CUVET: A Partitioning Approach for Continuous Treatment Assignment At Scale

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

Abstract

Treatment assignment problems arise wherever limited budget must be allocated to heterogeneous users, with applications ranging from personalized recommendations to online advertising and healthcare. In such settings, individuals exhibit heterogeneous responses to different treatments, making it essential to learn cost-aware personalized treatments. This paper introduces the **Cost per Unit Value Equalization Tree (CUVET)** algorithm, a novel treatment assignment approach that partitions the user space. Under a diminishing-returns (power-law) assumption, it solves the treatment assignment problem by equalizing the marginal cost per unit value across each user group. This leads to a **closed-form cost-aware treatment assignment solution**, making it particularly suited for large-scale applications such as digital advertising. We also release **CUVET-policy, a** 87**-million-impression public benchmark derived from real-world industrial A/B tests**, providing an open-source evaluation framework for decision-focused learning. On both CUVET-policy and the public MT-LIFT dataset, CUVET **significantly improves baselines' total value by +1% and +12.5%** respectively, satisfying budget constraints.[2]

1 Introduction

Treatment assignment problems are fundamental to decision-making because they involve allocating scarce resources to maximize outcomes. The significance of these problems arises from their direct impact on efficiency, productivity, equity, and sustainability, whether for personalized online recommendations (Tu et al., 2021; Betlei et al., 2024), energy consumption forecasting (Wahdany et al., 2023), food delivery platforms (Huang et al., 2024). Traditional machine learning-based methods excel at modeling correlations, but often fail to correctly identify the underlying causal relationship between treatment and outcome. In most of these problems, the relationship between resource allocation and the resulting value exhibits *diminishing returns* or non-linear scaling behavior (Shephard and Färe, 1974; Hulatt, 2023; Mongeon et al., 2016).

Our contributions We propose a new method, the Cost per Unit Value Equalization Tree (CUVET) approach, that leverages the economic nature of the value and cost functions under the power-law elasticity assumption, resulting in a *closed-form downstream solution* as in a policy learning approach. We empirically demonstrate the advantages of the proposed method for user-space bucketization and policy learning over baseline approaches on synthetic and real-world data.

[*]The work was done during the master internship at Criteo AI Lab.
[2]The dataset is available on Hugging Face platform under the title `CUVET-policy`.

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: Uncovering Causality in Science (CauScien).

2 The CUVET approach

Usually, the criterion we want to maximize (see Appendix C for a detailed problem formulation) is not expressible in a closed-form way using estimates for a given partition. However *the elasticity assumption* on the value and cost, often observed in real data, enables this criterion to be expressed directly from offline data estimates (see Appendix C.2 for details), 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.

Treatment assignment problem We consider treatments parameterized continuously as $\mathcal{T} = \{t_0(\alpha),\ \alpha \in [\alpha_{\min}, \alpha_{\max}]\}$, with $t_0(1)$ the reference treatment. The user space $\mathcal{X}$ is partitioned into buckets $\mathcal{B}$ via a mapping $\pi : \mathcal{X} \to \mathcal{B}$. For each bucket $b \in \mathcal{B}$, we denote the expected cost and value outcomes under treatment α as $(c_b(\alpha), v_b(\alpha))$. We assume a *power-law elasticity* within each bucket: $v_b(\alpha) = v_b(1)\,\alpha^{\gamma_b}, c_b(\alpha) = c_b(1)\,\alpha^{\eta_b}$, with elasticity parameters $\gamma_b < \eta_b$ ensuring concavity and thus capturing diminishing returns. Our treatment assignment for any given partition should respect total fixed budget C. Since $v(\cdot)$ is assumed to be an increasing function of $c(\cdot)$, the maximum value under cost constraint is always reached at the constraint limit. Thus, we write the optimization problem as a strict iso-cost constraint with a total budget C: $\sum_b c_b(\alpha_b) = C$. Overall, our optimization problem is written as: $\max_{\phi:\mathcal{B}\to\mathcal{T}} \sum_{b\in\mathcal{B}} v_b(\phi(b))\ \ s.t.\ \ \sum_{b\in\mathcal{B}} c_b(\phi(b)) = C$, In the following, for all $b \in \mathcal{B}$, we will denote $\alpha_b = \phi(b)$.

2.1 Marginal Cost equalization

In econometric theory, we describe a market as being at the equilibrium when the market supply and market demand curves meet. As a result this allows to estimate a price and quantity at the equilibrium. In competitive markets, players should set their marginal cost equal to this price. Now, we can see our setting as a market where each bucket is a player and the outcome value is the quantity. At equilibrium, the marginal cost per unit of value in every bucket should be equal to the equilibrium price. We can therefore write that a treatment assignment enables the market to be at the equilibrium when marginal cost per unit of value is equal across buckets to fixed value μ (to be computed later):

$$\forall b, b' \in \mathcal{B} \quad mCUV_b(\alpha_b) = mCUV_{b'}(\alpha_{b'}) := \mu, \tag{1}$$

where for all $b \in \mathcal{B}$, $mCUV_b(\alpha_b)$ is the marginal cost of unit value in bucket b, evaluated for $\alpha = \alpha_b$ with $\forall \alpha \in A, \quad mCUV_b(\alpha) = {}^{\partial c_b}/_{\partial v_b}(\alpha)$.

This yields a closed-form expression for the optimal α_b provided in Eq. (16).

We demonstrate in Appendix E.2 that the equalization of mCUV is equivalent to the Lagrangian multiplier optimization method.

Additionally, we want our allocation ϕ to respect the iso-cost constraint. This characterizes entirely the value of μ which is used in (1) according to the following implicit equation:

$$\sum_b c_b(1) \left(\frac{\mu}{mCUV_b(1)}\right)^{\frac{\eta_b}{\eta_b - \gamma_b}} = C. \tag{2}$$

Since $\forall b,\ \eta_b > \gamma_b$, the function $f : x \mapsto \sum_b c_b(1) \left(\frac{\mu}{mCUV_b(1)}\right)^{\frac{\eta_b}{\eta_b-\gamma_b}}$ is continuous and increasing and diverges in both infinities,which guarantee the existence of a unique solution μ to (2). The value of μ can be determined with an optimization solver (for example by dichotomy or Brent's method).

2.2 CUVET 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 (6). 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. The decision tree approach circumvents this problem by adopting a greedy, recursive

Algorithm 1: CUVET 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`$(\mathbf{X}, v, c, \alpha)$**:**

  `// Function to build the tree recursively`

  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` $\sum_b v_b(\alpha_b^*)$

    $\mathbf{X}^{\text{left}} \leftarrow \mathbf{X} \cap \mathbb{I}[x_i \leq s], \quad \mathbf{X}^{\text{right}} \leftarrow \mathbf{X} \setminus \mathbf{X}^{\text{left}}$;

    Estimate $(\eta_b, \gamma_b, \alpha_b, mCUV_b(1))$ for $b \in \{\text{left}, \text{right}\}$;

    Compute μ from (1);

    Compute α_b^* for $b \in \{\text{left}, \text{right}\}$, see (7);

    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

approach to partition exploration. We assume access to a `candidate_generator` function that produces candidate splits expressed as a tuple (feature, threshold), see more details in appendix. The splitting criterion is the one presented in (8) 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 parameters `max_depth`, `min_samples_split`, `min_samples_leaf` the definitions of which are given in appendix. Appendix E.2) states the optimality of the greedy mCUV-equalization tree.

3 Experiments

We provide a comprehensive evaluation of our CUVET algorithm and several baselines on two large-scale, real-world datasets. We also provide an illustrative synthetic example which demonstrates the blindspot of HTE-based baselines (which systematically find a suboptimal solution) while the CUVET algorithm finds the optimal one in Appendix F. Baselines are discussed in Appendix H.2.

Evaluation As treatment assignment is the targeted downstream problem, we estimate the quality of produced assignments by comparing total value and total cost generated by the assignment with those of the reference treatment. For any bucketization $\mathcal{B}$, we learn the optimal assignment $\phi_{\mathcal{B}}^* : \mathcal{B} \to \mathcal{T}$ and compute the total value and cost of this assignment under the power-law assumption (6): $v(\phi_{\mathcal{B}}^*) = \sum_{b \in \mathcal{B}} v_b(\phi_{\mathcal{B}}^*(b))$, $c(\phi_{\mathcal{B}}^*) = \sum_{b \in \mathcal{B}} c_b(\phi_{\mathcal{B}}^*(b))$. Denote the reference total value by $v(1) = \sum_{b \in \mathcal{B}} v_b(1)$ and reference total cost by $c(1) = \sum_{b \in \mathcal{B}} c_b(1)$. Our final metrics of interest are

relative uplift in value and cost, $\Delta v_{\mathcal{B}}$ and $\Delta c_{\mathcal{B}}$ respectively:

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

Note that the greater $\Delta v_{\mathcal{B}}$ the better while for the cost we have the constraint $\Delta c_{\mathcal{B}} \leq 0$.

3.1 MT-LIFT data

This public[3] dataset was collected from two months of randomized controlled trials on coupon marketing scenarios for food delivery in the Meituan – China's local living platform (Huang et al., 2024). The dataset contains nearly 5.5M instances, 99 features, 5 treatments (coupons), and 2 labels: click and conversion. See Appendix G.1 for preprocessing details.

Results Results for $\Delta v_{\mathcal{B}}$ and $\Delta c_{\mathcal{B}}$ with respect to all evaluation methods are shown in Table 1 (graphical representation is deferred to Appendix I). Overall, CUVET consistently outperforms baselines across all evaluation methods. On mCUV eq. evaluation, CUVET$_{uncl}$ achieves **+12.5% total value compared to the reference treatment**, though at the cost of high constraint violations. This aligns with our earlier discussion on clipping: since the unclipped version allows $\alpha^* > 0$, it can explore a wider range of values, potentially leading to higher returns but also exceeding cost constraints. Conversely, CUVET$_{cl}$ appears more risk-averse. Under ILP evaluation, while CUVET$_{cl}$ achieves the highest $\Delta v_{\mathcal{B}}$, results appear noisier. This is expected, as discretizing the optimal treatment space (rather than allowing continuity) introduces variability. On LP evaluation, both CUVET$_{uncl}$ and CF-Het perform similarly, though the former exhibits higher variance. LP results surpass ILP due to stochasticity, enabling smoother optimization and better generalization.

High-variance performance on MT-LIFT data indicates that capturing user behavior changes under various treatments is difficult due to inherent noise present in the data. It would be useful to compare results on a larger data volume to assess whether the observed trends hold consistently.

Table 1: MT-LIFT data: relative uplifts in v and c on test split. Highest $\Delta v_{\mathcal{B}}$ result reported in bold.

Evaluation	Metric	CF-Het	CF-MSE	CUVET$_{cl}$	CUVET$_{uncl}$
mCUV eq.	$\Delta v_{\mathcal{B}}$	0.0054 ± 0.0084	0.0063 ± 0.0065	0.0791 ± 0.0154	$\mathbf{0.1254 \pm 0.0081}$
	$\Delta c_{\mathcal{B}}$	0.0094 ± 0.0032	0.0049 ± 0.0042	0.0230 ± 0.0062	0.0411 ± 0.0053
ILP	$\Delta v_{\mathcal{B}}$	-0.0007 ± 0.0228	0.0 ± 0.0	$\mathbf{0.0069 \pm 0.0382}$	-0.0062 ± 0.0381
	$\Delta c_{\mathcal{B}}$	-0.0234 ± 0.0083	0.0 ± 0.0	0.0185 ± 0.0125	0.0098 ± 0.0126
LP	$\Delta v_{\mathcal{B}}$	0.0344 ± 0.0284	-0.0221 ± 0.0151	0.0085 ± 0.0379	$\mathbf{0.0345 \pm 0.0442}$
	$\Delta c_{\mathcal{B}}$	0.0265 ± 0.0099	-0.0078 ± 0.0057	0.0210 ± 0.0124	0.0155 ± 0.0126

3.2 CUVET-policy data

We publish the *CUVET-policy* benchmarking dataset[4]. It was generated by an online advertising platform that conducted a two-week A/B test of 5 different treatments. Data have been properly anonymized so as to not disclose any private information, see Appendix G.2 for details. The dataset contains 86.7M samples and each sample represents a bidding opportunity for which a multi-dimensional context $x \in \mathbb{R}^5$ is observed. Let $\{\alpha_k\}_k$ be the set of possible treatments (*i.e.* different bidding strategies) parameters. A treatment with parameter α_k corresponds to uniformly multiplying the bid values of the reference treatment t_0. Therefore, our treatment set is defined in that case as $\mathcal{T} = \{\alpha \cdot t_0, \quad \alpha \in A\}$ where $A = [0.5, 1.5]$. The treatment parameters which are present (randomly assigned) in the dataset are $\{\alpha_1, \alpha_2, \alpha_0, \alpha_3, \alpha_4\} = \{0.8, 0.9, 1, 1.1, 1.2\}$ (where 1 corresponds to the reference treatment), restricting ourselves in our assignment problem. The value represents an advertising objective. Particular care has been taken to guarantee that each sample (x, α, v, c) is independent. The goal is to learn a policy that assigns a continuous treatment α to users and generates more value in expectation than the reference under the cost constraints.

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

[4] The dataset is available on Hugging Face platform under the title `CUVET-policy`.

Results Results for $\Delta v_{\mathcal{B}}$ and $\Delta c_{\mathcal{B}}$ using all evaluation methods are provided in Table 2 (graphical representation is deferred to Appendix J). As in the MT-LIFT case, CUVET consistently outperforms baselines across all evaluation methods. On mCUV eq. evaluation, both CUVET versions achieve the highest total value, with a **+1% gain** compared to the reference – a significant improvement in this domain – while only slightly violating the cost constraint ($\Delta c_{\mathcal{B}} = 0$). This demonstrates the general effectiveness of the CUVET algorithm and highlights the efficiency of clipping in particular. Even under ILP evaluation, both CUVET versions yield superior results, with the unclipped version appearing more risk-averse than the clipped one. In addition, analysis of the performance depending on `max_depth` is provided in Appendix J.

Table 2: CUVET-policy data: relative uplifts in v and c on test split. Highest $\Delta v_{\mathcal{B}}$ result reported in bold.

Evaluation	Metric	CF-Het	CF-MSE	CUVET$_{cl}$	CUVET$_{uncl}$
mCUV eq.	$\Delta v_{\mathcal{B}}$	0.0042 ± 0.0029	0.0055 ± 0.0036	$\mathbf{0.0103 \pm 0.0033}$	0.0098 ± 0.0036
	$\Delta c_{\mathcal{B}}$	0.0015 ± 0.0016	-0.0017 ± 0.0018	0.0019 ± 0.0013	0.0010 ± 0.0010
ILP	$\Delta v_{\mathcal{B}}$	0.0105 ± 0.0100	0.0082 ± 0.0131	$\mathbf{0.0149 \pm 0.0100}$	0.0118 ± 0.0060
	$\Delta c_{\mathcal{B}}$	0.0026 ± 0.0032	-0.0005 ± 0.0057	0.0045 ± 0.0038	-0.0013 ± 0.0021
LP	$\Delta v_{\mathcal{B}}$	0.0111 ± 0.0098	0.0053 ± 0.0102	0.0094 ± 0.0096	$\mathbf{0.0155 \pm 0.0109}$
	$\Delta c_{\mathcal{B}}$	0.0036 ± 0.0035	0.0002 ± 0.0036	0.0035 ± 0.0043	0.0023 ± 0.0029

4 Conclusion

This paper introduces CUVET (Cost per Unit Value Equalization Tree), a novel policy learning approach which partitions the user space based on marginal cost per unit value equalization while respecting the cost constraint. CUVET offers an interpretable, structured framework for efficient treatment assignment, integrating domain knowledge into a closed-form solution, particularly beneficial for large-scale digital advertising and decision-focused learning. The CUVET method is based on the power-law form for the value-cost relationship, which may not always hold in real-world use-cases but is realistic in practice: indeed, our algorithm outperforms baselines on real-world data for which we do not have prior guarantees that the power-law assumption is satisfied, possible extensions and future work are reported to Appendix D. We also release the CUVET-policy benchmark, a new dataset generated from real-world A/B tests, to contribute to the advance of open and reproducible research in decision-focused learning.

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. *ACM Web Conference*.

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.

Battocchi, K., Dillon, E., Hei, M., Lewis, G., , 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. In *AAAI Conference on Artificial Intelligence*.

Bompaire, M., Désir, A., and Heymann, B. (2021). Robust label attribution for real-time bidding. *arXiv preprint arXiv:2012.01767*.

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

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

Diemert, E., Betlei, A., Renaudin, C., Amini, M.-R., Gregoir, T., and Rahier, T. (2021). A large scale benchmark for individual treatment effect prediction and uplift modeling. *arXiv preprint arXiv:2111.10106*.

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

Du, R., Zhong, Y., Nair, H. S., Cui, B., and Shou, R. (2019a). Causally driven incremental multi touch attribution using a recurrent neural network. *AdKDD Workshop*.

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

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

Fernández-Loría, C., Provost, F., Anderton, J., Carterette, B., and Chandar, P. (2022). A comparison of methods for treatment assignment with an application to playlist generation. *Information Systems Research*.

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

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

Hulatt, L. (2023). Marginal returns: Diminishing & increasing.

Ji, W. and Wang, X. (2017). Additional multi-touch attribution for online advertising. In *Conference on Artificial Intelligence*.

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

Kallus, N. (2018). Balanced policy evaluation and learning. *Advances in Neural Information Processing Systems*.

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*.

Kitagawa, T. and Tetenov, A. (2018). Who should be treated? Empirical welfare maximization methods for treatment choice. *Econometrica*, 86(2):591–616.

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*.

Kohavi, R., Deng, A., Longbotham, R., and Xu, Y. (2014). Seven rules of thumb for web site experimenters. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Lee, H.-S., Zhang, Y., Zame, W., Shen, C., Lee, J.-W., and van der Schaar, M. (2020). Robust recursive partitioning for heterogeneous treatment effects with uncertainty quantification. *Advances in Neural Information Processing Systems*, 33:2282–2292.

Lee, S., Liu, R., Song, W., Li, L., and Zhang, P. (2024). Subgroupte: Advancing treatment effect estimation with subgroup identification. *arXiv preprint arXiv:2401.12369*.

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*.

Mongeon, P., Brodeur, C., Beaudry, C., and Larivière, V. (2016). Concentration of research funding leads to decreasing marginal returns. *CoRR*, abs/1602.07396.

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

Ren, K., Fang, Y., Zhang, W., Liu, S., Li, J., Zhang, Y., Yu, Y., and Wang, J. (2018). Learning multi-touch conversion attribution with dual-attention mechanisms for online advertising. In *International Conference on Information and Knowledge Management*.

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

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.

Simchi-Levi, D. and Wang, C. (2023). Pricing experimental design: causal effect, expected revenue and tail risk. In *International Conference on Machine Learning*.

Swaminathan, A. and Joachims, T. (2015). Counterfactual risk minimization: Learning from logged bandit feedback. In *International Conference on Machine Learning*.

Tang, D., Agarwal, A., O'Brien, D., and Meyer, M. (2010). Overlapping experiment infrastructure: More, better, faster experimentation. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

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*.

Vladimirova, M., Pavone, F., and Diemert, E. (2024). FairJob: A real-world dataset for fairness in online systems. In *Advances in Neural Information Processing Systems Datasets and Benchmarks Track*.

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.

Wahdany, D., Schmitt, C., and Cremer, J. L. (2023). More than accuracy: end-to-end wind power forecasting that optimises the energy system. *Electric Power Systems Research*, 221:109384.

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*.

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.

Zhu, H., Murali, P., Phan, D., Nguyen, L., and Kalagnanam, J. (2020). A scalable mip-based method for learning optimal multivariate decision trees. *Advances in neural information processing systems*, 33:1771–1781.

A Related works

In many decision-making scenarios, individuals exhibit significant heterogeneity in their responses to different treatments or interventions. Effectively leveraging this heterogeneity requires learning a personalized decision-making rule–called a policy–that maps a user's individual characteristics to the treatment to be assigned to that user. This paper focuses on the problem of learning treatment assignment policies from (offline) experimental data, a setting that has gained prominence with the increasing availability of large-scale datasets across diverse domains such as healthcare, incentive recommendation, and digital advertising (Huang et al., 2024; Vladimirova et al., 2024; Diemert et al., 2018, 2021). The data usually contain features representing individual characteristics, actions corresponding to treatment assignments and observed outcomes resulting from those policies.

A/B tests (randomized controlled experiments) are widely used to evaluate the effectiveness of different interventions, and constitute a particularly relevant source of exploitable offline data (Kohavi et al., 2013, 2014; Tang et al., 2010; Xu et al., 2015). In an A/B test setting, individuals are randomly assigned to different treatment groups, ensuring that the collected data provides an unbiased estimate of treatment effects. However, traditional A/B testing often focuses on estimating the average treatment effect rather than learning personalized policies that account for individual heterogeneity (Tu et al., 2021). By leveraging policy learning techniques, we aim to go beyond average treatment effect estimation, in order to identify individualized treatment rules that maximize outcomes at the individual level.

Estimating the causal effect of a specific treatment assignment on our outcome of interest (*i.e.* cost and value) is made difficult due to the fundamental problem of causal inference. In addition, in some applications, the experimental data contains additional complexities. For instance, online advertising suffers from imperfect attributions: there usually are several ads displayed in the few hours preceding each conversion, making it not straightforward to choose which action or ad placement should receive the reward for a subsequent conversion. One could assign rewards to several previous ad displays following a multi-touch attribution rule (Du et al., 2019a; Ji and Wang, 2017; Ren et al., 2018; Bompaire et al., 2021), though it does not entirely solve the label imperfection issue. These factors lead to typically very noisy experimental data, which contribute to making individual causal effect estimation an even harder problem.

A way to simplify the problem is to group similar users together into *buckets* (also called *cohorts*) and study aggregated causal effects, which are less noisy and easier to estimate than individual ones. This approach involves heterogeneous treatment effect (HTE) estimation or one of its sub-disciplines, namely subgroup discovery and subgroup analysis, to find an effective space partitioning (Athey and Imbens, 2016; Wager and Athey, 2018; Tu et al., 2021; Lee et al., 2020, 2024). The main drawback of these methods is a lack of scalability. Another more pragmatic approach is to opt for scalable, quantile-based partitionings (Betlei et al., 2024), which are less precise in regrouping users with similar causal effects. A reasonable partitioning can also be proposed by a domain expert or by taking into account the economic nature of the data.

Once both the user space and the treatment effects are estimated, there only remains a standard knapsack-like optimization problem in order to optimally assign policies to each bucket of users. Such problems are known to be NP-hard and solving them requires approximation algorithms that are often not suited for large-scale data (Tu et al., 2021; Betlei et al., 2024; Zhou et al., 2023a; Albert and Goldenberg, 2022).

Diminishing returns assumption. The diminishing returns assumption is widely used across economics, business, engineering, and psychology, particularly in knapsack-like allocation problems.

It states that as input increases, the marginal gain from each additional unit of input decreases (Gabaix, 2009; Shephard and Färe, 1974; Hulatt, 2023; Mongeon et al., 2016). Many real-world phenomena exhibit diminishing returns, such as efficiency improvements in hardware systems (e.g., car engines, batteries), where early optimizations provide large gains, but subsequent improvements become increasingly difficult. In machine learning, this principle underlies scaling laws, where increasing compute or amount of data yields diminishing performance improvements (Kaplan et al., 2020). Similarly, in marketing, increasing advertising budgets does not lead to proportional increases in sales due to market saturation (Zhou et al., 2023a; Zhao et al., 2019). In assignment problems, returns are often modeled as following a power-law relationships, where a small number of items contribute most of the value (Clauset et al., 2009). For example, in digital advertising, initial spending brings high returns, but additional budget allocation saturates quickly, making cost-aware allocation crucial (Simchi-Levi and Wang, 2023). Understanding these diminishing returns is essential for optimizing treatment assignment policies and improving resource allocation in various applications.

Two-stage learning. From a causal perspective, the allocation problems have two different issues to tackle: (i) treatment effect estimation, which predicts the expected benefit of each intervention, and (ii) optimization, often referred to as the treatment assignment problem. A common approach is to treat each step (treatment effect estimation and assignment) separately (Ai et al., 2022; Albert and Goldenberg, 2022; Tu et al., 2021; Zhao et al., 2019; Yan et al., 2023; Wu et al., 2022). However, proceeding in two steps has significant drawbacks: it incorporates cost considerations only in the second stage, leading to decision errors if the first-stage estimates are high-variance or biased (Fernández-Loría et al., 2022). In particular, errors in treatment effect estimation propagate into the optimization step, resulting in suboptimal allocations. To address this, the decision-focused approach (Mandi et al., 2023) integrates cost-aware learning in the first stage, ensuring that predictions are directly optimized for downstream decision-making tasks, reducing the impact of estimation errors.

Decision-focused learning. Instead of separating treatment effect estimation and allocation, decision-focused learning integrates them into a single optimization problem, aligning predictions with decision outcomes (Mandi et al., 2023; Elmachtoub and Grigas, 2022). One challenge in decision-focused learning is computational complexity, *i.e.* solving the end-to-end optimization problem often requires to reformulate the problem into differentiable optimization layers, for instance, focusing on ranking problem as an end goal (Du et al., 2019b; Zhou et al., 2023a). Other formulations cast treatment assignment as a stochastic optimization problem (Tu et al., 2021), assuming normally distributed outcomes of bucket-level objective and constraints, with the final problem remains in the knapsack form. However, the bucketization is based on the causal tree approach (Athey and Imbens, 2016) which is not fully incorporated into the decision-focused framework. Similarly, Betlei et al. (2024) reformulate the optimization criteria, focusing on maximizing the probability of success, such as the likelihood of a positive A/B test outcome. However, their method does not explicitly incorporate user-space bucketization in the optimization phase. Addressing these limitations remains an active area of research.

Policy learning. From a policy learning perspective, the goal is to learn an optimal treatment assignment policy that directly maps individual characteristics to the best treatment, rather than relying on a separate treatment effect estimation step (Zhou et al., 2023b). Advances in offline policy learning allow for direct learning of policies from observational and experimental data, effectively bypassing the challenges of high-variance treatment effect estimation (Swaminathan and Joachims, 2015; Kitagawa and Tetenov, 2018; Kallus, 2018). Most of the state-of-the-art methods that take into account partitioning are built on the idea of optimal trees (Bertsimas and Dunn, 2017; Jo et al., 2021; Amram et al., 2022; Zhou et al., 2023b). Compared to traditional two-stage approaches, policy learning can be more robust to estimation errors and better suited for large-scale decision-making problems where individual-level heterogeneity plays a crucial role. Despite these advantages, policy learning can be computationally demanding (Jo et al., 2021; Zhou et al., 2023b), particularly in settings with high-dimensional feature spaces or limited interventional data.

Our work empirically compares our algorithm to several baseline approaches with a similar focus, categorized into three main groups: HTE estimation methods (e.g., Causal Tree (Athey and Imbens, 2016), Causal Forest (Athey et al., 2019)), treatment assignment approaches via HTE estimation (Ai et al., 2022; Tu et al., 2021; Wu et al., 2022), and direct treatment assignment methods (Jo et al., 2021; Amram et al., 2022; Zhou et al., 2023a). These baselines, while effective, have limitations in

Table 3: Comparison of different methods and their characteristics. In the "Problem" column, "HTE" stands for "HTE estimation", "HTE for TA" is for "HTE estimation for Treatment Assignment" and "TA" is for "Treatment Assignment".

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)	HTE	Multiple	✓	✗	✗	✓
Merging trees (Tu et al., 2021)	HTE for TA	Multiple	✓	✗	✗	✗
Large-Scale Budget-Constrained CF (Ai et al., 2022)	HTE for TA	Multiple	✓	✓	✗	✗
Distill-HTE (Wu et al., 2022)	HTE for TA	Multiple	✓	✗	✗	✗
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	✗	✓	✓	✓
CUVET (ours)	TA	Continuous or Discretized	✓	✓	✓	✓

flexibility, scalability, or computational efficiency, which we discuss in detail in Section B and which our algorithm aims to address.

B Baseline approaches

We discuss possible connection of the described problem with two-stage, decision-focused and policy learning in Appendix A. Here we focus on baseline approaches relevant to our problem, where our main goal is to find a relevant user space bucketization. A detailed comparison of the methods is shown in Table 3.

Most of the approaches can be divided into three main categories depending on the exact problem they are designed to solve in the first place, even if they can all be used to partition the user space (which is what our work focuses on).

HTE estimation methods The Causal Tree (CT) Athey and Imbens (2016) algorithm uses a recursive partitioning approach to identify the buckets which share similar heterogeneous effects of a given binary treatment. Its splitting criteria is a modified version of the mean squared error (MSE) while penalizing higher estimation variances. Since CT is only applicable to binary treatments, it is much less flexible than our approach but can still be used as a baseline in those cases. Causal Forest (CF) (Athey et al., 2019) extend CT to ensemble models and solves a local moment equation problem – allowing CF trees to handle multiple treatment.

HTE estimation methods are used only as the first step (*partitioning*) of a two-stage approach to treatment assignment, requiring downstream methods to solve the second step (*assignment*).

Treatment assignment approaches via HTE estimation Several methods improve on CT and CF to optimize on the downstream, treatment assignment task. Ai et al. (2022) modify the splitting criteria of CF that allows similar users from multiple treatment groups to reside in the same node. Tu et al. (2021) learn one causal tree for each (treatment, outcome) pair and merge resulted buckets into a single tree. In Distill-HTE (Wu et al., 2022), distillation techniques are used to learn one multi-task decision tree from a black-box HTE model – in particular a Gradient Boosting Decision Tree (GBDT) model may be learned each (treatment, outcome) pair.

Although latter methods are claimed to be scalable, training and maintaining $|\mathcal{T}| \cdot |\mathcal{Y}|$ models is computationally intensive.

Direct treatment assignment methods Methods of this category are mostly built on the idea of optimal trees (Bertsimas and Dunn, 2017). Zhu et al. (2020) develop scalable mixed-integer programming tree method for training multivariate decision trees, using a 1-norm SVM to maximize the number of correctly classified instances and to maximize the margin between clusters at the leaf nodes. Jo et al. (2021) propose a method for learning optimal prescriptive trees using mixed-integer optimization. Under mild conditions their method converges to an optimal out-of-sample treatment assignment policy, as the data size tends to infinity. Amram et al. (2022) extend the exact optimal trees using coordinate ascent to the problem of learning prescription policies based on complete counterfactual information. Resulting Optimal Policy Trees are interpretable and scalable, handling

both discrete and continuous treatments. (Zhou et al., 2023b) study a family of doubly robust algorithms for multi-action policy learning. They develop a customized tree-search based algorithm that finds the exact optimal tree in the policy optimization step.

Unfortunately, most of direct treatment assignment approaches are NP-hard (Jo et al., 2021), insufficiently scalable (Zhu et al. (2020)'s handles up to 245,000 samples; Zhou et al. (2023b)'s has complexity of $\mathcal{O}(|\mathcal{X}|^2)$, solving an exact tree search problem in 2.5 hours on a small dataset with $|\mathcal{X}| = 10^5, d = 12$ and $|\mathcal{T}| = 3$), or use proprietary software (Amram et al., 2022).

C 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 the reference treatment is a placebo). We denote $\mathbf{Y} = (Y^c, Y^v) \in \mathbb{R}^2$ the random *outcome vector* we observe after treatment assignation 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.

C.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:

$$\max_{\psi:\mathcal{X}\to\mathcal{T}} \mathbb{E}[Y^v(\psi(X))] \quad s.t. \quad \mathbb{E}[Y^c(\psi(X))] \leq C, \tag{3}$$

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 multi-treatment 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 (3), 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 (3) can be rewritten as

$$\max_{\phi:\mathcal{B}\to\mathcal{T}} \sum_{b\in\mathcal{B}} Y_b^v(\phi(b)) \quad s.t. \sum_{b\in\mathcal{B}} Y_b^c(\phi(b)) \leq C, \tag{4}$$

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 *bucket level* expected outcome vector.

C.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

$$\mathcal{T} = \{t_0(\alpha),\ \alpha \in A\}, \tag{5}$$

where $A \in \mathbb{R}+$ is an interval of the form $[\alpha_{\min}, \alpha_{\max}]$, 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 in 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.* $\mathbf{Y}(\alpha) = \mathbf{Y}(t(\alpha))$. Moreover, we will 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 $v_b := Y_b^v$.

Elasticity assumption It is quite common to model the behavior of quantitative metrics using elastic relationships, which describe the sensitivity of these variables to a change in a cause variable (Phillips, 2021; Zhao et al., 2019; Zhou et al., 2023a). This assumption is commonly used in real-world applications: for example, advertising spend typically shows an elastic return curve (Yan et al., 2023; Zhou et al., 2023a).

We choose to model the relationships between the treatment parameter α, cost y^c and value y^v as power-law (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:

$$v_b(\alpha) = v_b(1)\alpha^{\gamma_b}, \quad c_b(\alpha) = c_b(1)\alpha^{\eta_b}. \tag{6}$$

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 that $\gamma_b < \eta_b$, 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 spend in bucket b, the less spending an additional marginal amount has an effect on the value.

D Limitations and future work

The power-law form for the value/cost relationship may not hold universally (as we describe in the next paragraph), however, it is a natural first order assumption for concave functions (equivalent to a local linear approximation for the log of the quantities).

Power-law assumption exception cases While the power-law elasticity assumption is very common, its applicability depends on the system dynamics. Below we provide several examples with illustration from digital advertising when the power-law elasticity assumption is not realistic:

- If all competitors increase spending simultaneously or if most of the audience is already reached, spending more leads to diminishing returns but not necessarily power-law. For example, a brand is already dominating a browser search. The returns in these cases often flatten out, but remain locally elastic
- If users are driven entirely by price, extra ad spend will not yield nonlinear gains. For example, generic campaigns for undifferentiated products like USB cables.

CUVET future work Despite the strengths of CUVET demonstrated empirically, there exist several interesting extensions:

- *Temporal drift of elasticities*: We assume elasticity parameters are fixed through time, but this typically does not hold in the real-world. Future work could incorporate uncertainty-aware methods like conformalized quantile regression.
- *Scalability*: Efficient pruning techniques could improve stability and prevent over-fitting.
- *Multi-objective optimization*: Extending CUVET to handle trade-offs (e.g. revenue vs. engagement) and constraints (e.g. fairness, budget caps).
- *Observational data*: Adapting CUVET for non-randomized data using covariate adjustment and causal inference methods would be a natural extension.

Addressing these challenges will enhance CUVET's scalability, robustness, and applicability to complex decision-making, inspiring further methods that leverage domain knowledge in large-scale settings.

E Proofs

E.1 Optimal policy expression proposition

E.1.1 Optimal splitting criterion

We formulate the optimal policy in the following proposition, the proof of which is referred to Appendix E.1.2

Proposition 1. *For a given partition $\mathcal{B}$, a treatment set $\mathcal{T}$ parametrized by a continuous $\alpha \in A$ as in* (5) *and assuming power-law relationships as in* (6) *hold in every bucket from $\mathcal{B}$, the optimal treatment parameter α for each bucket $b \in \mathcal{B}$ is expressed in the closed-form:*

$$\alpha_b^* = \left(\frac{\mu}{mCUV_b(1)} \right)^{\frac{1}{\eta_b - \gamma_b}}, \tag{7}$$

where μ is the unique solution to (2)*, $\forall \alpha \in A, \quad mCUV_b(\alpha) = {}^{\partial c_b}/_{\partial v_b}(\alpha)$ and γ_b, η_b are the elasticity parameters from* (6)*.*

Now, computing the value of μ from (2), then injecting the resulting values of α_b^* given by (7) in our optimization objective (2) yields the following closed-form criterion:

$$\sum_{b \in \mathcal{B}} v_b(\alpha_b^*), \tag{8}$$

which can be readily estimated using offline data estimates of the elasticity parameters for cost and value. The core-idea of our CUVET 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.

E.1.2 Optimal splitting criterion proof

As a reminder, for $\alpha \in A$

$$c_b(\alpha) = c_b(1) \times \alpha^{\eta_b}, \tag{9}$$
$$v_b(\alpha) = v_b(1) \times \alpha^{\gamma_b}. \tag{10}$$

Straightforward derivations yield that for any possible value c_b of the cost in b, v_b can be expressed as a function of c_b as follows:

$$v_b(c_b) = v_b(c_b(1)) \times \left(\frac{c_b}{c_b(1)}\right)^{\gamma_b/\eta_b}.$$

We call $\epsilon_b = \gamma_b/\eta_b$ the elasticity of v_b (value in b) relative to c_b (cost in b). We assume that $\epsilon_b < 1$ since the cost-value relationship is classically assumed to be concave in every bucket (decreasing marginal gains). This in turn implies that $\gamma_b < \eta_b$ for all $b \in \mathcal{B}$.

Cost of Unit Value (CUV). For $b \in \mathcal{B}$ and $\alpha \in A$ we define the cost of unit value in g for bid multiplier α as:

$$CUV_b(\alpha) = \frac{c_b(\alpha)}{v_b(\alpha)}, \tag{11}$$

which can be explicitly written using (9) and (10) as:

$$CUV_b(\alpha) = CUV_b(1)\alpha^{\eta_b - \gamma_b}. \tag{12}$$

This quantity corresponds to the *average cost of value* in the bucket b. In particular, for every $b \in \mathcal{B}$, $CUV_b(1) = \frac{c_b(1)}{v_b(1)}$ and $mCUV_b(1) = \frac{CUV_b(1)}{\epsilon_b}$ are computable from offline data directly, without the need for counterfactual estimation methods.

Marginal Cost of Unit Value (mCUV). For $b \in \mathcal{B}$ and $\alpha \in A$ we define the marginal cost of unit value in b for bid multiplier α as:

$$mCUV_b(\alpha) = \frac{\partial c_b(\alpha)}{\partial v_b(\alpha)},$$

which can be explicitly written using (9) and (10) as:

$$mCUV_b(\alpha) = \left(\frac{CUV_b(1)}{\epsilon_b}\right)\alpha^{\eta_b - \gamma_b}. \tag{13}$$

This quantity corresponds to the cost of the *next unit of value*: it is the current cost of value in the bucket b.

Equalization. For all $b, b' \in \mathcal{B}$ let μ be

$$mCUV_b(\alpha_b) = mCUV_{b'}(\alpha_{b'}) := \mu. \tag{14}$$

and with μ satisfying[5]

$$\sum_b c_b(1) \times \left(\frac{\mu}{mCUV_b(1)}\right)^{1/(1-\epsilon_b)} = \sum_b c_b(1) = C. \tag{15}$$

For a given partition $\mathcal{B}$, the optimal treatment for each bucket can be expressed in a closed-form expression. From $mCUV_b(\alpha_b^*) = \mu$ we get

$$\left(\frac{CUV_b(1)}{\epsilon_b}\right)(\alpha^*)^{\eta_b - \gamma_b} = \mu.$$

Indeed, the optimal value of the treatment parameter α in each bucket, α_b^*, can be written as:

$$\alpha_b^*(\mu) = \left(\frac{\mu \cdot \epsilon_b}{CUV_b(1)}\right)^{1/(\eta_b - \gamma_b)} = \left(\frac{\mu}{mCUV_b(1)}\right)^{1/(\eta_b - \gamma_b)}, \tag{16}$$

taking into account the relationship between $mCUV_b(1)$ and $CUV_b(1)$.

[5]Equation (15) has a unique solution since the expression on the right-hand side is strictly increasing in μ and has value 0 for $\mu = 0$ and goes to $+\infty$ when μ goes to $+\infty$.

E.2 Equivalence to Lagrange multipliers method

Proposition 2. *Consider a given partition $\mathcal{B}$, a treatment set $\mathcal{T}$ parametrized by a continuous $\alpha \in A$ as in* (5) *and assume power-law relationships as in* (6) *hold in every bucket from $\mathcal{B}$. Then the greedy mCUV-equalization strategy consisting in equalizing marginal cost per unit value across user buckets yields an optimal policy, i.e. a policy that maximizes total expected value under a fixed (iso-cost) budget.*

We prove Proposition 2, *i.e.* that equalization of mCUV in buckets yields the same parameters α that are found using constrained optimization.

Firstly, we formulate the constrained optimization problem for K buckets. We are seeking to maximize the total expected value (sum of the values in each buckets), constrained by the total cost not being greater than our fixed budget C. This writes:

$$\max_{\alpha_1,\dots,\alpha_K} \sum_{b=1}^{K} V_b(\alpha_b) \quad \text{s.t.} \sum_{b=1}^{K} C_b(\alpha_b) = C.$$

This can be reformulated using the Lagrangian function $\mathcal{L}$ as follows:

$$\begin{aligned}\mathcal{L}(\alpha_1, ..., \alpha_K, \lambda) &= \sum_{b=1}^{K} V_b(\alpha_b) - \lambda \left(C - \sum_{b=1}^{K} C_b(\alpha_b) \right) \\ &= \sum_{b=1}^{K} V_b(1) \cdot {\alpha_b}^{\gamma_b} - \lambda \left(C - \sum_{b=1}^{K} C_b(1) \cdot {\alpha_b}^{\eta_b} \right)\end{aligned}$$

Writing out the first-order conditions yields:

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial \alpha_b} = 0 \Rightarrow \frac{\partial V_b(\alpha_b)}{\partial \alpha_b} - \lambda \frac{\partial C_b(\alpha_b)}{\partial \alpha_b} &= 0 \\ \Rightarrow V_b(1) \cdot \gamma_b \cdot \alpha_b^{\gamma_b - 1} &= \lambda \cdot C_b(1) \cdot \eta_b \cdot \alpha_b^{\eta_b - 1} \\ \Rightarrow \lambda = \frac{V_b(1) \cdot \gamma_b \cdot \alpha_b^{\gamma_b - 1}}{C_b(1) \cdot \eta_b \cdot \alpha_b^{\eta_b - 1}} &= \frac{\alpha_b^{\gamma_b - \eta_b}}{mCUV_b(1)} = \frac{1}{mCUV_b(\alpha_b)} \\ \Rightarrow mCUV_b(\alpha_b) &= \frac{1}{\lambda} = \mu \\ \Rightarrow \alpha_b &= (\lambda \cdot mCUV_b(1))^{\frac{1}{\gamma_i - \eta_b}} \\ \frac{\partial \mathcal{L}}{\partial \lambda} = 0 \Rightarrow \sum_{b=1}^{K} C_b(\alpha_b) &= C.\end{aligned}$$

This exactly matches the values of α_b^* given in (7) and iso-cost constraint (2), and therefore proves that equalizing marginal cost of unit value is equivalent to maximizing value under cost constraint as stated.

E.3 Equivalence to KKT conditions

We consider the multi-bucket treatment assignment problem

$$\max_{\alpha_1,\dots,\alpha_K} \sum_{b=1}^{K} v_b(\alpha_b) \quad \text{s.t.} \quad \sum_{b=1}^{K} c_b(\alpha_b) \le C, \;\; \alpha_b \in [\alpha_{\min}, \alpha_{\max}].$$

This is a convex optimization problem under the concavity assumption $\gamma_b < \eta_b$. Its optimal solution can be characterized by the Karush–Kuhn–Tucker (KKT) conditions.

KKT conditions. There exists $\lambda \geq 0$ such that for all $b = 1, \ldots, K$:

$$\text{(Stationarity)} \quad \frac{\partial v_b(\alpha_b)}{\partial \alpha_b} - \lambda \frac{\partial c_b(\alpha_b)}{\partial \alpha_b} = 0, \tag{17}$$

$$\text{(Primal feasibility)} \quad \sum_{b=1}^{K} c_b(\alpha_b) \leq C, \tag{18}$$

$$\text{(Dual feasibility)} \quad \lambda \geq 0, \tag{19}$$

$$\text{(Complementary slackness)} \quad \lambda\Big(\sum_{b=1}^{K} c_b(\alpha_b) - C\Big) = 0. \tag{20}$$

Interpretation. Equation (17) can be rewritten as

$$mCUV_b(\alpha_b) = \frac{1}{\lambda} =: \mu, \quad \forall b,$$

i.e. the marginal cost per unit value is equalized across all buckets. Together with the cost constraint (18)–(20), this uniquely determines μ and hence the optimal assignments α_b^* as given in Proposition 1. Therefore, the equalization strategy used in CUVET is exactly the KKT characterization of the optimum.

F Synthetic example when CUVET outperforms Causal Forest

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 radomized 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 learn 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:

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

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:

$$\begin{array}{c} mCUV(b_1) = mCUV(b_2) \approx 0.91, \quad mCUV(b_3) \approx 0.11 \\ \mu \approx 0.1, \quad \alpha^*(b_1) = \alpha^*(b_2) \approx 1.2, \quad \alpha^*(b_3) \approx 0.63 \end{array}$$

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 that will make any HTE-based method consider threshold t_1 as more important since it better at splitting different heterogeneous treatment effects. However this HTE splitting is not consistent with the best treatment assignment split in this context, making our method conceptually outperform any HTE based method at the task of treatment assignment targeted partitioning.

Data We generate data of 300000 points with the defined parameters and create train/test splits randomly in $50/50$ proportion. For the outcome of CF we use $y = v - \frac{\eta}{\gamma} c$.

Results. Firstly we plot the CUVET criterion value along with the optimal value α^* with respect to the chosen threshold on Figure 1. Following our expectations, CF always chooses $t^* = t_1 = 0.1$, while CUVET chooses $t^* = t_2 = 0.5$.

Results for $\Delta v_{\mathcal{B}}$ and $\Delta c_{\mathcal{B}}$ using mCUV equalization evaluation are provided in Table 4. We observe that, on both train and test datasets, CUVET reaches much larger $\Delta v_{\mathcal{B}}$ than the baselines. At the same time, while on the train set $\Delta c_{\mathcal{B}} = 0$, we observe a small deviation of $\Delta c_{\mathcal{B}}$ from 0 on the test set, explained by the variance in elasticity parameters estimations.

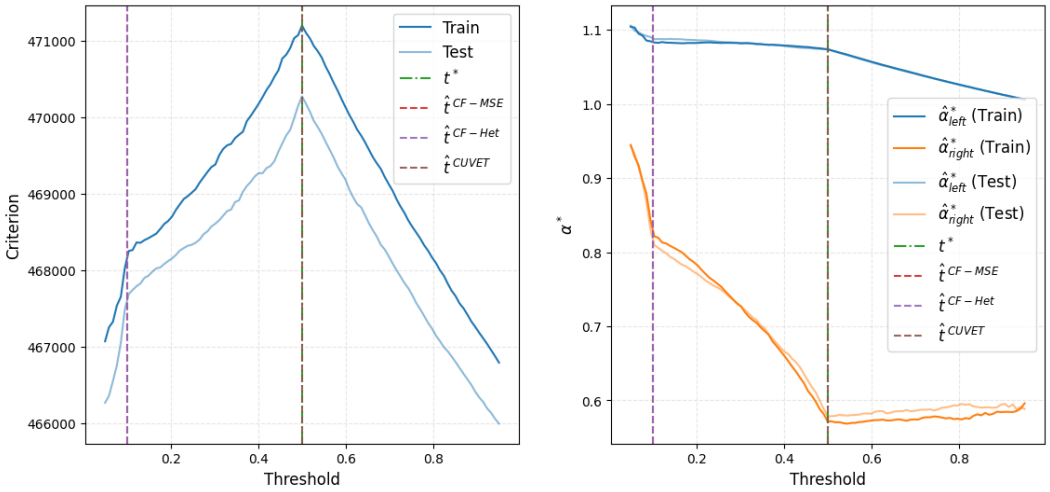

Figure 1: Synthetic example: criterion and $\alpha^*_{b_i}$ as functions of the threshold t.

Table 4: Synthetic example: relative uplifts $\Delta v_{\mathcal{B}}$ and $\Delta c_{\mathcal{B}}$ on train and test splits.

Split	Metric	CF-Het	CF-MSE	CUVET
$Train$	$\Delta v_{\mathcal{B}}$	0.0039	0.0039	0.0102
	$\Delta c_{\mathcal{B}}$	0.0	0.0	0.0
$Test$	$\Delta v_{\mathcal{B}}$	0.0038	0.0038	0.009
	$\Delta c_{\mathcal{B}}$	-0.001	-0.001	-0.0016

G Additional data description

G.1 MT-LIFT details

Preprocessing As treatments represent coupons, we can treat them as ordered and apply our methodology, allocating different coupons to different buckets. However, we introduce several modifications of the data. Firstly, we normalize treatment values to be $\{0.8, 0.9, 1, 1.1, 1.2\}$ and define $t_0 = 1$ to be our reference treatment. Similar to (Zhou et al., 2023a; Yan et al., 2023), we use the "click" label as a proxy of the cost and "conversion" as the value. Surprisingly, we observe the conversion elasticity with respect to treatment to be greater than the click one ($\gamma > \eta$).

In order to preserve $\gamma < \eta$, we scale the conversion variable by adding constants depending on treatment:

$$v'_{t_i} = exp\left(log\left(1 + v_{t_i}\right) + \mu_{t_i}\right) - 1$$

G.2 CUVET-policy privacy details

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. Further, we manually selected some continuous user features, added noise, and standardized them, making the re-identification impossible as the original scale is not provided. We further scaled and added noise to the value and cost features to keep business confidentiality.

H Reproducibility details

H.1 CUVET algorithm

When developing our algorithm we define multiple hyper-parameters that define `stopping_criterion`, see Algorithm 1:

- maximum depth `max_depth` as we do not know the true number of buckets,
- minimum number of samples per leaf `min_samples_leaf`. In all our experiments we will fix this parameter to $\frac{|\mathbf{X}|}{2^{\texttt{max\_depth}+2}}$,
- minimum number of samples to perform split `min_samples_split`. In all our experiments we will fix this parameter to $\frac{|\mathbf{X}|}{2^{\texttt{max\_depth}+1}}$

For clipping, we use min and max treatment values.

Estimating elasticity parameters The value and cost elasticity parameters are not known in advance and should be estimated for each region of interest. To do so, for each potential (feature, threshold) split, we estimate those parameters by performing a linear regression on the logarithm of the total outcome value and cost in both the 'left' and 'right' regions formed by the split. Note that this implies our dataset contains at least two different (randomly assigned) values of the treatment parameter α in each region: this typically is the case when using data extracted from a randomized controlled trial (or A/B test) with two or more treatments.

H.2 Baselines for experiments

In order to find an optimal treatment assignment $\phi^*_{\mathcal{B}}$, three different approaches were chosen:

1. **mCUV equalization**: optimal policy for each bucket is found in the closed-form expression (see Section 2.1), representing the case where we let optimal policies to be continuous, though inside the $[\alpha_{\min}, \alpha_{\max}]$ interval.
2. **Integer Linear Programming (ILP)**: optimization is with respect to decision variables δ_{kb} that are boolean indicators to assign treatment k to bucket b (this works when the treatment set is discrete).
3. **Linear Programming (LP)**: stochastic version of ILP - decision variables p_{kb} are probabilities to assign treatment k to bucket b.

As explained in Section B, defining appropriate baselines to compare with CUVET is not straightforward. We compare CUVET to single trees output by the CF algorithm, as CF handles multiple treatments and is suitable for large-scale datasets. As CF inputs only one outcome, we incorporate the cost constraint in the splitting criterion by defining a proxy outcome as linear combination of v and c. For the experiments, the CF algorithm from the EconML library (Battocchi et al., 2019) was used. We use two splitting criterion of the CF tree, namely mean squared error CF-MSE and heterogeneity score CF-Het (details are in Appendix H.5).

We also apply two versions of CUVET: in the *clipped* one (denoted in Seciton 3 as CUVET$_{cl}$) – the values of α^*_b are always clipped to $[\alpha_{min}, \alpha_{max}]$ during training, while in *unclipped* version (denoted in Seciton 3 as CUVET$_{uncl}$) we simply constrain $\alpha^*_b > 0$ but allow the values to leave the $[\alpha_{min}, \alpha_{max}]$ interval. Comparing output partitions from both approaches enables us to evaluate how much clipping α^* (and thus introducing a bias with respect to our theoretical result expressed in Proposition 1) is harmful for the final treatment assignment performance.

H.3 Example on building a tree with cost constraint

Cost constraint As we would like the possibility to have leaves at different levels (i.e to have an odd number of buckets) , we should ensure the conservation of the total cost C that we use 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 leafs $4 \leftarrow 2 \rightarrow 5$, $6 \leftarrow 3 \rightarrow 7$. 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: $C_2 = c_2(\alpha_2^*)$. From this example, we can conclude we need to apply as a cost constraint when growing the tree the optimal cost in the parent node. For the root node however, we would use the total cost of the samples if it were under the policy of reference (i.e $\alpha = 1$).

H.4 Evaluation

For the outcome of CF we use $y = v - \frac{\gamma}{\eta}c$ – in this form y can be treated as proxy of revenue.

For the model selection, we run all methods with `max_depth` $\in \{1, 2, 3\}$. On CUVET-policy data we used validation split to select the best model, though on MT-LIFT data we reported the best models on train split – as splitting it more would increase the noise in performance results.

In order to build confidence intervals for the results on real datasets, 10 bootstraps were generated for both metrics: it was done on test split for MT-LIFT data and on both validation and test splits for CUVET-policy data. For the error estimation, $1.96 * \sigma$ values were used.

H.5 Causal Forest splitting criteria

These criteria used in Battocchi et al. (2019) package that solve any linear moment problem of the form:

$$\mathbb{E}[J \cdot \theta(x) - A \mid X = x] = 0$$

The **"mse" criterion** finds splits that maximize the score:

$$\sum_{child} w_{child} \cdot \theta_{child}^T \cdot \mathbb{E}[J \mid X \in child] \cdot \theta_{child}$$

This coincides with minimizing the MSE:

$$\sum_{child} \mathbb{E}[(Y - \langle \theta_{child}, T \rangle)^2 \mid X = child] \cdot w_{child}$$

Internally, for the case of more than two treatments this criterion is approximated by computationally simpler variants for computational purposes. In particular, it is replaced by:

$$\sum_{child} w_{child} \cdot \rho_{child}^T \cdot \mathbb{E}[J \mid X \in child] \cdot \rho_{child}$$

where:

$$\rho_{child} := J_{parent}^{-1} \mathbb{E}[A - J \cdot \theta_{parent} \mid X \in child]$$

This can be thought of as a heterogeneity-inducing score, but putting more weight on scores with a large minimum eigenvalue of the child Jacobian $\mathbb{E}[J \mid X \in child]$, which leads to smaller variance of the estimate and stronger identification of the parameters.

The **"het" criterion** finds splits that maximize the pure parameter heterogeneity score:

$$\sum_{child} w_{child} \cdot \rho_{child}^T \cdot \rho_{child}$$

This can be thought of as an approximation to the ideal heterogeneity score:

$$\frac{w_{left} \cdot w_{right} \cdot \|\theta_{left} - \theta_{right}\|_2^2}{w_{parent}^2}$$

I MT-LIFT data

I.1 Graphical representation of Table 1

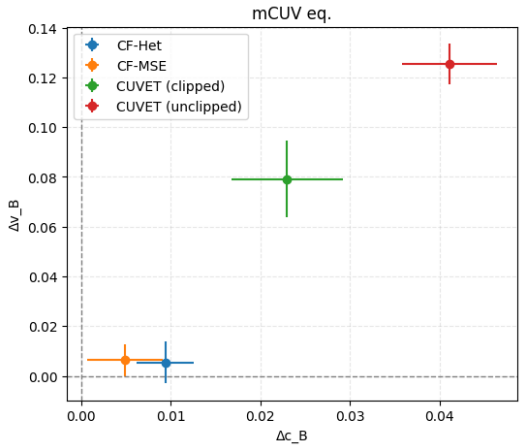

Figure 2: MT-LIFT data: relative uplifts in v and c on test split.

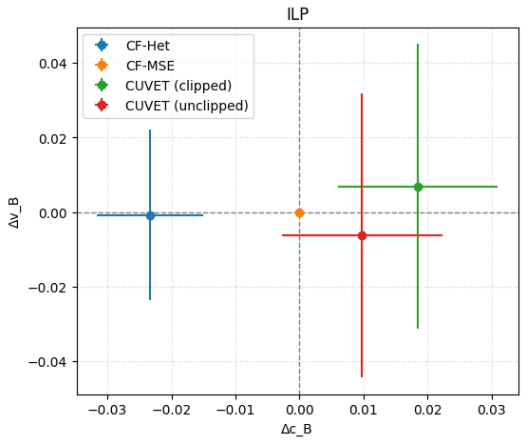

Figure 3: MT-LIFT data: relative uplifts in v and c on test split.

J CUVET-policy data

J.1 Graphical representation of Table 2

J.2 Performance as a function of `max_depth`

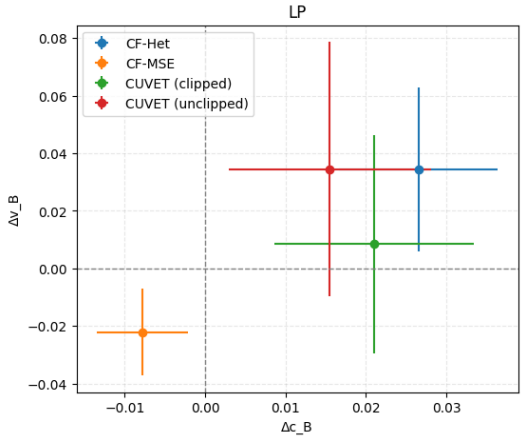

Figure 4: MT-LIFT data: relative uplifts in v and c on test split.

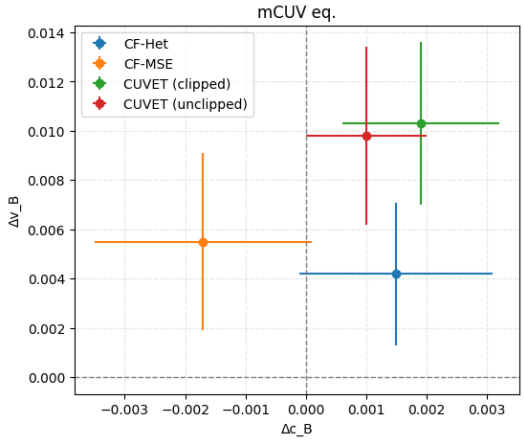

Figure 5: CUVET-policy data: relative uplifts in v and c on test split.

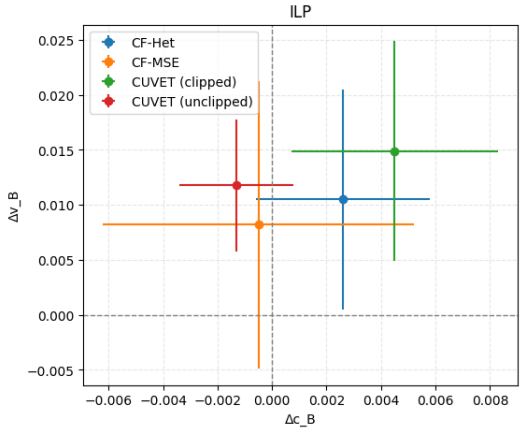

Figure 6: CUVET-policy data: relative uplifts in v and c on test split.

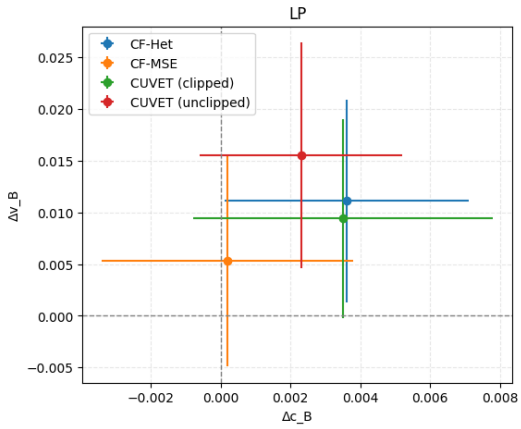

Figure 7: CUVET-policy data: relative uplifts in v and c on test split.

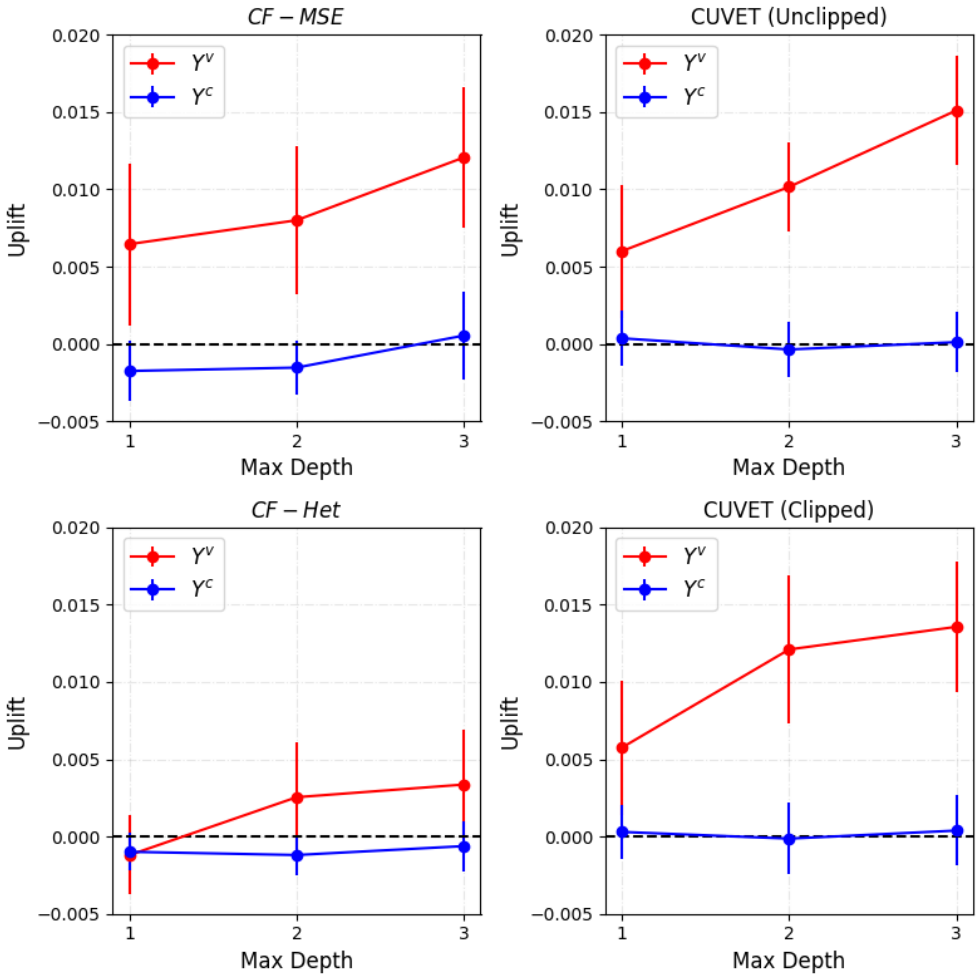

Figure 8: CUVET-policy data: relative uplifts in v and c with respect to max depth of the tree using $mCUV - Eq$ evaluation approach, on validation split.

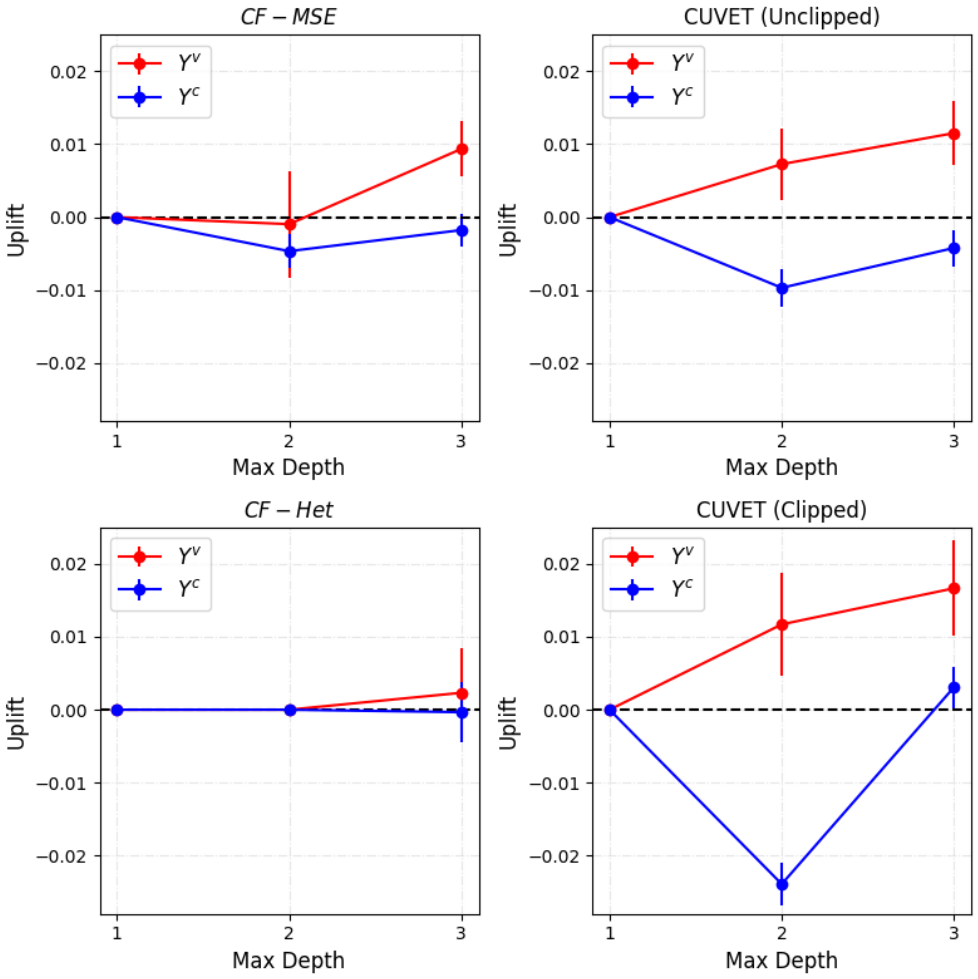

Figure 9: CUVET-policy data: relative uplifts in v and c with respect to max depth of the tree using ILP evaluation approach, on validation split.

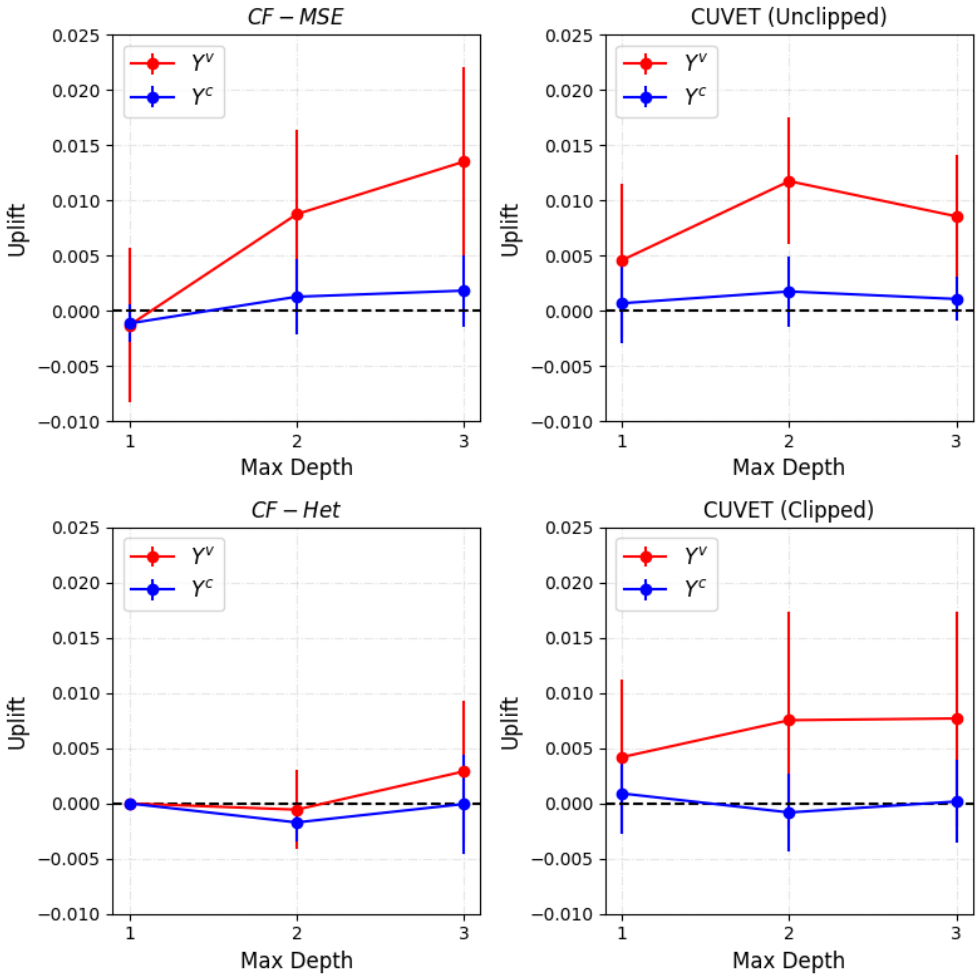

Figure 10: CUVET-policy data: relative uplifts in v and c with respect to max depth of the tree using LP evaluation approach, on validation split.