum_b n_b \, c_b(\alpha_b^*). \tag{35}$$

Recall that the target cost C is enforced exactly only under the assumed power-law model. When the true cost function deviates from this assumption, the realized cost may differ slightly from the target.

We define the *cost-constraint violation* as the relative deviation of the realized cost from the target:

$$\mathrm{CostViolation}(\%) \;=\; 100 \times \frac{C^{\mathrm{learned}} - C}{C}. \tag{36}$$

Positive values correspond to budget overruns, while negative values indicate conservative allocations that remain below the available budget.

Implementation note. C^{learned} and C are computed as sums of per-unit costs, so $\mathrm{CostViolation}(\%)$ equals the relative difference between the empirical mean cost under the learned policy and the empirical mean cost at the reference $\alpha{=}1$ on the evaluation sample (equivalently, the same ratio for total costs).

Interpretation. This metric quantifies the robustness of the marginal cost equalization mechanism to functional-form misspecification. Small violations indicate that the learned policy continues to approximately respect the intended budget even when the assumed elasticity model is incorrect.

H.2.3 Stability and Recovery of the Learned Partition

Finally, we evaluate the robustness of the learned segmentation itself, independently of allocation accuracy.

Partition recovery. Because all robustness experiments are conducted on synthetic data with known latent groups, we assess how well the learned tree partition recovers the ground-truth grouping. Let g_i denote the true group assignment of unit i, and let $\hat{g}_i$ denote the bucket assignment induced by the learned tree. We measure partition recovery using standard clustering similarity metrics, including the Adjusted Rand Index (ARI) and Normalized Mutual Information (NMI).

Partition stability. Unless stated otherwise, reported means and error bars pool replicate runs with independent random seeds (treatment assignment and noise). We do not compute an explicit all-pairs similarity matrix between learned partitions across seeds; variability across replicates nevertheless indicates sensitivity of ARI/NMI to sampling noise.

Interpretation. High recovery indicates that the learned tree captures the underlying heterogeneity structure even under elasticity misspecification, while high stability indicates that the splits are not overly sensitive to sampling noise or model error. Together, these metrics allow us to disentangle errors arising from incorrect segmentation from those arising from miscalibrated allocation within otherwise stable buckets.

H.3 Misspecified but Concave Response Functions

We first study settings in which the true response functions are concave and monotone, but not power laws.

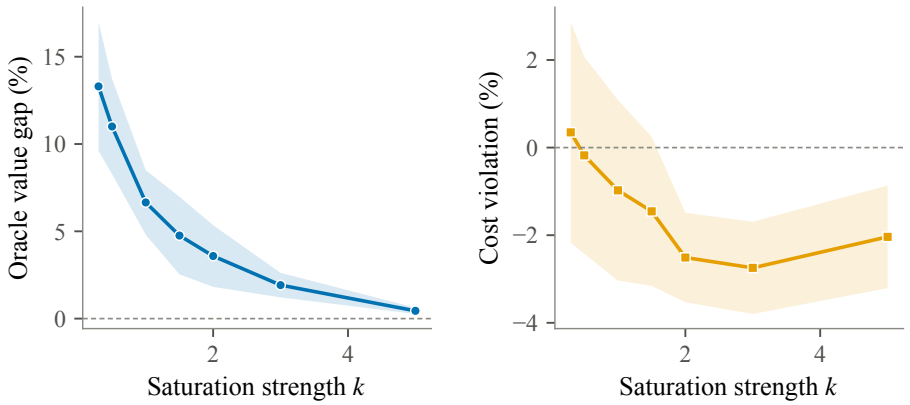
H.3.1 Saturating Value Responses

We generate value functions of the form

$$v_g(\alpha) = v_{0g}\left(1 - e^{-k_g \alpha}\right), \tag{37}$$

or alternatively Michaelis–Menten curves $v_g(\alpha) = v_{0g}\alpha/(\alpha + s_g)$, while costs remain linear or mildly convex. These functions exhibit diminishing returns with a pronounced flattening at high α, violating the constant-elasticity assumption.

Results. The mean oracle value gap falls from 13.3% at $k{=}0.3$ to 0.5% at $k{=}5$ (Figure 6). Strong saturation flattens every value curve over the treatment range, so where the budget goes matters little. Mean cost violations lie between -2.8% and $+0.4\%$.



(a) Oracle value gap (%)

(b) Cost violation

Figure 6: Exponential saturation in synthetic value responses vs. strength k; shaded bands are ± 1 std over five seeds.

H.3.2 Piecewise Power Laws

We next consider regime-switching responses:

$$v_g(\alpha) = \begin{cases} v_{0g}\alpha^{\gamma_{g,1}}, & \alpha \le \tau_g, \\ v_{0g}\tau_g^{\gamma_{g,1}-\gamma_{g,2}}\alpha^{\gamma_{g,2}}, & \alpha > \tau_g. \end{cases} \tag{38}$$

Such functions satisfy the power-law assumption locally but not globally.

Results. Unless noted otherwise, means and standard deviations are computed over five random seeds (training-set seed offsets $42{+}r$ and test seed $1042{+}r$ for $r \in \{0, \dots, 4\}$) at $n_{\text{train}}{=}5000$, $n_{\text{test}}{=}10000$, $D{=}2$, and minimum-leaf size 200. With four latent groups, the mean oracle value gap is 4.6% (std 1.4% across seeds), the mean cost violation -1.3% (std 1.3%), and the mean ARI 0.17, so the leaves match the latent groups only in part (piecewise row of Figure 7).

H.3.3 Logarithmic Returns

We also consider logarithmic responses $v_g(\alpha) = v_{0g}\log(1 + k_g\alpha)$, which are concave everywhere but have no constant elasticity.

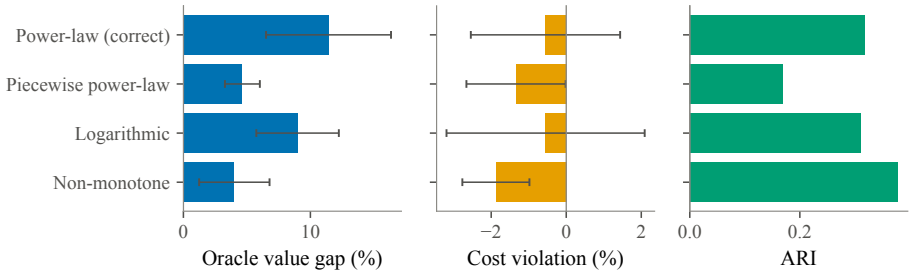
Results. With the same protocol, the mean oracle value gap is 9.0% (std 3.2%), the mean cost violation -0.6% (std 2.6%), and the mean ARI 0.31. The correctly specified generator has a larger gap, 11.4% (std 4.9%), at a similar ARI of 0.32. A depth-2 tree recovers the latent groups only in part, and the generators differ in how much value the correct grouping is worth, so these gaps mostly measure segmentation error.

H.4 Violations of $\gamma_b < \eta_b$ and Non-Concavity

H.4.1 Increasing Returns in a Subset of Buckets

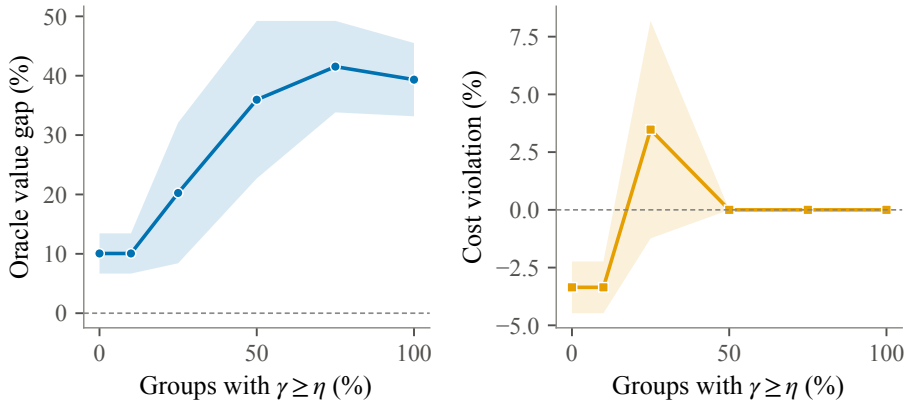
To test robustness to structural violations, we generate a fraction p of groups with value growing as fast as or faster than cost (i.e., $\gamma_g \ge \eta_g$).

Results. We vary the fraction p of *latent* groups for which increasing returns are enforced ($\gamma_g \ge \eta_g$). Non-violating groups use concave power-law values with $\gamma_g \in (0.3, 0.7)$ and costs with $\eta_g \in (1.1, 1.5)$ so that the splitting criterion remains informative at $p{=}0$. Up to $p{=}0.1$ the mean oracle value gap is 10.1% and the policy spends 3.4% below budget (Figure 8). At $p{=}0.25$ the gap doubles to 20.3% and the cost violation turns positive ($+3.5\%$, std 4.7%). From $p{=}0.5$ on, the global fit violates $\gamma < \eta$, the tree makes no split and keeps the reference policy, so the cost violation is zero and the gap, 36 to 42%, is the value the oracle gains by exploiting the increasing returns. This is the



(a) Oracle value gap (%) (b) Cost violation (c) ARI

Figure 7: Structural departures from a global power law (default simulator). Bars: mean over five seeds (± 1 std on first two panels). The power-law column still uses a latent-group oracle at depth 2, so gaps need not vanish.



(a) Oracle value gap (%) (b) Cost violation

Figure 8: Increasing returns in a fraction of latent groups ($\gamma \geq \eta$).

same failure as on raw MT-LIFT (Table 15), and the global check of $\gamma < \eta$ detects it before any tree is grown.

H.4.2 Non-Monotone Value Responses

We consider oversaturated responses of the form

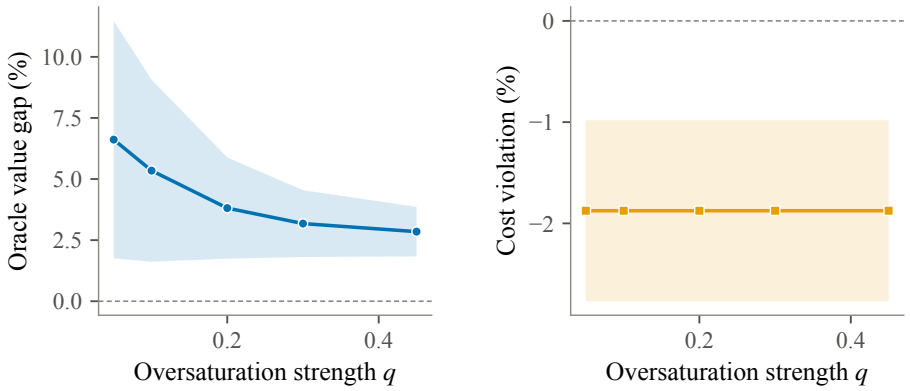
$$v_g(\alpha) = v_{0g}\alpha^{\gamma_g} \exp\big(-q_g(\alpha - 1)^2\big), \tag{39}$$

which peak and then decline.

Results. With $\eta_g \in (1.1, 1.5)$ on costs and the default oversaturation strength $q_g \in (0.05, 0.20)$ (five seeds, same sample sizes as above), the mean oracle value gap is 4.0% (std 2.8%), the mean cost violation -1.9%, and the mean ARI 0.38. With a single q for all groups (Figure 9), the gap falls from 6.6% at q=0.05 to 2.9% at q=0.45, and the cost violation stays at -1.9%.

H.5 Split clipping in the tree criterion (policy grid vs. wide range)

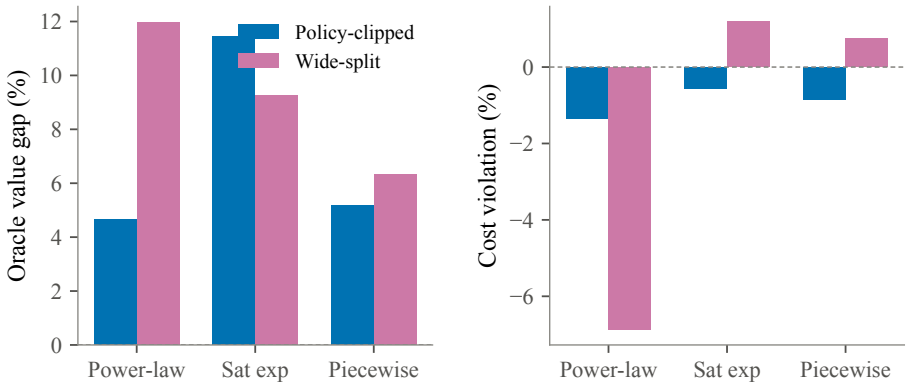
We distinguish two UVET variants that differ only in the treatment-bound interval used when scoring splits and computing per-leaf allocations. UVET_{cl} (*policy-clipped*) restricts the internal α solves to the convex hull of the randomized policies observed in the training draw, i.e. $[\alpha_{\min}, \alpha_{\max}] = [\min_k \alpha_k, \max_k \alpha_k]$. This is the default production setting because it never extrapolates beyond observed support. $\text{UVET}_{\text{uncl}}$ (*wide-split*) widens the clipping interval to $[0.001, 1.999]$ when solving



(a) Oracle value gap (%)

(b) Cost violation

Figure 9: Non-monotone synthetic value responses vs. oversaturation parameter q (± 1 std over five seeds).



(a) Oracle value gap (%)

(b) Cost violation

Figure 10: Split-phase clipping: *Policy-clipped* vs. *Wide-split* tree criterion (Section H); legend matches plot labels.

child allocations during recursive partitioning, as on the benchmarks, allowing the criterion to score splits that imply more aggressive treatment changes.

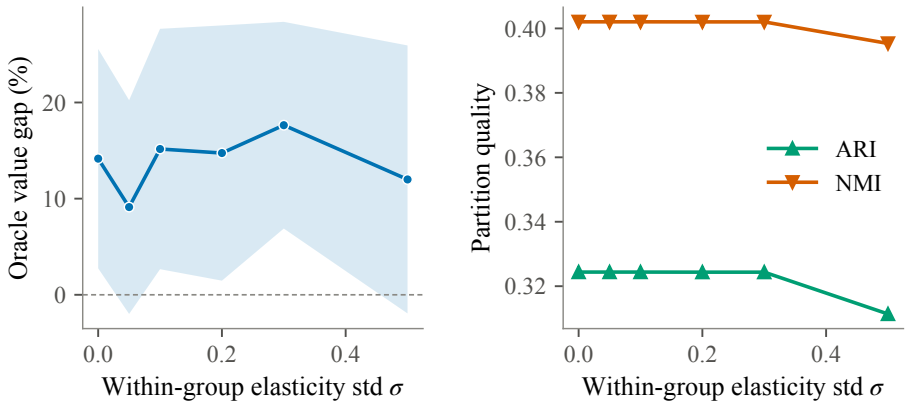
Results. Figure 10 compares the two modes on three generators at $D{=}2$. Neither dominates. Under the correct power law the wide range lowers the gap from 11.4% to 9.2% at a 1.2% overspend, and on saturating responses it raises the gap from 5.2% to 6.3%. On the piecewise generator it raises the gap from 4.6% to 12.0% and spends 6.9% below budget.

H.6 Within-Bucket Heterogeneity

We relax the assumption that a single elasticity governs each bucket by introducing individual-level heterogeneity:

$$\gamma_i \sim \mathcal{N}(\bar{\gamma}_g, \sigma_\gamma^2), \qquad \eta_i \sim \mathcal{N}(\bar{\eta}_g, \sigma_\eta^2).$$

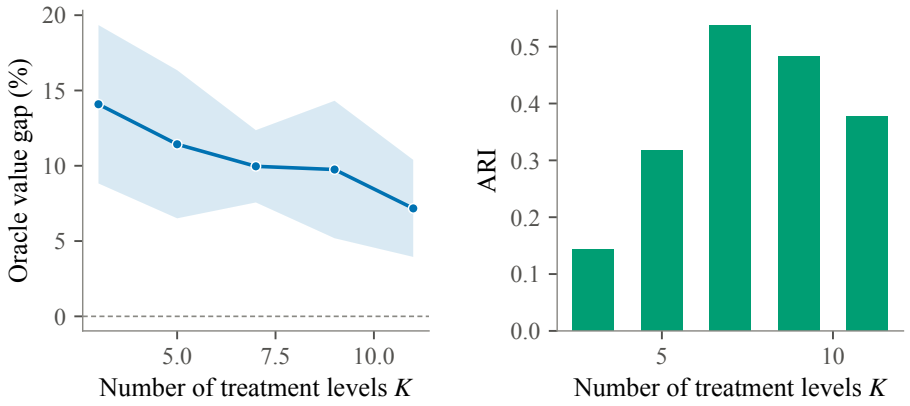
Results. As the within-group standard deviation σ grows from 0 to 0.5, the mean oracle value gap moves between 9% and 18% with no trend, inside a seed-to-seed std of 11 to 14 points, and ARI and NMI stay at about 0.32 and 0.40 (Figure 11). At this sample size, a spread of up to 0.5 around the group elasticities changes neither the partition nor the gap measurably.



(a) Oracle value gap (%)

(b) Partition recovery

Figure 11: Within-bucket elasticity noise (std. σ).



(a) Oracle value gap (%)

(b) Partition recovery

Figure 12: Discrete treatment support: number of randomized treatment levels K (Appendix H).

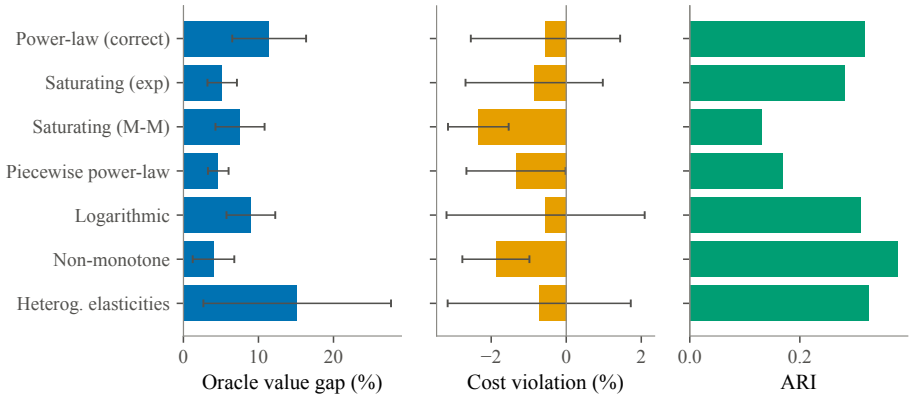
H.7 Number of Discrete Treatment Levels

We focus on varying the number of discrete treatment policies (balanced uniform randomization across policies). Sweeps over assignment imbalance and additional noise levels are left for future revisions of the simulator.

Results. The mean oracle value gap falls from 14.1% with three treatment levels to 7.2% with eleven, and ARI rises from 0.14 to 0.54 at seven levels before dropping to 0.38 at eleven (Figure 12). The grids also widen with K, from $[0.9, 1.1]$ to $[0.5, 1.5]$, so the sweep varies the number of levels and their range together.

H.8 Scenario overview

Figure 13 collects mean oracle value gap, cost violation, and ARI across the main synthetic stress tests (including saturating, heterogeneity, and non-monotone specifications) under a single evaluation protocol.



(a) Oracle value gap (%) (b) Cost violation (c) ARI

Figure 13: Robustness scenario overview (means ± one std where shown). Horizontal-axis labels are abbreviated single-line scenario names to avoid overlap.

H.9 Allocation Misspecification vs. Partition Misspecification

A sharper diagnostic would compare, on a fixed evaluation sample, (i) oracle welfare under the *learned* partition with per-leaf treatments re-optimized under the true responses, to (ii) the learned policy. This decomposition is not reported here: the current pipeline instantiates only the full-information oracle on latent groups (Appendix H.2.1).

H.10 Exception-Case Generators

Additional stressors (near-flat responses, train–test regime drift, aggressive elasticity shrinking) are natural extensions of the same simulator harness; we do not tabulate them in this version of the appendix.

H.11 Reproducibility of robustness experiments

Default settings. All robustness studies share a common backbone: G=4 latent groups with axis-aligned covariate rules, K=5 uniform-random treatment policies in $\{0.8, 0.9, 1.0, 1.1, 1.2\}$ unless varied, multiplicative log-normal noise on value and cost outcomes ($\sigma_v = \sigma_c = 0.1$), n_{train}=5000, n_{test}=10000, five replicates with seed offsets $\{0, 1, 2, 3, 4\}$ (training-set seed $42{+}r$, test-set seed $1042{+}r$), D=2 tree depth, minimum leaf size 200, and learned and oracle allocations bounded to $[\alpha_{\min}, \alpha_{\max}] = [0.5, 2.0]$. The increasing-returns and depth sweeps use 3 replicates instead of 5 to keep the total walltime tractable; per-scenario settings are given with each scenario above and in Appendix F.5.

Generators. Each scenario instantiates a distinct response family while keeping the rest of the backbone fixed: *Power-law (correct)* – the assumed model; *Saturating (exp)* – $v_g(\alpha) = v_{0g}(1 - e^{-k_g\alpha})$ with $k_g \in [0.3, 5]$; *Saturating (M-M)* – Michaelis–Menten; *Piecewise power-law* – two power-law segments joined at τ_g; *Logarithmic returns* – $v_g = v_{0g}\log(1 + k_g\alpha)$; *Increasing returns* – a fraction $p \in \{0, 0.25, 0.5, 0.75\}$ of groups violate $\gamma_g < \eta_g$; *Non-monotone* – power-law multiplied by $\exp(-q_g(\alpha - 1)^2)$; *Heterogeneous* – per-unit elasticities drawn around per-group means. Exact functional-form parameters are listed at the start of each scenario subsection.

Evaluation. Oracle value gaps use the constrained-welfare oracle on the true latent groups (Appendix H.2.1); cost violation as in Appendix H.2.2; partition recovery as ARI/NMI between learned leaves and latent groups (Appendix H.2.3). All metrics aggregate as mean±standard deviation across replicates; the same protocol is used for every scenario.

Summary. On saturating, piecewise, logarithmic and non-monotone responses, the oracle value gap is 4 to 9%, against 11.4% under the correct power law, the mCUV allocation stays within budget on average (mean violations from −2.3% to −0.6%), and ARI lies between 0.13 and 0.38. We do not separate allocation error from segmentation error. The method breaks down when a large share of groups has increasing returns, where it falls back to the reference policy.

I MT-LIFT data

Figure 14 compares the value uplift of every row of Table 1 on the training and test splits. Under LP and ILP every row loses part of its training gain on the test week, and UVET_{bay} keeps about half of it (22.9% on training, 12.5% on test, under LP). Under mCUV the training value carries over almost unchanged, for instance 8.4% to 8.0% for UVET_{cl}. LP and ILP pick, for each cohort, the logged treatment levels with the highest training totals, so part of their training gain is most likely noise in those totals that does not recur on new data; mCUV fits two parameters per cohort, and its allocation depends less on any single total. Cost carries over less well: every allocator meets the budget on the training split, ILP with some slack, and the test overruns are in Table 1 and Figures 2–4.

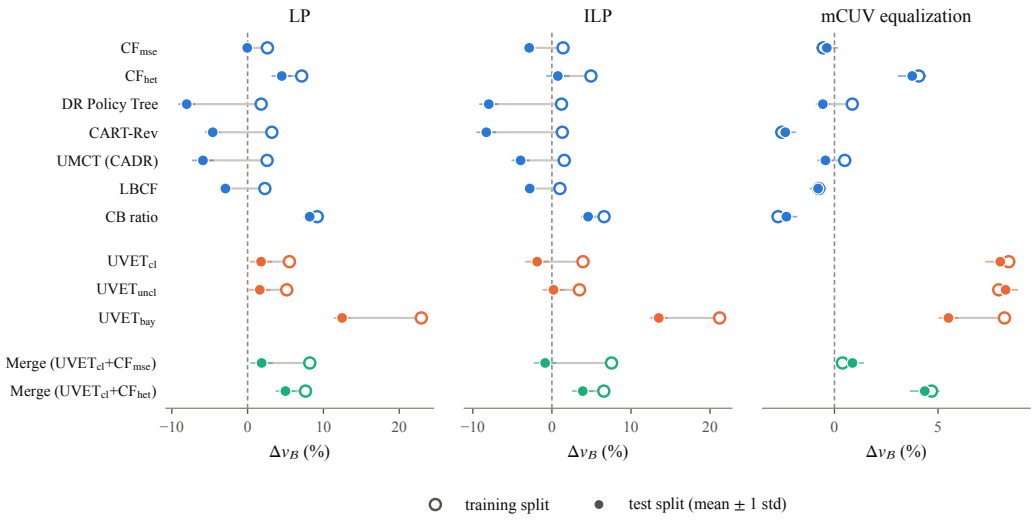


Figure 14: $\text{MT-LIFT}_{\text{scaled}}$, D=2: value uplift on the training split (open) and the test split (filled, mean ±1 bootstrap std, B=1000) for every row of Table 1, by allocator. LP and ILP lose much of their training gain on the test week, while mCUV keeps it.

Paired comparisons. Table 14 bootstraps the difference in LP Δv_B between UVET_{bay} and every other row of Table 1 on the same B=1000 test resamples. Every interval excludes zero; the closest row is the CB ratio, which overspends by 4.0%.

I.1 MT-LIFT preprocessing sensitivity

Table 1 reports results on $\text{MT-LIFT}_{\text{scaled}}$, where, after IPW normalization by the share of each treatment level, positive conversion values at treatment level α are multiplied by e^{m_α} with $(m_{0.8}, m_{0.9}, m_{1.0}, m_{1.1}, m_{1.2}) = (0.6, 0.2, 0, -0.1, -0.3)$, which enforces $\gamma < \eta$. To assess sensitivity to this choice, we run the same methods on two additional variants:

- **$\text{MT-LIFT}_{\text{iso}}$** applies isotonic regression on the aggregate value curve across treatments, enforcing monotonicity.
- **$\text{MT-LIFT}_{\text{raw}}$** preserves the original data with $\gamma > \eta$, violating UVET's diminishing-returns assumption. All methods operate under violated assumptions.

Table 14: MT-LIFT$_{\text{scaled}}$, LP allocator, D=2: paired bootstrap of the difference in $\Delta v_{\mathcal{B}}$ (percentage points) between UVET$_{\text{bay}}$ (12.5%) and each row of Table 1 or the no-split partition, over the B=1000 shared resamples of Table 1. ρ: correlation of the two rows across resamples.

Row	row $\Delta v_{\mathcal{B}}$ (%)	gap	95% interval	ρ
DR Policy Tree	-8.0	20.5	$[18.5, 22.4]$	0.59
UMCT (CADR)	-5.9	18.4	$[16.0, 20.6]$	0.62
CART-Rev	-4.6	17.1	$[15.0, 19.0]$	0.54
LBCF	-2.9	15.4	$[13.4, 17.3]$	0.50
CF$_{\text{mse}}$	0.0	12.5	$[10.5, 14.5]$	0.41
No split	0.0	12.5	$[10.4, 14.6]$	0.65
UVET$_{\text{uncl}}$	1.6	10.9	$[8.8, 12.9]$	0.69
UVET$_{\text{cl}}$	1.8	10.7	$[8.8, 12.6]$	0.73
Merge (UVET$_{\text{cl}}$ + CF$_{\text{mse}}$)	1.9	10.6	$[8.3, 12.9]$	0.65
CF$_{\text{het}}$	4.5	8.0	$[5.9, 10.0]$	0.65
Merge (UVET$_{\text{cl}}$ + CF$_{\text{het}}$)	5.0	7.5	$[5.4, 9.6]$	0.64
CB ratio	8.2	4.3	$[2.1, 6.5]$	0.39

Under MT-LIFT$_{\text{raw}}$, UVET is expected to degrade since its closed-form allocation relies on $\gamma < \eta$ (Proposition 1). The degree of degradation relative to two-stage HTE methods quantifies UVET's sensitivity to the power-law assumption.

Results Table 15 reports results for UVET and Causal Forest baselines across all three variants under mCUV equalization and LP allocation.

On MT-LIFT$_{\text{scaled}}$, where $\gamma < \eta$ holds by construction, UVET$_{\text{uncl}}$ attains the highest budget-feasible mCUV value uplift in Table 1 ($+8.3\%$, $\Delta c_{\mathcal{B}} \leq 0$); its LP uplift is $+1.6\%$, while UVET$_{\text{bay}}$ has the highest LP value, $+12.5\%$ at a 1.6% overspend. On MT-LIFT$_{\text{raw}}$, where $\gamma > \eta$, every UVET variant returns the no-split partition, whose LP value is -2.8%. Under LP allocation, which does not rely on power-law modeling, the causal forests reach $+15.0\%$ (CF$_{\text{het}}$) and $+22.4\%$ (CF$_{\text{mse}}$), at cost overruns of 3.9% and 3.4%; the mCUV allocator is unreliable on their partitions there (CF$_{\text{mse}}$ -18.5%), since the power-law premise fails. This confirms that UVET's advantage is tied to the power-law assumption holding, and that the LP allocator provides a robust fallback. MT-LIFT$_{\text{iso}}$ amplifies treatment effects through isotonic regression, producing artificially large LP uplifts ($> 100\%$) that are not directly comparable.

Table 15: MT-LIFT preprocessing variants (D=2): LP and mCUV $\Delta v_{\mathcal{B}}$ (%, metrics $\times 100$), mean over B=1000 test resamples; each variant is refit separately, and the CF baselines use the shadow-price target of Appendix F.3. The MT-LIFT$_{\text{scaled}}$ columns repeat Table 1.

	MT-LIFT$_{\text{scaled}}$		MT-LIFT$_{\text{raw}}$		MT-LIFT$_{\text{iso}}$	
Method	LP Δv (%)	mCUV Δv (%)	LP Δv (%)	mCUV Δv (%)	LP Δv (%)	mCUV Δv (%)
No split	0.0	0.0	-2.8	0.0	110.4	0.0
CF$_{\text{mse}}$	0.0	-0.4	22.4	-18.5	132.3	-22.9
CF$_{\text{het}}$	4.5	3.8	15.0	0.0	158.9	-11.6
UVET$_{\text{cl}}$	1.8	8.0	-2.8	0.0	110.4	0.0
UVET$_{\text{uncl}}$	1.6	8.3	-2.8	0.0	110.4	0.0
UVET$_{\text{bay}}$	12.5	5.5	-2.8	0.0	110.4	0.0

J UVET-benchmark data

Figure 15 compares the value uplift of every row of Table 2 on the training and test splits. Under LP most baseline partitions lose most or all of their training gain on the test split (UMCT, for instance, falls from 0.6% to 0.0%), while the UVET variants keep theirs (0.7% to 0.5% for UVET$_{\text{cl}}$, 0.9% to 1.0% for UVET$_{\text{uncl}}$, 0.6% to 0.6% for UVET$_{\text{bay}}$). The DR Policy Tree is the exception among

the baselines and gains more on the test split (1.0%) than on training (0.6%). Their leaves separate cohorts by value elasticity (Table 16), and that contrast carries over to the test rows. Under mCUV most rows keep their training value, as on MT-LIFT$_{\text{scaled}}$ (Appendix I).

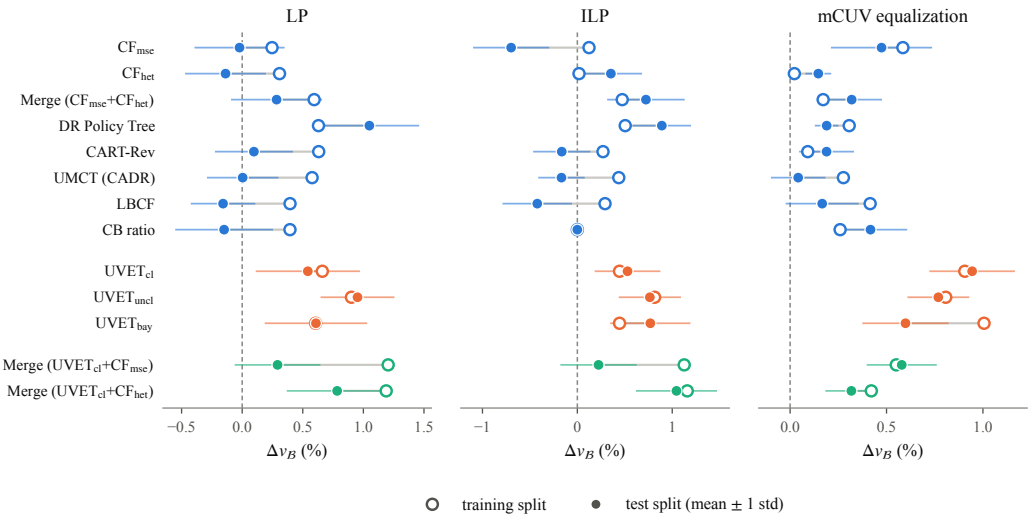


Figure 15: UVET-benchmark, D=2: value uplift on the training split (open) and the test split (filled, mean ± 1 bootstrap std, B=1000) for every row of Table 2, by allocator. Under LP most baseline partitions lose most of their training gain on the test split, while the UVET variants keep theirs.

Continuous intensities between logged treatment levels. The bid multiplier is continuous: any $\alpha \in A = [0.5, 1.5]$ scales the reference bids and can be deployed, while the A/B test logs only the five treatment levels $\{0.8, \ldots, 1.2\}$. Table 16 lists, for the UVET$_{\text{cl}}$ and UVET$_{\text{uncl}}$ partitions of Table 2, the leaf elasticities and the intensity α_b^* returned by the closed form (8) under the iso-cost constraint, next to the LP and ILP allocations over the logged treatment levels. Two leaves receive an intensity strictly between logged treatment levels (0.879 and 1.032, in bold); the LP allocator can only pick a logged level or mix them (0.9 for the UVET$_{\text{cl}}$ leaf, a mix of 0.9 and 1.0 for the UVET$_{\text{uncl}}$ leaf), and the ILP allocator assigns a single level (1.1 and 1.0). Offline evaluation is restricted to the logged treatment levels (Appendix E), so these intermediate intensities are scored through the fitted power law rather than measured directly. Appendix H.7 shows that the fit remains usable with as few as three logged levels.

Table 16: *UVET-benchmark*, D=2: per-leaf training share of rows, log–log elasticities, closed-form mCUV intensity α_b^* (clipped to $[0.8, 1.2]$), LP mixture over logged treatment levels (weight $\times$ level), and ILP level, all fitted on the training split.

Partition	leaf	rows (%)	$\hat{\gamma}_b$	$\hat{\eta}_b$	α_b^* (mCUV)	LP mixture	ILP
UVET$_{\text{cl}}$	1	23.6	0.119	1.028	**0.879**	1.00×0.9	1.1
	2	50.1	0.259	1.220	1.200	$0.04\times1.0 + 0.96\times1.2$	1.0
	3	13.2	0.036	1.041	0.800	1.00×0.8	0.8
	4	13.1	0.083	1.027	0.800	1.00×0.8	1.0
UVET$_{\text{uncl}}$	1	49.9	0.152	1.085	**1.032**	$0.22\times0.9 + 0.78\times1.0$	1.0
	2	24.6	0.359	1.316	1.200	1.00×1.2	1.2
	3	12.8	0.102	1.066	0.800	1.00×1.0	0.9
	4	12.8	0.020	1.003	0.800	1.00×0.8	0.8

Paired comparisons. Because every row of Table 2 is evaluated on the same B=1000 test resamples, the difference in LP $\Delta v_{\mathcal{B}}$ between two rows can be bootstrapped directly, and the covariance

between them enters by construction. Table 17 anchors the comparison on $\text{UVET}_{\text{uncl}}$, the pure variant with no merging step, and reports 95% percentile intervals of the paired difference. Six of the eight baselines fall below $\text{UVET}_{\text{uncl}}$ with intervals that exclude zero. The CF-only merge does not separate (gap 0.67, $[-0.18, 1.50]$), and neither does the DR Policy Tree, which gains 0.10 points more at a 0.38% overspend ($[-0.71, 0.53]$). No row built with a UVET partition separates either; UVET_{bay} trails by 0.34 points, $[-0.25, 0.93]$. The rows that share the UVET_{cl} partition are correlated with $\text{UVET}_{\text{uncl}}$ (correlation 0.45 to 0.76), yet their intervals still extend half a point or more on either side of the gap, so differences of that size in either direction remain compatible with the data and we claim no ranking among the UVET-based rows. The last two rows hold one input of a merge fixed and replace UVET_{cl} by the other CF partition. Removing the UVET input lowers LP value by 0.50 points in one merge and leaves it unchanged in the other (0.01), and neither interval excludes zero.

Table 17: *UVET-benchmark*, LP allocator, D=2: paired bootstrap of the difference in $\Delta v_{\mathcal{B}}$ (percentage points) over the B=1000 shared resamples of Table 2. "Gap" is $\text{UVET}_{\text{uncl}}$ minus the row (top block) or the UVET-derived merge minus the CF-only merge (bottom block). ρ: correlation of the two rows across resamples.

Comparison	row $\Delta v_{\mathcal{B}}$ (%)	gap	95% interval	ρ
$\text{UVET}_{\text{uncl}}$ (0.95%) against each row				
LBCF	−0.16	1.11	[0.42, 1.85]	0.17
CB ratio	−0.15	1.10	[0.29, 1.92]	0.29
CF_{het}	−0.14	1.09	[0.40, 1.79]	0.41
CF_{mse}	−0.02	0.97	[0.15, 1.82]	0.21
UMCT (CADR)	0.00	0.95	[0.18, 1.69]	0.17
CART-Rev	0.10	0.85	[0.19, 1.52]	0.41
Merge (CF_{mse} + CF_{het})	0.28	0.67	[−0.18, 1.50]	0.21
Merge (UVET_{cl} + CF_{mse})	0.29	0.66	[−0.02, 1.30]	0.45
UVET_{cl}	0.54	0.41	[−0.14, 0.94]	0.76
UVET_{bay}	0.61	0.34	[−0.25, 0.93]	0.71
Merge (UVET_{cl} + CF_{het})	0.78	0.17	[−0.50, 0.77]	0.64
DR Policy Tree	1.05	−0.10	[−0.71, 0.53]	0.65
UVET input removed from a merge				
Merge (UVET_{cl} + CF_{het}) vs. Merge (CF_{mse} + CF_{het})		0.50	[−0.27, 1.22]	
Merge (UVET_{cl} + CF_{mse}) vs. Merge (CF_{mse} + CF_{het})		0.01	[−0.70, 0.68]	

Training target of the `rev`-based baselines. Table 18 refits every baseline that trains on `rev` with cost priced at $\hat{\gamma}/\hat{\eta}$ alone instead of the shadow price of Appendix F.3, on the same resamples. Under the shadow price $\text{UVET}_{\text{uncl}}$ separates from six of the eight baselines, all but the CF-only merge and the DR Policy Tree. Under the $\hat{\gamma}/\hat{\eta}$ price it separates from seven: the CF-only merge and the DR Policy Tree fall below it (the DR Policy Tree from 1.05% to −0.01%), and the CB ratio, which rises to 0.29%, becomes the only exception. The merges built with UVET_{cl} move by at most 0.40 points.

K UVET variants and practitioner guidance

This appendix documents all public UVET variants (clipped, unclipped, Bayesian, forest, Bayesian–forest) and the UVET-derived merged partition, default behaviors, and when each is appropriate, together with allocator-choice guidance. Table 19 is the compact reference; subsections below give context.

K.1 When to use UVET

UVET is appropriate when the following conditions hold:

1. **Ordered or continuous treatment**: the treatment admits an intensity interpretation (e.g., bid multiplier, discount level, dosage) rather than being purely categorical.

Table 18: *UVET-benchmark*, LP allocator, D=2: `rev`-based rows under the shadow-price target used in Table 2 and under the price $\hat{\gamma}/\hat{\eta}$. "Gap" is $\text{UVET}_{\text{uncl}}$ (0.95%) minus the row, paired over the B=1000 shared resamples, with its 95% interval. UMCT and the UVET variants do not train on `rev`.

	shadow price $\hat{\lambda}$ (Table 2)			price $\hat{\gamma}/\hat{\eta}$		
Row	$\Delta v_{\mathcal{B}}$ (%)	gap	95% interval	$\Delta v_{\mathcal{B}}$ (%)	gap	95% interval
CF_{mse}	-0.02	0.97	$[0.15, 1.82]$	-0.18	1.14	$[0.42, 1.92]$
CF_{het}	-0.14	1.09	$[0.40, 1.79]$	-0.15	1.10	$[0.48, 1.76]$
Merge (CF_{mse} + CF_{het})	0.28	0.67	$[-0.18, 1.50]$	-0.13	1.08	$[0.35, 1.88]$
DR Policy Tree	1.05	-0.10	$[-0.71, 0.53]$	-0.01	0.96	$[0.38, 1.62]$
CART-Rev	0.10	0.85	$[0.19, 1.52]$	0.25	0.70	$[0.16, 1.25]$
LBCF	-0.16	1.11	$[0.42, 1.85]$	-0.33	1.28	$[0.54, 1.97]$
CB ratio	-0.15	1.10	$[0.29, 1.92]$	0.29	0.67	$[-0.01, 1.36]$
Merge (UVET_{cl} + CF_{mse})	0.29	0.66	$[-0.02, 1.30]$	0.69	0.26	$[-0.46, 0.95]$
Merge (UVET_{cl} + CF_{het})	0.78	0.17	$[-0.50, 0.77]$	0.66	0.30	$[-0.32, 0.84]$

2. **Diminishing returns**: higher treatment intensity yields increasing value and cost, but with decreasing marginal returns—i.e., the power-law assumption $\gamma_b < \eta_b$ is plausible.
3. **Cost constraint**: the problem involves a fixed budget or iso-cost requirement, so that value must be maximized subject to total cost being held constant.
4. **Randomized data**: at least two distinct treatment levels have been randomly assigned in each candidate split region (e.g., from an A/B/n test).
5. **Scale**: the dataset is large enough ($\gtrsim 10^5$ samples) for reliable elasticity estimation at the leaf level.

When any of these fail, consider alternatives:

- If treatment is *binary* or *categorical without order*, the scale is small, or per-inference optimization is affordable: use a causal forest or uplift model (e.g., S-/T-/X-Learner) followed by an LP or knapsack allocator.
- If there is *no cost constraint*: standard CATE-based methods (CF_{het}, CF_{mse}) may suffice since the mCUV equalization machinery provides no benefit without a budget.
- If diminishing returns are *violated* ($\gamma > \eta$): on $\text{MT-LIFT}_{\text{raw}}$ every UVET variant returns the no-split partition while causal forests with the LP allocator reach $+15.0\%$ to $+22.4\%$ (Table 15), so an HTE partition with the LP allocator is the better choice.
- If the data is *observational*: elasticity estimates may be confounded; debiasing (e.g., IPW) is required before applying UVET.

K.2 Choosing among UVET variants

Table 19 summarizes the available variants and when each is recommended.

K.3 Bayesian shrinkage strength λ_0 and minimum leaf sizes

The prior strength λ_0 in (12) puts weight $\lambda_0/(\lambda_0 + L)$ on the root-level elasticity in every node, trading local signal for lower variance. The gain of UVET_{bay} depends on this choice (Appendix G.6.2), so we recommend choosing λ_0 from a small grid by validation S_{comb} (Appendix G.6), the procedure that selects λ_0=20 on $\text{MT-LIFT}_{\text{scaled}}$ and λ_0=10 on UVET-benchmark. On $\text{MT-LIFT}_{\text{scaled}}$ the test gain is large only for λ_0 between 15 and 20, so a finer grid around the validation choice is worth its cost; on UVET-benchmark the choice matters little. The minimum leaf size can stay at $\lfloor |\mathcal{X}^{\text{tr}}|/2^{D+2} \rfloor$ (Appendix F.1): floors from 0.5 to 1.5 times that value return the same depth-2 partitions on both datasets.

Table 19: Decision guide for selecting among UVET variants.

Variant	Description	When to prefer
UVET_{cl}	Optimal α_b^* is clipped to $[\alpha_{\min}, \alpha_{\max}]$.	Default choice in practice. Use when the system enforces bounds on treatment intensity (e.g., bid multipliers cannot exceed a safety range). More robust to elasticity estimation noise.
$\text{UVET}_{\text{uncl}}$	Splits are scored with α_b^* clipped only to $[0.001, 1.999]$, i.e. effectively unconstrained; allocation is clipped to the logged range $[0.8, 1.2]$.	Use when the treatment range is wide and the power-law holds across a broad region. Highest LP value on UVET-benchmark (Table 2); may favor splits whose unclipped optimum lies outside the logged range.
UVET_{bay}	Elasticity parameters are estimated with priors, shrinking noisy estimates toward a global mean.	Largest gain on $\text{MT-LIFT}_{\text{scaled}}$. Prefer UVET_{cl} or $\text{UVET}_{\text{uncl}}$ when heterogeneity is weak: on UVET-benchmark UVET_{bay} reaches $+0.61\%$ LP value against $+0.95\%$ for $\text{UVET}_{\text{uncl}}$, a gap the paired test does not resolve (Table 17).
$\text{UVET}_{\text{forest}}$	Ensemble of UVET trees with subsampling (released in the code; not evaluated in this paper).	Use for improved stability and reduced overfitting when the dataset is large and features are numerous. Sacrifices single-tree interpretability.
$\text{UVET}_{\text{bay-forest}}$	Combines Bayesian elasticity estimation with forest ensembling (released in the code; not evaluated in this paper).	Use for maximum robustness when both noise and model instability are concerns.
Merge (UVET_{cl} + CF)	Intersection of a UVET_{cl} partition with a CF_{mse} or CF_{het} partition (Appendix F.4).	Use when a finer partition is acceptable and a causal-forest view of heterogeneity should be kept. On UVET-benchmark its LP value (0.29–0.78%) does not separate from $\text{UVET}_{\text{uncl}}$ (Table 17); under mCUV it overspends by at most 0.03%.

K.4 Choosing the downstream allocator

Once a partition is learned, three allocators can assign treatments:

- **LP** (linear programming, default for benchmarking): use when comparing partitioning methods fairly—LP does not assume a power-law form and optimizes over the observed discrete policies per bucket. Stochastic assignment (probabilities per policy) often yields higher value than ILP.
- **ILP** (integer programming): use when each bucket must receive exactly one discrete treatment level. Strong discrete benchmark; combinatorial cost.
- **mCUV equalization** (closed-form): use when the power-law assumption is trusted and continuous intensities are desired. Fastest at inference; pairs naturally with UVET's split criterion but should not be the sole metric when comparing against HTE-based partitions (Appendix G.5).

K.5 Quick-start decision flowchart

1. Is the treatment ordered/continuous with diminishing returns? $\rightarrow$ If no, use CATE-based methods.
2. Do you have randomized data with ≥ 2 treatment levels? $\rightarrow$ If no, collect experimental data or apply debiasing.
3. Is there a binding budget/cost constraint? $\rightarrow$ If no, simpler methods may suffice.
4. Is the treatment set discrete with few levels? $\rightarrow$ Use UVET_{cl} with LP or ILP allocator for evaluation; LP is the default fair comparator across methods.
5. Is the treatment set continuous or is the range wide and is the power-law trusted? $\rightarrow$ Use UVET_{cl} with mCUV allocator at deployment; still report LP/ILP in experiments for comparability.
6. Is elasticity estimation noisy (small subgroups, binary outcomes)? $\rightarrow$ Use UVET_{bay} or $\text{UVET}_{\text{bay-forest}}$.
7. Do you need maximal stability over interpretability? $\rightarrow$ Use $\text{UVET}_{\text{forest}}$ or $\text{UVET}_{\text{bay-forest}}$.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state the two contributions, the UVET tree with its closed-form allocation and the UVET-benchmark dataset, and report the empirical results with their scope: the large gain is on MT-LIFT$_{\text{scaled}}$, a variant rescaled into the diminishing-returns regime, and the gains on UVET-benchmark are small. Sections 2–4 and Appendix E support these claims.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Section 6 has a Limitations paragraph: UVET needs randomized data with at least two treatment levels per candidate region and assumes $\gamma_b < \eta_b$; leaf elasticities are noisy, and the gain of UVET$_{\text{bay}}$ depends on the shrinkage strength; splits are greedy, with no global optimality guarantee; the mCUV allocation can overspend under misspecified responses; and the largest gain is on a semi-synthetic variant.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best

judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: All propositions and Theorem 1 state their assumptions explicitly; proofs are in Appendix C.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Section 4 and Appendices F–G specify the data splits, hyperparameters, seeds and evaluation protocol. The code, with a script that reruns every experiment, is released at `https://github.com/criteo-research/uvet` under the Apache 2.0 license, and UVET-benchmark is on Hugging Face.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.

(c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: UVET-benchmark is on Hugging Face (https://huggingface.co/datasets/criteo/uvet-benchmark) under CC BY-NC-SA 4.0, with a dataset card. The code is at https://github.com/criteo-research/uvet under the Apache 2.0 license, with a README, a guide that maps every table and figure to the command that produces it, and a script that reruns all experiments.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://neurips.cc/public/guides/CodeSubmissionPolicy) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Section 4 gives the data splits, tree depth, clipping bounds and evaluation metrics. Appendices F.1 and F.4 give all hyperparameters; the shrinkage strength λ_0 is a fixed default, with its sensitivity and a validation choice in Appendix G.6.2.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: Tables 1 and 2 report the mean $\pm$ one standard deviation over $B{=}1000$ bootstrap resamples of the test split, shared by all methods, and Tables 14 and 17 report paired 95% percentile intervals for differences between methods. These intervals capture test-set sampling variability. Each partition is fit once: the UVET trees are deterministic given the training split, and the baselines use fixed seeds (Appendix F.4), so variability over training draws is not included. The robustness experiments of Appendix H average over five seeds (three for two of the sweeps), and the sweeps of Appendix F.5 use one seed.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: Appendix F.6 lists the hardware, per-method fit and bootstrap wall times, and total CPU-hours; no GPU is used.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

Answer: [Yes]

Justification: The research uses anonymized advertising data released under appropriate licenses. No human subjects were involved beyond standard A/B testing practices.

Guidelines:

- The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: **Positive impacts:** Better offline methods for budget-constrained treatment assignment can improve the statistical efficiency of randomized experiments (fewer impressions needed to learn safe policies), support transparent cohort-level rules in regulated or high-stakes domains (e.g. clinical dosing, energy demand response) where interpretable segmentation matters, and reduce wasted spend when interventions have diminishing returns, which frees budget for broader access or lower prices. Releasing UVET-benchmark lowers the barrier to reproducible research on dual-outcome, multi-treatment industrial data. **Negative impacts and mitigations:** Like any targeting technology, improved allocation could concentrate benefits (e.g. ad spend on already profitable users) or, if elasticities are misspecified, shift resources incorrectly; fairness is not explicitly constrained in UVET, so protected-attribute correlations in covariates could interact with splits. We flag greedy partitioning and RCT assumptions as limitations (Conclusion) and point to constrained-tree literature for extensions. The benchmark is anonymized (Appendix E), which limits re-identification risk.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

Justification: The released dataset (UVET-benchmark) contains anonymized advertising auction data with no personally identifiable information. The algorithm does not pose a high risk of misuse.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: MT-LIFT (Huang et al., 2024a) is used as released by its authors, whose repository states no license. Baselines and solvers are cited and used under their licenses: EconML (MIT), scikit-learn (BSD-3-Clause), and cvxpy (Apache 2.0) with the ECOS and CVXOPT/GLPK solvers (GPL-3.0). UVET-benchmark is released under CC BY-NC-SA 4.0 (Appendix E) and our code under Apache 2.0.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: UVET-benchmark is documented in Appendix E and in its dataset card on Hugging Face: collection, anonymization, schema, splits, treatment levels, intended use, limitations and license. The code comes with a README and a reproduction guide.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.

- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: No crowdsourcing or human subjects research was conducted.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: No human subjects research was conducted; the data comes from standard A/B tests on an advertising platform.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: LLMs are not used as a component of the UVET algorithm or experimental methodology.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.