Optimal Subgroup Discovery at Every Support Threshold

Lincen Yang[1*] **Qi Huang**[1*] **Niki van Stein**[1]

Thomas Bäck[1] **Matthijs van Leeuwen**[1] **Zhong Li**[2,1†]

[1]LIACS, Leiden University, The Netherlands
[2]Great Bay University, China
{l.yang, q.huang, n.van.stein, t.h.w.baeck,
m.van.leeuwen, z.li}@liacs.leidenuniv.nl

Abstract

Subgroup discovery aims to identify regions of the feature space where a target variable deviates from its marginal distribution. A fundamental challenge in this problem is the trade-off between a subgroup's *deviation* and its *support*: small subgroups can be highly deviating but statistically meaningless, while large ones cannot deviate much by construction. Existing methods tackle this challenge by collapsing the two into a single scalar quality measure, implicitly committing to one particular trade-off and making it difficult to recover the most deviating subgroup under a user-specified support constraint. We instead provide the first theoretical and structural characterization of optimal subgroups with maximum Kullback-Leibler (KL) divergence under arbitrary support constraints. Under a partition-based generative model, where the feature space decomposes into regions of homogeneous conditional distributions of the target variable called *atoms*, we prove that every point on the deviation-support Pareto front is a union of whole atoms together with at most one fractional atom. This reduces an uncountable search to a finite combinatorial problem. Guided by this theory, we propose a simple two-step recipe: learn a homogeneous partition, then select atoms greedily. Across 19 benchmark datasets, it finds the most deviating subgroups at a range of support thresholds, and reliably improves subgroups returned by existing methods when applied as a post-processing step. These results empirically validate the sufficiency and necessity of our theoretical contributions.

1 Introduction

Subgroup discovery is the task of finding subsets of the feature space such that some target variable deviates from its marginal distribution. It has been successfully applied across various domains, and recent examples include knowledge discovery in heterogeneous catalysts in chemistry [13] and recommendation systems [18]. As the goal is often to obtain insights from data, subgroups are typically described by rules or unions of rules, where each rule is a conjunction of literals [15]. For example, in a dataset of housing prices with features such as location, size, and age, a subgroup might be "(location $=$ coastal) $\wedge$ (size $\geq 200\text{m}^2$)", which isolates a region of the feature space in which the conditional distribution of price differs markedly from the overall distribution.

*Equal contribution.
†Corresponding author.

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

Formally, given a feature vector $X \in \mathcal{X}$ and target $Y \in \mathcal{Y}$ distributed according to some joint distribution P, a subgroup is any measurable subset $Q \subseteq \mathcal{X}$, and its quality is measured by how much the conditional distribution $P(Y \mid X \in Q)$ deviates from the marginal $P(Y)$, typically via a divergence such as the Kullback-Leibler (KL) divergence, and by how large the subgroup is, measured by its support $P(X \in Q)$.

The support-deviation trade-off. A fundamental challenge in subgroup discovery is that deviation and support pull in opposite directions: very small subgroups can be arbitrarily deviating but statistically meaningless, while large ones cannot deviate much by construction. Practitioners need subgroups that are both deviating enough to be informative and large enough to be actionable.

Existing work conflates this trade-off into a single score. The dominant approach to handling this challenge is to fold support and deviation into a single scalar quality measure. Classic measures include weighted relative accuracy, weighted mean differences, and support-weighted KL divergence [22, 21, 31]. Recent state-of-the-art methods retain this design: RSD [29] penalizes small subgroups via an MDL description-length term, and SyFlow [34] maximizes a support-weighted KL objective through differentiable neural optimization. While this single-score strategy enables unified optimization, it implicitly commits to one particular support–deviation trade-off, and different weighting choices yield different subgroups.

More fundamentally, support-weighting does not actually resolve the underlying problem. Weighted scores such as the weighted-KL-divergence discourage small subgroups by penalizing coverage, on the implicit assumption that larger subgroups yield more reliable KL estimates. However, multiplying an unreliable estimate by its support merely rescales the noise, and because the weight asymmetrically rewards high-support subgroups, upward noise in a high-support subgroup's KL estimate can translate into a ranking advantage over genuinely deviating low-support ones. Figure 1 illustrates this: with two subgroups drawn from a Gaussian-mixture background—one (Subgroup A) genuinely deviating at support 0.08, the other (Subgroup B) a pure sample from the background at support 0.15—the support-weighted KL ranks the spurious Subgroup B above the genuine Subgroup A more than 20% of the time at sample size $k = 30$. Full reproducibility details are in Appendix C.1.

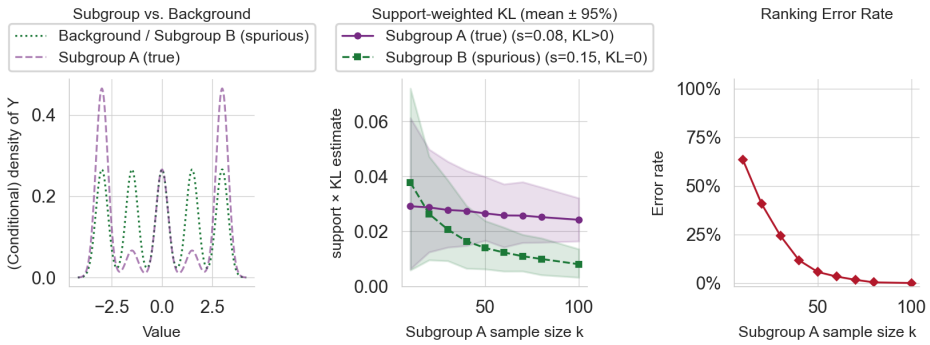

Figure 1: Ranking error under support-weighted KL estimation with finite samples. Two subgroups are drawn from a 5-mode Gaussian mixture background: Subgroup A (true, support $= 0.08$, KL ≈ 0.25) and Subgroup B (spurious, support $= 0.15$, KL $= 0$). **Left:** Conditional densities of the subgroups and background. **Middle:** Support-weighted KL estimates (support $\times$ KL) as a function of Subgroup A's sample size k, with B's size scaled by the support ratio; shaded regions show the 2.5th–97.5th percentile range over 300 draws. At small k, estimation noise inflates B's score above A's despite its zero true divergence. **Right:** Fraction of trials in which the spurious Subgroup B is ranked above the true Subgroup A. At $k = 30$ the error rate still exceeds 20%, showing that support-weighted KL ranking is unreliable at small sample sizes.

Research gap. What is missing is a rigorous understanding of what makes a subgroup maximally deviating *given an arbitrary support constraint*, together with a practical recipe for identifying the entire deviation-support Pareto front [12]. That is, rather than committing to a single implicit trade-off via a weighted score, one would ideally like to ask: *for any user-specified minimum support α, what is the most deviating subgroup?* Answering this question in its full generality has remained open:

existing exhaustive algorithms with guarantees [16, 25] and recent gradient-based methods [34] all target weighted objectives rather than the constrained problem directly.

Our approach. We approach subgroup discovery from a probabilistic modeling perspective. Our key insight is that when the data-generating distribution admits a partition-based structure where the feature space decomposes into regions with homogeneous conditional distributions $P(Y|X)$, which we refer to as *atoms*, the search for optimal subgroups at *every* support level collapses to a combinatorial search problem over a finite set of atoms. Specifically, we prove that under any support constraint, the optimal subgroup is a union of whole atoms together with at most one fractional atom. This structural result characterizes the full deviation-support Pareto front as a discrete selection problem over atoms that are determined by the distribution alone, independent of the support threshold. This suggests a clean two-step recipe: first learn a partition of the feature space into atoms with (approximately) homogeneous conditional distributions of the target, then select atoms to optimize deviation under any desired support constraint.

Contributions. Our contributions can be summarized as follows.

- We develop, to our knowledge, the first theoretical and structural characterization of KL-optimal subgroups under support constraints. Under a partition-based generative model, we prove that every point on the deviation-support Pareto front is a union of whole atoms plus at most one fractional atom (Theorem 3.5). Under distinct generating distributions, this structure is also necessary (Corollary 3.6).
- Our theory suggests a modular recipe for learning optimal subgroups from data, i.e., estimate a homogeneous partition, then select atoms, which reduces an uncountable search space to a finite combinatorial problem that can be solved at any user-specified support threshold.
- We instantiate the recipe with a partition estimator (CDTree [35]) and a greedy atom-selection algorithm (GAP), which together form the *AtomSD* framework. Across 19 benchmark datasets, AtomSD achieves the best average rank across a wide range of support thresholds, empirically validating the *sufficiency* of our theoretical results. When applied as a post-processing step, AtomSD also reliably improves subgroups returned by four existing methods, validating the *necessity* condition implied by our theorem.

2 Related work

Subgroup discovery was introduced by Klösgen [21] and Wrobel [33], building on an earlier formalization under the name data surveying [30]. Early work targeted discrete targets, where probability distributions are easier to estimate [22, 20, 2]. Common quality measures include weighted accuracy and its variants [22, 20], weighted relative accuracy [2], KL divergence for discrete variables [31], and weighted mean differences [21]; we refer to Atzmueller [3] and Herrera et al. [17] for comprehensive surveys. Further, a separate strand evaluates entire subgroup sets rather than individual subgroups, incorporating diversity between subgroups [32, 8, 6].

Recent work has shifted to the harder case of numeric targets. Some methods focus on specific quality measures such as weighted KL divergence and its variants [28, 29, 34, 7, 19]; others propose frameworks compatible with a range of measures [26]. All such measures fold support and deviation into a single score, committing implicitly to one trade-off. Our work instead targets the support-constrained problem directly: for any user-specified support threshold, find the most deviating subgroup. This yields the full support–deviation Pareto front [12] rather than a single point on it.

Most existing methods assume one-dimensional targets, but exceptional model mining methods handle multi-dimensional ones [23, 10, 11]. Our theorems do not require one-dimensional targets and apply in principle to this setting, though our empirical validation focuses on one-dimensional cases.

A parallel research line focuses on algorithms. Most are heuristic discrete-optimization procedures [29, 32, 22, 27, 28]. More recent work represents subgroup discovery as a neural network optimized by stochastic gradient [34], approximating the discrete problem with a continuous one. Algorithms with guarantees also exist, based on optimistic estimates [33, 26, 25, 16] or FP-growth [5, 4, 7, 19]. Our work is complementary to them as these algorithms all optimize quality measures that fold support and deviation together, so they do not directly solve the support-constrained problem we target.

3 Theory

We consider a supervised setting where the feature X can be continuous, discrete, or a mix of both, and the target Y can be binary, multi-class, or continuous. A subgroup is any measurable subset $Q \subseteq \mathcal{X}$, and we are interested in those whose conditional target distribution deviates from the population. We use P_X, P_Y, and $P_{Y|X}$ to denote the marginal distributions of X, Y, and the conditional distribution of Y given X, respectively. Throughout the development of our theory, we work with population-level probability distributions.

The degree of deviation of a subgroup Q is measured by the KL-divergence:

$$d(Q) = \mathrm{KL}(P(Y \mid X \in Q) \,\|\, P(Y)), \quad \forall Q \subseteq \mathcal{X},\ P(X \in Q) > 0. \tag{1}$$

The goal of subgroup discovery is to find the subgroup with the largest deviation, possibly with the support constraint that $P(X \in Q) \geq \alpha$. That is, in practice, analysts and users often look for subgroups large enough to be actionable (e.g., a subgroup of hospital patients large enough to support a future clinical trial). For interpretability, subgroup discovery methods commonly express Q as a rule or union of rules, where a *rule* is a conjunction of the form $\bigwedge_{j \in J}(X_j \in R_j)$ with each R_j an interval (if X_j is continuous) or a discrete set. Our theoretical results do not depend on this restriction, though it motivates Definition 3.2 below.

Our central question is: *what structure must a maximally deviating subgroup possess, and how does imposing a support constraint alter that structure?* We show that both questions admit sharp answers that together characterize the full deviation-support Pareto-front and reduce an otherwise intractable search to a structured combinatorial problem over a finite set of atoms.

In the following, we describe our theoretical results by first considering subgroups without support constraints, and then moving to our main theorem in the constrained case.

Subgroups without support constraints. We begin with the following observation that constrains what any optimal subgroup can look like. A subgroup *cannot* benefit from internal heterogeneity: if some sub-region of Q had a different conditional distribution from the rest, one could split Q along that boundary and keep the more deviating side, strictly improving on Q. The optimum must therefore be internally uniform. Formally (all proofs are deferred to Appendix A):

Theorem 3.1 (Homogeneity of Optimal Subgroups)**.** *Let $(X, Y) \sim P$. Suppose $Q^\star \subseteq \mathcal{X}$ maximizes $d(Q)$ over all measurable Q with $P(X \in Q) > 0$. Then the conditional distribution of Y is constant throughout $Q^\star$: for every measurable $Q' \subset Q^\star$ with $0 < P(X \in Q') < P(X \in Q^\star)$,*

$$P(Y \mid X \in Q') \;=\; P(Y \mid X \in Q^\star),$$

and consequently $d(Q') = d(Q^\star \setminus Q') = d(Q^\star)$.

Practically, it may be more insightful to read Theorem 3.1 in its contrapositive form, which does not require such a $Q^\star$ with positive mass to exist: if a subgroup Q is not homogeneous in the above sense, i.e., there exists $Q' \subset Q$ with $P(Y \mid X \in Q') \neq P(Y \mid X \in Q \setminus Q')$, then $d(Q) < \max\{d(Q'),\, d(Q \setminus Q')\}$ and hence Q must *not* be the maximizer.

Thus, Theorem 3.1 says the optimum is already internally uniform in a distributional sense: whatever region achieves maximum deviation, every non-trivial piece of it has the same conditional law as the whole. This is a strong internal-structure constraint that holds for *any* distribution P, without modeling assumptions. It also motivates viewing the feature space as built out of regions of homogeneous conditional distribution, each of which behaves as an *atom* for the purpose of subgroup discovery: splitting it never changes its conditional distribution. The following definition formalizes this picture and will serve as the generative model underlying the rest of our analysis.

Definition 3.2 (Partition-based Generative Model)**.** We say (X, Y) follows a *partition-based model* with generating subgroups $\mathcal{S} = \{S_1, \ldots, S_m\}$ if $\{S_i\}_{i \in \{1,\ldots,m\}}$ partition $\mathcal{X}$ (i.e., $\bigcup\{S_i\}_{i \in \{1,\ldots,m\}} = \mathcal{X}$ and all pairs $S_i \cap S_j = \emptyset$ for $i \neq j$), each atom has positive mass $P(X \in S_i) > 0$, and $P(Y \mid X = x) = P(Y \mid X \in S_i)$ for all $x \in S_i$ and all i.

This model class is a natural fit for the problem rather than a strong additional assumption. Every rule-based subgroup discovery method already returns a rule paired with a single probabilistic model of Y given the subgroup—this is what "a subgroup" means operationally in the literature. Assuming the data-generating process has the same form is therefore the standard realizability assumption

for this setting, analogous to assuming the true regressor lies in the hypothesis class in supervised learning. Real data will of course deviate from this model to varying degrees; our experiments in Section 5 test whether the structural consequences we derive below remain useful when it does.

Under this model, Theorem 3.1 has a much sharper consequence than mere internal homogeneity: the global search over all measurable subsets of $\mathcal{X}$ collapses to a search over just m candidates.

Theorem 3.3 (Optimality of Generating Subgroups)**.** *Under the partition-based model of Definition 3.2,*

$$\max_{Q \subseteq \mathcal{X}} d(Q) \;=\; \max_{i \in \{1,\dots,m\}} d(S_i).$$

Further, if the maximizer is unique, i.e., $d(S_k) > d(S_i)$ for all $i \neq k$, then the optimal subgroups are exactly S_k and, by homogeneity, its subsets: a subgroup Q with $P(X \in Q) > 0$ attains the maximum if and only if $P(X \in Q \setminus S_k) = 0$. In particular, no subgroup that places positive mass outside S_k, whether another atom or any union or mixture of atoms, attains $d(S_k)$.

The reduction here is dramatic: an uncountable search space becomes a finite one of size m, with no loss of optimality. Practically, this means that once a good partition of $\mathcal{X}$ is in hand, unconstrained subgroup discovery reduces to checking m candidates. This yields a modular recipe: first fit a partition-based model of $P(Y \mid X)$ via standard supervised learning (e.g., a decision tree with a non-parametric estimate of the conditional distribution at each leaf), then pick the leaf whose conditional distribution has the largest KL to the marginal. Subgroup discovery, which has classically required bespoke combinatorial search, thus reduces to conditional distribution estimation followed by a trivial argmax over leaves.

Subgroups with support constraints. In practice with finite samples, a minimum support constraint $P(X \in Q) \geq \alpha$ is usually required, since very small subgroups are statistically unreliable (see Appendix C.2 for more thorough discussions with a bootstrap experiment) or uninteresting (non-actionable). As we show next, the clean atom-level structure of Theorem 3.3 carries over to this constrained setting almost intact. We first state the problem formally.

Definition 3.4 (Support-Constrained Optimal Subgroup)**.** Given a minimum support threshold $\alpha \in (0, 1)$, define

$$Q^* \;=\; \operatorname*{arg\,max}_{Q \subseteq \mathcal{X}:\, P(X \in Q) \geq \alpha} d(Q). \tag{2}$$

Our core theoretical contribution, as shown by the next theorem, reveals that it is never advantageous to take partial slices from two different generating regions simultaneously. Therefore, Q^* is a union of whole atoms together **with at most one fractional atom**. Specifically, Theorem 3.5 below establishes that a subgroup of this form *suffices* for optimality; Corollary 3.6 shows that under distinct generating distributions this structure is also *necessary*.

Theorem 3.5 (Main Result: Structure of Support-Constrained Optimal Subgroups)**.** *Let $\mathcal{S} = \{S_1, \dots, S_m\}$ be the generating partition with capacities $\pi_i = P(X \in S_i)$, and assume the atoms are* divisible*: for every i and every $\beta \in [0, \pi_i]$, some $S' \subseteq S_i$ has $P(X \in S') = \beta$. For any $\alpha \in (0, 1)$, there* exists *an optimal solution Q^* to Definition 3.4 of the form*

$$Q^* \;=\; \left(\bigcup_{i \in \mathcal{I}} S_i \right) \cup S'_k,$$

for some index set $\mathcal{I}$ and some $S'_k \subseteq S_k$ with $k \notin \mathcal{I}$ (possibly $S'_k = \emptyset$), i.e., at most one atom is included partially.

Corollary 3.6 (Uniqueness under Distinct Distributions)**.** *Under the assumptions of Theorem 3.5, if the generating distributions $\{P(Y \mid X \in S_i)\}$ are distinct, then* every *optimal solution must have the structure of Theorem 3.5. In particular, no optimal subgroup can contain strictly partial amounts of two or more different generating regions.*

Theorem 3.5 gives a complete discrete characterization of the deviation-support Pareto front: at every support level α, the optimal subgroup is pinned down by which generating atoms to include in full, plus at most one fractional component. The entire Pareto front is therefore traced by a combinatorial selection over m atoms, rather than a continuous search over all measurable subsets of $\mathcal{X}$.

Two observations make this reduction strong. First, m is a property of the generating partition alone (at most the number of regions needed to separate the distinct conditional laws of Y given X), whereas the unreduced search over rules in continuous feature spaces effectively enumerates all candidate splitting points per feature—a far larger combinatorial object. Second, the atoms depend on P alone, independent of α: a single partition of $\mathcal{X}$ determines the atom set, and the same set traces the Pareto front across every support level.

We close the theory section with a remark on how these results resolve a longstanding design tension between tractability and expressiveness in the subgroup discovery literature.

Remark 3.7 (Representational Limits of Single-Conjunction Methods). Theorem 3.5 also sharpens a longstanding design tension in subgroup discovery. For tractability, most existing methods restrict Q to a conjunction of conditions and avoid *disjunctive normal form* (DNF), i.e., unions of rules, since searching over arbitrary DNFs is combinatorially expensive. Under realizability, with atoms given by rules (as in Section 4), however, the optimal subgroup *is* a DNF: a union of atomic rules. Any single-conjunction method is therefore representationally limited and will in general miss the optimum. Our atom-union formulation replaces the intractable search over arbitrary DNFs with a selection problem over a small set of m learned atoms, making DNF-aware subgroup discovery practical for the first time.

4 Learning the KL-optimal subgroup given any support threshold

Theorem 3.5 guarantees that the support-constrained optimum has the form $(\bigcup_{i\in\mathcal{I}} S_i) \cup S'_k$: a union of whole atoms together with at most one fractional atom. Learning such a subgroup from data therefore decomposes into two problems: estimating the atom set $\{S_i\}$, and selecting $\mathcal{I}$ and S'_k under the support constraint. We address each in turn and refer to the combined framework as *AtomSD*.

The theorem reduces the problem as far as structural arguments can: further simplification would require distributional assumptions beyond partition-based realizability. Even the sub-problem of selecting which whole atoms to include is NP-hard in the worst case—we give a reduction from SUBSET-SUM in Appendix E—which rules out a polynomial-time exact algorithm and motivates a principled heuristic. We adopt a greedy algorithm operating directly within the structural form guaranteed by Theorem 3.5.

We deliberately instantiate both steps with simple off-the-shelf components: if a simple instantiation of the theory already outperforms existing specialized algorithms, the gains are attributable to the structural simplification our theorems provide, not to algorithmic sophistication. While our theoretical results apply to both continuous and discrete targets, the instantiation and experiments focus on the arguably more challenging case of continuous targets.

Step 1: Learning an approximate generating partition via CDTree. We employ the conditional density estimation tree (CDTree) model [35] to induce a partition of $\mathcal{X}$ into disjoint leaf regions $\mathcal{S} = \{S_1, \ldots, S_K\}$, where each S_i is a conjunction of feature constraints paired with a non-parametric (histogram) estimate of $P(Y \mid X \in S_i)$. CDTree jointly optimizes the tree structure, splitting conditions, and leaf densities; model complexity is controlled via a minimum description length criterion, removing the need to tune regularization parameters [35]. The resulting leaves provide a data-driven approximation of the homogeneous conditional distributions required by our theory, and we treat them as *atomic subgroups* throughout, reusing the notation of Section 3 (with K rather than m to emphasize that the leaves are learned rather than given).

Step 2: Selecting atoms via Greedy Atomic Pruning (GAP). Following Theorem 3.5, a candidate subgroup is described by an index set $\mathcal{I} \subseteq [K] := \{1, \ldots, K\}$ of whole atoms together with an optional partial component $p \subseteq S_k$ for some $k \notin \mathcal{I}$; let $\mathrm{score}(\mathcal{I}, p)$ denote the empirical KL divergence of $(\bigcup_{i\in\mathcal{I}} S_i) \cup p$. GAP (Algorithm 1) maintains a current solution $(\mathcal{I}, p)$ and alternates two greedy moves until convergence. We slightly abuse terminology and also use "support" for its empirical counterpart, the number of training instances covered by the subgroup, and write α for the minimum support threshold in this sample-level sense as well. We describe each of them next.

SearchFull refines the index set $\mathcal{I}$ with p held fixed. Starting from $\mathcal{I} = [K]$ and $p = \varnothing$, it considers every single-atom toggle—adding an atom currently absent from $\mathcal{I}$, or removing one currently present—and commits to the best improving move, subject to the minimum-support constraint that

Algorithm 1 Greedy Atomic Pruning (GAP)

Input: Candidate atomic subgroups (CDTree leaves) $\mathcal{S} = \{S_1, \ldots, S_K\}$, training data (x^n, y^n), minimum support α. Throughout, $\textsc{Score}(\mathcal{I}, p)$ denotes the empirical KL divergence of $(\bigcup_{i \in \mathcal{I}} S_i) \cup p$ on the training data and $\textsc{supp}(\mathcal{I}, p)$ the number of training instances it covers.

Output: The learned subgroup, given as an index set $\mathcal{I} \subseteq [K]$ of whole atoms and an optional partial subgroup p.

$\mathcal{I} \leftarrow [K]$; $\quad p \leftarrow \varnothing$ ▷ Initialize with all candidate atoms
$(\mathcal{I}, p) \leftarrow \textsc{SearchFull}(\mathcal{I}, p)$ ▷ Initial greedy selection of whole atoms
repeat ▷ Alternate partial refinement and whole-atom selection
  $\mathit{improved} \leftarrow$ **false**
  $(\mathcal{I}', p') \leftarrow \textsc{SearchPartial}(\mathcal{I}, p)$ ▷ See Algorithm 3 in Appendix D
  if $\textsc{Score}(\mathcal{I}', p') > \textsc{Score}(\mathcal{I}, p)$ **then**
    $(\mathcal{I}, p) \leftarrow (\mathcal{I}', p')$; $\quad \mathit{improved} \leftarrow$ **true**
  end if
  $(\mathcal{I}', p') \leftarrow \textsc{SearchFull}(\mathcal{I}, p)$ ▷ See Algorithm 2 in Appendix D
  if $\textsc{Score}(\mathcal{I}', p') > \textsc{Score}(\mathcal{I}, p)$ **then**
    $(\mathcal{I}, p) \leftarrow (\mathcal{I}', p')$; $\quad \mathit{improved} \leftarrow$ **true**
  end if
until $\neg\, \mathit{improved}$
return $(\mathcal{I}, p)$

$(\bigcup_{i \in \mathcal{I}} S_i) \cup p$ covers at least α training instances. The loop terminates when no single-atom toggle improves score.

SearchPartial refines the fractional component p with the structure of $\mathcal{I}$ held fixed. For each candidate atom S_i, it considers using S_i as the fractional atom (removing i from $\mathcal{I}$ if present) and greedily shrinks its bounding box. At each step it considers every *cut*, that is, raising the lower bound or lowering the upper bound of a single feature to a value observed within the current box, and applies the cut that most improves score, until no cut improves it. The candidate $(\mathcal{I}', p'_i)$ achieving the highest score across all i is returned.

The two moves alternate until neither improves score. The output is a subgroup of the form prescribed by Theorem 3.5: a union of whole atoms plus at most one shrunken partial atom. Full pseudocode for SearchFull and SearchPartial is given in Algorithms 2 and 3 in Appendix D.

A second use of the theory: improving subgroups returned by other methods. Theorem 3.5 and Corollary 3.6 jointly imply a structural *necessity* condition: under distinct generating distributions, any Pareto-optimal subgroup must be a union of whole atoms plus at most one fractional atom. A subgroup that cuts partially through two or more atoms violates this condition and therefore cannot lie on the Pareto front; however, it can often be modified into one that does.

AtomSD exploits this observation directly. Given a subgroup Q produced by any existing subgroup discovery algorithm, we restrict $\mathcal{S}$ to the CDTree leaves overlapping Q, initialize $\mathcal{I}$ to their indices, and apply GAP (Algorithm 1) within this restricted candidate set. The resulting subgroup satisfies the necessity condition by construction. As we show in Section 5, this post-processing reliably improves subgroups produced by every competitor we test. This is evidence that the condition is frequently violated in practice, and that enforcing it recovers otherwise unavailable KL divergence.

5 Experiments

Our experiments aim to answer whether the structural insights of Section 3 transfer to practice, i.e., when the partition must be learned from finite data and the partition-based generative assumption holds only approximately. We organize the evaluation around Theorem 3.5 and Corollary 3.6: *sufficiency* (Section 5.2), that a subgroup of the atom-union form suffices for Pareto-optimality; and *necessity* (Section 5.3), that any subgroup violating this form can be improved by enforcing it.

5.1 General setups

Datasets. We use 19 benchmark datasets from the UCI repository [1], commonly used in the subgroup discovery literature [28, 35, 34], spanning a wide range of sample sizes and dimensionalities (see Table 2 in Appendix B).

Competitor algorithms. We compare against four methods: SyFlow [34], a state-of-the-art neural approach that optimizes a support-weighted KL objective via stochastic gradient descent; RSD [29], a strong heuristic-search method based on the minimum description length principle; and the classic subgroup discovery methods SD-μ and SD-KL [24] for numeric targets.

Evaluation. All results are averaged over 3-fold cross-validation: the training set is used to obtain subgroups, and KL divergence is estimated on the held-out test set to avoid the bias of training-set evaluation. Full reproducibility details are provided in Appendix B, and our source code is available at `https://github.com/ylincen/AtomSD`.

5.2 Sufficiency: AtomSD finds deviating subgroups at every support threshold

The theory predicts that the atom-union form suffices for Pareto-optimality at any support threshold. The empirical question is whether this remains true when the partition is learned from finite data rather than given exactly, and whether it translates into better subgroups than methods committed to a single implicit trade-off.

Table 1 shows a clear pattern. As the minimum support α increases, every competitor increasingly fails to return any subgroup satisfying the threshold; in this case, we fall back to treating the whole dataset as the trivial subgroup, which by construction yields $\mathrm{KL} = 0$. By $\alpha \geq 150$, RSD returns no valid subgroup on most datasets, SD-KL on about half, and SD-μ and SyFlow on a growing number of them. This is exactly the failure mode anticipated in the introduction: these methods optimize a folded quality measure that commits to one point on the Pareto front, and when the user's support requirement disagrees with that point, the method has nothing to fall back on. In principle one could re-tune each competitor's hyperparameters per threshold, but this requires knowledge of each algorithm's internal trade-off and defeats the purpose of a support-constrained interface.

AtomSD, in contrast, returns a non-trivial subgroup on nearly every dataset at every threshold, and the KL divergences remain substantial even under the tightest constraints. In aggregate, AtomSD achieves an average rank of 1.25 and 78 wins out of 95 (dataset, α) combinations, and the pattern is consistent across datasets of varying sample size and dimensionality. A Friedman test (with the Iman–Davenport correction [9]) rejects the null of equal performance at every support level ($p < 10^{-5}$ in each case); the corresponding critical-difference diagram [9] is in Appendix C.5. The gap is most pronounced on larger datasets such as *California*, *House*, and *Supercond*, where AtomSD sustains strong divergence at every α while most competitors degenerate.

We read this as direct evidence that the structural insight of Theorem 3.5 transfers to the finite-sample setting: even though the partition is estimated rather than given, and even though real data deviates from the partition-based model to varying degrees, targeting the atom-union form yields Pareto-optimal subgroups in practice. Finally, in some applications the support threshold is naturally expressed as a ratio rather than an absolute count; AtomSD performs equally strongly in that regime, as reported in Appendix C.3.

5.3 Necessity: enforcing the structural condition improves any subgroup

Corollary 3.6 implies a necessity condition: under distinct generating distributions, any Pareto-optimal subgroup *must* be a union of whole atoms plus at most one fractional atom. A subgroup that cuts partially through two or more atoms therefore cannot lie on the Pareto front, and can in principle be improved by enforcing the condition. We of course do not have access to the true atoms; we test instead whether enforcing the condition against the *learned* atoms still improves the subgroup, which would indicate that the necessity insight transfers to practice.

For each competitor's returned subgroup Q, we apply AtomSD as a post-processing step: restrict the candidate atoms to the CDTree leaves overlapping Q, and re-run GAP within that restricted footprint. Figure 2 reports the held-out KL-divergence difference between the refined subgroup and the original. Across all four competitors and every support threshold, the distribution is overwhelmingly positive:

Table 1: Average held-out KL divergence (higher is better) across 3-fold CV for support threshold α. **Bold** indicates the best method per row. If a method returns no subgroup satisfying the support threshold on none of the three folds, KL is set to 0 (the default, trivial subgroup covering all instances); otherwise, the reported value is averaged only over folds on which the method returns a valid subgroup. Held-out KL estimates can be negative; see Appendix B.

Dataset	α	**AtomSD**	RSD	SD-KL	SD-μ	SyFlow	Dataset	α	**AtomSD**	RSD	SD-KL	SD-μ	SyFlow
	50	**1.768**	1.363	0.981	1.053	1.312		50	**1.909**	1.035	-0.458	0.699	0.592
	100	**1.718**	1.244	0.981	1.053	1.312		100	**1.532**	0.000	-0.458	0.699	0.592
Ailerons	150	**1.684**	0.000	0.981	1.053	1.312	SkillCraft	150	**0.715**	0.000	-0.458	0.699	0.592
	200	**1.475**	0.000	0.981	1.053	1.312		200	**0.715**	0.000	-0.459	0.699	0.592
	300	**1.684**	0.000	0.981	1.053	1.312		300	**0.715**	0.000	0.000	0.699	0.645
	50	**3.679**	0.891	1.290	2.889	1.246		50	**2.506**	2.029	0.000	1.180	1.253
	100	**3.288**	0.882	1.290	2.889	1.246		100	**1.868**	0.000	0.000	1.180	1.063
Bikesharing	150	**3.151**	0.923	1.290	2.818	1.246	SML2010	150	**1.552**	0.000	0.000	1.180	1.063
	200	**3.043**	0.725	1.290	2.818	1.246		200	**1.397**	0.000	0.000	1.180	0.925
	300	2.771	0.000	1.290	**2.818**	1.246		300	**1.151**	0.000	0.000	0.000	0.948
	50	**3.936**	-0.694	0.000	1.316	0.747		50	**2.620**	0.855	0.839	0.773	0.665
	100	**3.936**	0.000	0.000	1.316	0.747		100	**2.956**	0.896	0.839	0.773	0.665
California	150	**3.936**	0.000	0.000	1.316	0.747	Supercond	150	**3.023**	0.000	0.839	0.773	0.665
	200	**3.936**	0.000	0.000	1.316	0.747		200	**2.925**	0.000	0.839	0.773	0.665
	300	**2.504**	0.000	0.000	1.316	0.747		300	**2.672**	0.000	0.839	0.773	0.665
	50	**1.278**	1.174	0.000	1.220	1.213		50	**0.919**	0.119	-0.181	-0.175	-0.163
	100	**0.594**	0.000	0.000	0.000	0.460		100	**0.589**	0.124	-0.181	-0.226	-0.163
Concrete	150	**0.510**	0.000	0.000	0.000	0.186	Support2	150	**0.801**	0.125	-0.181	-0.253	-0.163
	200	**0.390**	0.000	0.000	0.000	0.000		200	**0.406**	0.125	-0.181	-0.276	-0.163
	300	**0.266**	0.000	0.000	0.000	0.000		300	**0.142**	0.000	-0.181	0.000	-0.163
	50	0.318	0.000	-0.268	-0.100	**0.447**		50	1.375	**1.525**	0.000	1.204	0.965
	100	0.318	0.000	-0.268	-0.100	**0.447**		100	**0.846**	0.000	0.000	0.000	0.789
Elevators	150	0.318	0.000	-0.268	-0.100	**0.447**	Synchronous	150	**0.593**	0.000	0.000	0.000	0.554
	200	0.318	0.000	-0.268	-0.100	**0.447**		200	**0.449**	0.000	0.000	0.000	0.000
	300	0.318	0.000	-0.268	-0.100	**0.447**		300	**0.106**	0.000	0.000	0.000	0.000
	50	**2.272**	2.229	1.088	1.250	0.844		50	**0.946**	0.554	0.000	0.000	0.451
	100	1.846	**2.017**	1.088	1.250	0.844		100	**0.591**	0.412	0.000	0.000	0.272
Energy	150	**1.955**	1.720	1.088	1.250	0.844	Thermography	150	**0.524**	0.349	0.000	0.000	0.000
	200	**1.871**	1.720	1.088	1.250	0.844		200	**0.319**	0.289	0.000	0.000	0.000
	300	**1.800**	0.000	1.088	1.250	0.844		300	**0.243**	0.000	0.000	0.000	0.000
	50	0.000	0.131	0.000	0.000	**0.579**		50	**0.897**	0.465	0.000	0.000	0.617
	100	0.000	0.000	0.000	0.000	0.000		100	**0.682**	0.334	0.000	0.000	0.544
ForestFires	150	0.000	0.000	0.000	0.000	0.000	Toxicity	150	**0.385**	0.000	0.000	0.000	0.000
	200	0.000	0.000	0.000	0.000	0.000		200	**0.250**	0.000	0.000	0.000	0.000
	300	0.000	0.000	0.000	0.000	0.000		300	**0.088**	0.000	0.000	0.000	0.000
	50	**3.609**	0.000	0.000	1.501	0.447		50	1.605	1.282	0.000	**1.923**	0.817
	100	**3.141**	0.000	0.000	1.468	0.447		100	**1.302**	0.000	0.000	0.000	0.817
House	150	**2.357**	0.000	0.000	0.000	0.447	Treasury	150	**1.184**	0.000	0.000	0.000	0.817
	200	**1.559**	0.000	0.000	0.000	0.447		200	**1.006**	0.000	0.000	0.000	0.817
	300	**1.495**	0.000	0.000	0.000	0.447		300	**0.782**	0.000	0.000	0.000	0.607
	50	**1.525**	1.242	0.878	0.449	0.952		50	1.516	**1.801**	1.629	1.072	1.016
	100	**1.649**	1.420	0.878	0.449	0.952		100	1.534	1.226	**1.629**	1.072	1.016
NavalProp	150	**1.693**	0.000	0.494	0.449	0.813	Wizmir	150	**1.429**	0.000	0.000	1.072	1.016
	200	**1.681**	0.000	0.494	0.120	0.448		200	**1.115**	0.000	0.000	1.072	1.016
	300	**1.635**	0.000	0.494	0.120	0.003		300	0.839	0.000	0.000	0.000	**0.920**
	50	**3.174**	3.088	0.574	1.913	0.439							
	100	**2.686**	1.885	0.574	1.913	0.439							
Puma32h	150	**2.640**	1.312	0.574	1.913	0.439							
	200	**2.549**	1.361	0.574	1.913	0.439							
	300	**2.335**	0.000	0.574	1.913	0.439							
#Best	50	**14**	2	0	1	2	Avg. Rank	50	**1.42**	2.61	4.29	3.32	3.37
	100	**15**	1	1	0	1		100	**1.26**	3.24	3.97	3.42	3.11
	150	**17**	0	0	0	1		150	**1.16**	3.79	3.89	3.18	2.97
	200	**17**	0	0	0	1		200	**1.16**	3.74	3.79	3.13	3.18
	300	**15**	0	0	1	2		300	**1.26**	4.08	3.63	3.03	3.00
Total	–	**78**	3	1	2	7	*Overall*	–	**1.25**	3.49	3.92	3.22	3.13

medians are consistently above zero, and negative cases are rare and mostly small in magnitude. This pattern holds regardless of which baseline produced the initial subgroup, which we interpret as evidence that 1) the necessity condition is frequently violated in practice, and 2) enforcing it recovers additional KL divergence beyond what the original algorithm captured. The corresponding results when the support threshold is expressed as a ratio are reported in Appendix C.4 and show the same pattern.

Beyond the theoretical validation, this post-processing use of AtomSD has practical value: in applications where a domain expert or prior analysis has identified a candidate subgroup of interest, AtomSD can refine it into a Pareto-optimal subgroup built from the atoms overlapping the original region.

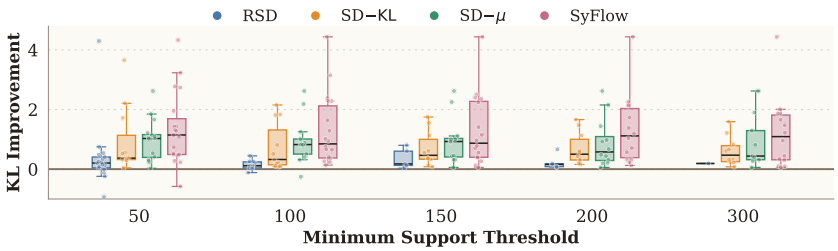

Figure 2: Held-out KL-divergence difference between AtomSD's refined subgroup and each competitor's original subgroup, across 19 datasets and five support thresholds. Positive values indicate that AtomSD's refinement improves the given subgroups.

6 Conclusion

We revisited subgroup discovery from a probabilistic modeling perspective and gave a structural characterization of KL-optimal subgroups under support constraints. Under a partition-based generative model, the entire deviation-support Pareto front is traced by unions of whole atoms together with at most one fractional atom, collapsing an uncountable search over measurable subsets to a finite combinatorial problem. This structural reduction is both 1) sufficient as an optimal subgroup of this form always exists at every support level, and 2) necessary (under distinct generating distributions), as no optimal subgroup can have any other form. Together, these results overcome what has been a longstanding challenge in the literature between committing to an implicit support–deviation trade-off via a weighted scalar measure and explicitly searching under user-specified support constraints. Our experiments verify that this structural insight continues to guide useful subgroup discovery even when the partition is estimated from finite data and the generative assumption holds only approximately.

Limitations and future work. Our theoretical guarantees hold under a partition-based generative model that only approximates real data; our experiments indicate that the resulting recipe remains useful when this assumption is approximately satisfied, but formal robustness results to distributional misspecification remain open. Further, future directions may include extending our theoretical results to learning subgroup sets in which the diversity among subgroups is considered.

Acknowledgments and Disclosure of Funding

We thank Jilles Vreeken for helpful discussions. This research was partly supported by the Dutch Research Council (NWO) through the XAIPre project (project number 19455, `https://doi.org/10.61686/XDODD77262`).

References

[1] The uci machine learning repository. URL `https://archive.ics.uci.edu/`.

[2] T. Abudawood and P. Flach. Evaluation measures for multi-class subgroup discovery. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 35–50. Springer, 2009.

[3] M. Atzmueller. Subgroup discovery. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 5(1):35–49, 2015.

[4] M. Atzmueller and F. Lemmerich. Fast subgroup discovery for continuous target concepts. In *International symposium on methodologies for intelligent systems*, pages 35–44. Springer, 2009.

[5] M. Atzmueller and F. Puppe. Sd-map–a fast algorithm for exhaustive subgroup discovery. In *European Conference on Principles of Data Mining and Knowledge Discovery*, pages 6–17. Springer, 2006.

[6] A. Belfodil, A. Belfodil, A. Bendimerad, P. Lamarre, C. Robardet, M. Kaytoue, and M. Plantevit. Fssd-a fast and efficient algorithm for subgroup set discovery. In *2019 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pages 91–99. IEEE, 2019.

[7] M. Boley, B. R. Goldsmith, L. M. Ghiringhelli, and J. Vreeken. Identifying consistent statements about numerical data with dispersion-corrected subgroup discovery. *Data Mining and Knowledge Discovery*, 31(5):1391–1418, 2017.

[8] G. Bosc, J.-F. Boulicaut, C. Raïssi, and M. Kaytoue. Anytime discovery of a diverse set of patterns with monte carlo tree search. *Data mining and knowledge discovery*, 32(3):604–650, 2018.

[9] J. Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine learning research*, 7(Jan):1–30, 2006.

[10] W. Duivesteijn, A. Knobbe, A. Feelders, and M. van Leeuwen. Subgroup discovery meets bayesian networks–an exceptional model mining approach. In *2010 IEEE International Conference on Data Mining*, pages 158–167. IEEE, 2010.

[11] W. Duivesteijn, A. J. Feelders, and A. Knobbe. Exceptional model mining: Supervised descriptive local pattern mining with complex target concepts. *Data Mining and Knowledge Discovery*, 30(1):47–98, 2016.

[12] M. T. Emmerich and A. H. Deutz. A tutorial on multiobjective optimization: fundamentals and evolutionary methods. *Natural computing*, 17(3):585–609, 2018.

[13] J. A. Esterhuizen, B. R. Goldsmith, and S. Linic. Interpretable machine learning for knowledge generation in heterogeneous catalysis. *Nature catalysis*, 5(3):175–184, 2022.

[14] D. Freedman and P. Diaconis. On the histogram as a density estimator: L 2 theory. *Zeitschrift für Wahrscheinlichkeitstheorie und verwandte Gebiete*, 57(4):453–476, 1981.

[15] J. Fürnkranz, D. Gamberger, and N. Lavrač. *Foundations of rule learning*. Springer Science & Business Media, 2012.

[16] H. Grosskreutz, S. Rüping, and S. Wrobel. Tight optimistic estimates for fast subgroup discovery. In *Joint European conference on machine learning and knowledge discovery in databases*, pages 440–456. Springer, 2008.

[17] F. Herrera, C. J. Carmona, P. González, and M. J. Del Jesus. An overview on subgroup discovery: foundations and applications. *Knowledge and information systems*, 29(3):495–525, 2011.

[18] M. Iferroudjene, C. Lonjarret, C. Robardet, M. Plantevit, and M. Atzmueller. Methods for explaining top-n recommendations through subgroup discovery. *Data Mining and Knowledge Discovery*, 37(2):833–872, 2023.

[19] J. Kalofolias, M. Boley, and J. Vreeken. Efficiently discovering locally exceptional yet globally representative subgroups. In *2017 IEEE International Conference on Data Mining (ICDM)*, pages 197–206. IEEE, 2017.

[20] B. Kavšek and N. Lavrač. Apriori-sd: Adapting association rule learning to subgroup discovery. *Applied Artificial Intelligence*, 20(7):543–583, 2006.

[21] W. Klösgen. Explora: A multipattern and multistrategy discovery assistant. In *Advances in knowledge discovery and data mining*, pages 249–271. 1996.

[22] N. Lavrač, B. Kavšek, P. Flach, and L. Todorovski. Subgroup discovery with cn2-sd. *Journal of Machine Learning Research*, 5(Feb):153–188, 2004.

[23] D. Leman, A. Feelders, and A. Knobbe. Exceptional model mining. In *Joint European conference on machine learning and knowledge discovery in databases*, pages 1–16. Springer, 2008.

[24] F. Lemmerich and M. Becker. pysubgroup: Easy-to-use subgroup discovery in python. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 658–662, 2018.

[25] F. Lemmerich, M. Becker, and F. Puppe. Difference-based estimates for generalization-aware subgroup discovery. In *Joint European conference on machine learning and knowledge discovery in databases*, pages 288–303. Springer, 2013.

[26] F. Lemmerich, M. Atzmueller, and F. Puppe. Fast exhaustive subgroup discovery with numerical target concepts. *Data Mining and Knowledge Discovery*, 30(3):711–762, 2016.

[27] M. Meeng and A. Knobbe. For real: a thorough look at numeric attributes in subgroup discovery. *Data Mining and Knowledge Discovery*, 35(1):158–212, 2021.

[28] H. M. Proença, P. Grünwald, T. Bäck, and M. v. Leeuwen. Discovering outstanding subgroup lists for numeric targets using mdl. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 19–35. Springer, 2020.

[29] H. M. Proença, P. Grünwald, T. Bäck, and M. van Leeuwen. Robust subgroup discovery: Discovering subgroup lists using mdl. *Data Mining and Knowledge Discovery*, 36(5):1885–1970, 2022.

[30] A. Siebes. Data surveying: Foundations of an inductive query language. In *KDD*, pages 269–274, 1995.

[31] M. van Leeuwen. Maximal exceptions with minimal descriptions. *Data Mining and Knowledge Discovery*, 21(2):259–276, 2010.

[32] M. Van Leeuwen and A. Knobbe. Diverse subgroup set discovery. *Data Mining and Knowledge Discovery*, 25(2):208–242, 2012.

[33] S. Wrobel. An algorithm for multi-relational discovery of subgroups. In *European symposium on principles of data mining and knowledge discovery*, pages 78–87. Springer, 1997.

[34] S. Xu, N. P. Walter, J. Kalofolias, and J. Vreeken. Learning exceptional subgroups by end-to-end maximizing KL-divergence. In *Forty-first International Conference on Machine Learning*, 2024.

[35] L. Yang and M. van Leeuwen. Conditional density estimation with histogram trees. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

A Proofs of main theorems

Theorem 3.1 (Homogeneity of Optimal Subgroups)**.** *Let $(X, Y) \sim P$. Suppose $Q^\star \subseteq \mathcal{X}$ maximizes $d(Q)$ over all measurable Q with $P(X \in Q) > 0$. Then the conditional distribution of Y is constant throughout $Q^\star$: for every measurable $Q' \subset Q^\star$ with $0 < P(X \in Q') < P(X \in Q^\star)$,*

$$P(Y \mid X \in Q') \;=\; P(Y \mid X \in Q^\star),$$

and consequently $d(Q') = d(Q^\star \setminus Q') = d(Q^\star)$.

Proof. Fix a subset Q with $P(X \in Q) > 0$. Let $A \subset Q$ be a measurable subset with $0 < P(X \in A) < P(X \in Q)$. Define

$$\lambda = P(X \in A \mid X \in Q) \in (0, 1),$$
$$P_A = P(Y \mid X \in A),$$
$$P_{A^c} = P(Y \mid X \in Q \setminus A).$$

Then the conditional distribution within Q decomposes as

$$P_Q \;=\; P(Y \mid X \in Q) \;=\; \lambda P_A + (1 - \lambda) P_{A^c}.$$

By convexity of the KL divergence in its first argument,

$$\begin{aligned}\mathrm{KL}(P_Q \,\|\, P(Y)) \;\le\; &\lambda \mathrm{KL}(P_A \,\|\, P(Y)) \\ &+ (1 - \lambda)\mathrm{KL}(P_{A^c} \,\|\, P(Y)),\end{aligned}$$

with equality if and only if $P_A = P_{A^c} = P_Q$.

That is,

$$d(Q) \;\le\; \lambda d(A) + (1 - \lambda) d(Q \setminus A) \;\le\; \max\{d(A),\, d(Q \setminus A)\}.$$

Now let $Q = Q^\star$ be a global maximizer. Since $d(Q^\star) \ge d(A)$ and $d(Q^\star) \ge d(Q^\star \setminus A)$, the inequality above forces equality throughout. Thus,

$$d(A) = d(Q^\star \setminus A) = d(Q^\star),$$

and moreover $P_A = P_{A^c} = P_{Q^\star}$, which shows that $P(Y \mid X \in Q^\star)$ is homogeneous across all measurable subsets of $Q^\star$ with positive probability mass. □

Theorem 3.3 (Optimality of Generating Subgroups)**.** *Under the partition-based model of Definition 3.2,*

$$\max_{Q \subseteq \mathcal{X}} d(Q) \;=\; \max_{i \in \{1,\ldots,m\}} d(S_i).$$

Further, if the maximizer is unique, i.e., $d(S_k) > d(S_i)$ for all $i \neq k$, then the optimal subgroups are exactly S_k and, by homogeneity, its subsets: a subgroup Q with $P(X \in Q) > 0$ attains the maximum if and only if $P(X \in Q \setminus S_k) = 0$. In particular, no subgroup that places positive mass outside S_k, whether another atom or any union or mixture of atoms, attains $d(S_k)$.

Proof. Write $P_i := P(Y \mid X \in S_i)$ for the generating distribution of atom S_i. Let Q be any subgroup with $P(X \in Q) > 0$, and let $w_i = P(X \in S_i \mid X \in Q)$ be the proportion of Q that falls into S_i, so that $w_i \ge 0$ and $\sum_{i=1}^m w_i = 1$. Under the partition-based model,

$$P_Q := P(Y \mid X \in Q) = \sum_{i=1}^m w_i P_i.$$

Since $\mathrm{KL}(\cdot \,\|\, P(Y))$ is convex in its first argument, Jensen's inequality gives

$$d(Q) \;=\; \mathrm{KL}\Big(\sum_{i=1}^m w_i P_i \,\Big\|\, P(Y)\Big) \;\le\; \sum_{i=1}^m w_i\, d(S_i) \;\le\; \max_i d(S_i). \tag{3}$$

Each S_i is itself a feasible subgroup ($w_i = 1$), so the upper bound is attained and $\max_Q d(Q) = \max_i d(S_i)$.

Now suppose the maximizer k is unique, i.e., $d(S_i) < d(S_k)$ for all $i \neq k$. If $P(X \in Q \setminus S_k) > 0$, then $w_k < 1$, so some $j \neq k$ has $w_j > 0$, and the second inequality in (3) is strict:

$$\sum_i w_i\, d(S_i) \;=\; d(S_k) - \sum_{i \neq k} w_i \big(d(S_k) - d(S_i)\big) \;<\; d(S_k).$$

Hence $d(Q) < d(S_k)$, and Q is not optimal. Conversely, if $P(X \in Q \setminus S_k) = 0$, then $w_k = 1$, so $P_Q = P_k$ and $d(Q) = d(S_k)$. Therefore Q is optimal if and only if $P(X \in Q \setminus S_k) = 0$. □

Theorem 3.5 (Main Result: Structure of Support-Constrained Optimal Subgroups)**.** *Let $\mathcal{S} = \{S_1, \ldots, S_m\}$ be the generating partition with capacities $\pi_i = P(X \in S_i)$, and assume the atoms are* divisible*: for every i and every $\beta \in [0, \pi_i]$, some $S' \subseteq S_i$ has $P(X \in S') = \beta$. For any $\alpha \in (0, 1)$, there* exists *an optimal solution Q^* to Definition 3.4 of the form*

$$Q^* \;=\; \left(\bigcup_{i \in \mathcal{I}} S_i \right) \cup S'_k,$$

for some index set $\mathcal{I}$ and some $S'_k \subseteq S_k$ with $k \notin \mathcal{I}$ (possibly $S'_k = \emptyset$), i.e., at most one atom is included partially.

Proof. Under the partition-based model, $d(Q)$ depends on Q only through its allocation $\big(P(X \in Q \cap S_i)\big)_{i=1}^m$, since $P(Y \mid X \in Q)$ is the mixture of the P_i with weights proportional to these masses. Since the atoms are divisible, every allocation with $x_i \in [0, \pi_i]$ is realized by some subgroup, so we may optimize over allocations directly. An optimal allocation exists: the objective is continuous (and finite, by positivity of the atoms) on the compact feasible set $\{\mathbf{x} \in \prod_i [0, \pi_i] : \sum_i x_i \geq \alpha\}$. Let $\mathbf{x}^* = (x_1^*, \ldots, x_m^*)$ be an optimal allocation vector for Definition 3.4, where $x_i^* = P(X \in Q^* \cap S_i) \in [0, \pi_i]$ with $\pi_i = P(X \in S_i)$. We show that there exists an optimal allocation in which at most one coordinate satisfies $0 < x_i < \pi_i$ strictly; this coordinate corresponds to the fractional atom S'_k, and the remaining $x_i \in \{0, \pi_i\}$ determine the index set $\mathcal{I}$.

Suppose that some optimal $\mathbf{x}^*$ has two distinct indices j, k with $0 < x_j^* < \pi_j$ and $0 < x_k^* < \pi_k$. We construct a perturbation along the (j, k) direction and show that either a strictly better feasible point exists (contradicting optimality) or a boundary point attains the same objective (reducing the number of strictly partial atoms by one).

Consider the one-parameter family $\mathbf{x}(\epsilon)$ obtained from $\mathbf{x}^*$ by setting $x_j(\epsilon) = x_j^* + \epsilon$ and $x_k(\epsilon) = x_k^* - \epsilon$, holding all other coordinates fixed. Since $\sum_i x_i(\epsilon) = \sum_i x_i^*$ is preserved, the support constraint $P(X \in Q) \geq \alpha$ remains satisfied for any ϵ such that $x_j(\epsilon) \in [0, \pi_j]$ and $x_k(\epsilon) \in [0, \pi_k]$. Let $[\epsilon_{\min}, \epsilon_{\max}]$ denote this feasible interval, which contains $\epsilon = 0$ in its interior by assumption.

Let $g(\epsilon) = d(Q(\mathbf{x}(\epsilon)))$ denote the objective along this segment. Under the partition-based model, $P(Y \mid X \in Q(\mathbf{x}(\epsilon)))$ is a mixture $\sum_i w_i(\epsilon) P_i$ where $w_i(\epsilon) = x_i(\epsilon) / \sum_i x_i(\epsilon)$. Since $\sum_i x_i(\epsilon)$ is constant in ϵ, the mixture weights $w_i(\epsilon)$ are affine in ϵ. The KL divergence is convex in its first argument, so $g(\epsilon)$ is a convex function of ϵ.

Case 1 ($P_j \neq P_k$). The mixture distribution $\sum_i w_i(\epsilon) P_i$ changes non-trivially with ϵ (since only w_j and w_k vary, and $P_j \neq P_k$), so g is *strictly* convex on $[\epsilon_{\min}, \epsilon_{\max}]$(all values are finite, since $dP_i/dP(Y) \leq 1/\pi_i$ by positivity of the atoms). The maximum of a strictly convex function on a closed interval is attained at one of the endpoints, and not at any interior point. Since $\epsilon = 0$ is interior, $g(\epsilon_{\min})$ or $g(\epsilon_{\max})$ strictly exceeds $g(0)$, contradicting the optimality of $\mathbf{x}^*$.

Case 2 ($P_j = P_k$). The mixture $\sum_i w_i(\epsilon) P_i$ is invariant to shifting mass between j and k, since $(w_j(\epsilon) + w_k(\epsilon)) P_j = (w_j(\epsilon) + w_k(\epsilon)) P_k$ equals $w_j(0) P_j + w_k(0) P_k$ at every ϵ. Hence $g(\epsilon)$ is constant on $[\epsilon_{\min}, \epsilon_{\max}]$. In particular, $g(\epsilon_{\min}) = g(\epsilon_{\max}) = g(0)$, so we may replace $\mathbf{x}^*$ with the allocation at either endpoint without changing the objective. At an endpoint, at least one of x_j, x_k equals 0 or its capacity (π_j or π_k, respectively), reducing the number of strictly partial coordinates by one.

Combining the two cases: any optimal $\mathbf{x}^*$ with two or more strictly partial coordinates can be replaced with an equally-optimal allocation having strictly fewer such coordinates. Iterating this reduction terminates with an optimal allocation $\tilde{\mathbf{x}}$ in which at most one coordinate is strictly partial. Setting $\mathcal{I} = \{i : \tilde{x}_i = \pi_i\}$, k equal to the unique strictly-partial index (if any) with $S'_k \subseteq S_k$ the

corresponding partial inclusion, and $S'_k = \emptyset$ otherwise, yields a solution of the form claimed in the theorem statement. □

Corollary 3.6 (Uniqueness under Distinct Distributions)**.** *Under the assumptions of Theorem 3.5, if the generating distributions* $\{P(Y \mid X \in S_i)\}$ *are distinct, then* every *optimal solution must have the structure of Theorem 3.5. In particular, no optimal subgroup can contain strictly partial amounts of two or more different generating regions.*

Proof. Let Q be optimal with allocation $\mathbf{x}^*$, and suppose two indices $j \neq k$ satisfy $0 < x_j^* < \pi_j$ and $0 < x_k^* < \pi_k$. Since the generating distributions are distinct, $P_j \neq P_k$, so Case 1 of the proof of Theorem 3.5 applies: g is strictly convex on $[\epsilon_{\min}, \epsilon_{\max}]$ with 0 in the interior, and one endpoint allocation has a strictly larger objective. That allocation is feasible and, by divisibility, realized by a subgroup, contradicting the optimality of Q. Hence at most one atom is strictly partial in Q: up to a set of probability zero, $Q = \left(\bigcup_{i \in \mathcal{I}} S_i\right) \cup S'_k$ with $\mathcal{I} = \{i : x_i^* = \pi_i\}$. □

B Additional Details of Experimental Setup

Datasets and implementation details. The processed datasets are summarized in Table 2. In total, we use 19 real-world datasets drawn from diverse application domains and commonly used in existing literature [28], [35], or [34]. All nominal features are one-hot encoded. We benchmark our method against four competitor algorithms, with all results obtained on the held-out folds using three-fold cross-validation. For the competing methods, we use the default hyperparameter settings provided in their original implementations to ensure a fair comparison without over-tuning, while our method requires no hyperparameter tuning. All experiments are conducted on a cluster equipped with AMD EPYC 7662 64-Core Processors.

Table 2: Summary of the datasets used in the experiments

Dataset	Sample Sizes	#Discrete Features	#Continuous Features
Ailerons	13750	4	36
Bikesharing	17379	13	4
California	20640	5	3
Concrete	1030	0	8
Elevators	16599	4	14
Energy	9568	0	4
ForestFires	517	0	12
House	22784	1	15
NavalProp	11934	0	16
Puma32h	8192	0	32
SkillCraft	3338	0	18
SML2010	2764	0	20
Supercond	21263	0	81
Support2	9105	0	26
Synchronous	557	0	4
Thermography	1003	0	25
Toxicity	908	0	6
Treasury	1049	0	15
Wizmir	1461	0	9

Estimating KL for held-out test datasets. Similar to SyFlow [34], we evaluate all discovered subgroups by estimating the KL divergence between the subgroup's conditional target distribution and the overall target distribution using histogram densities on the held-out test set. Specifically, bin edges are determined from training data via the Freedman-Diaconis rule [14], with separate bin grids for the subgroup and overall distributions. Both grids are clamped to the global target range $[y_{\min}, y_{\max}]$ to increase the stability especially for small sample size cases. The overall histogram uses Laplace smoothing (pseudo-count of 1 per bin), while the subgroup histogram uses a Bayesian Dirichlet prior whose concentration parameters are proportional to the overall bin probabilities, with

total prior strength equal to the number of bins. This provides data-driven shrinkage toward the overall distribution for bins with few subgroup observations. Both smoothed histograms are converted to densities by dividing by total count and bin width. The KL divergence is then estimated via a sample-wise approximation, i.e., the average of $\log \hat{p}(y) - \log \hat{q}(y)$ over the held-out subgroup members, where $\hat{p}$ and $\hat{q}$ are the subgroup and overall densities fitted on the training split. Because the densities are fitted on training data and evaluated on held-out points, this estimate can be negative (see, e.g., RSD on California in Table 1); a negative value indicates a subgroup whose training-set deviation does not generalize to the held-out split.

All bin edges and density parameters are learned from training data; at test time only bin assignments are computed. This train/test separation eliminates the optimistic bias that arises when the same data is used for both subgroup selection and evaluation. However, while the bias is controlled, the *variance* of the KL estimate remains high for small subgroups: as we demonstrate in Appendix C.2, bootstrap resampling reveals that subgroups with fewer than 50 training points yield KL estimates with prohibitively large variance across replicates.

C Additional Experiments

C.1 Support-weighted KL-divergence leads to finding spurious subgroups

We provide the experiment details for the reproducibility of Figure 1 in Section 1.

Distributions. The background distribution P_0 is a 5-component Gaussian mixture with equal weights and modes at $\{-3, -1.5, 0, 1.5, 3\}$, each with standard deviation $\sigma = 0.3$. We define two subgroups:

- **Subgroup A (true):** A 5-component Gaussian mixture sharing the same modes and standard deviations as the background, but with weights $(0.35, 0.05, 0.20, 0.05, 0.35)$ that over-represent the extreme modes. This creates a genuine distributional shift ($\mathrm{KL}(P_A \| P_0) > 0$). Its population support is $s_A = 0.08$.
- **Subgroup B (spurious):** Drawn from the background distribution itself, so $\mathrm{KL}(P_B \| P_0) = 0$ by construction. Its population support is $s_B = 0.15$.

Procedure. For each subgroup sample size $k_A \in \{10, 20, \ldots, 80\} \cup \{100\}$, we set $k_B = \lceil k_A \cdot s_B / s_A \rceil$ to reflect the proportionally larger sample size of the higher-support subgroup. We repeat the following 300 times independently:

1. Draw k_A i.i.d. samples from subgroup A's distribution and k_B i.i.d. samples from subgroup B's (i.e., the background).
2. Estimate the KL divergence of each subgroup against the background using histogram density estimation with Freedman–Diaconis binning and Laplace smoothing. The background density P_0 is estimated once from a fixed reference sample of $10{,}000$ points; both subgroup densities are estimated on the same bin grid.
3. Compute the support-weighted score for each subgroup: $\mathrm{score} = s \times \widehat{\mathrm{KL}}$.
4. Record whether subgroup B incorrectly outranks subgroup A, i.e., $\mathrm{score}_B > \mathrm{score}_A$.

Results. The result is shown in Figure 1. The left panel displays the true conditional densities of subgroup A versus the background (which coincides with subgroup B). The middle panel plots the mean and 95% quantile band of the support-weighted KL estimates as a function of k_A. The right panel reports the ranking error rate—the fraction of repetitions in which the spurious subgroup B's score exceeds that of the true subgroup A. At small sample sizes ($k_A \leq 30$), the error rate exceeds 20%, showing that KL estimation noise combined with the higher support of the spurious subgroup leads to frequent mis-rankings.

C.2 Instability of the estimated KL

To assess the reliability of KL divergence estimates for subgroups of varying sizes, we perform a bootstrap stability analysis on all subgroups discovered by five algorithms (AtomSD, RSD, SD-KL, SD-μ, SyFlow) across all datasets and cross-validation folds.

For each subgroup, we run $B = 300$ bootstrap replicates. In each replicate, we resample both the training and test sets independently with replacement, preserving the original train/test split ratio. Subgroup membership is inherited from the original indices: if a resampled row corresponds to an original subgroup member, it is included in the bootstrap subgroup (possibly with duplicates). We then re-estimate the KL divergence between the subgroup's target distribution P and the overall target distribution $P(Y)$ using the same histogram-based procedure employed during subgroup discovery: bin edges are determined separately for the subgroup and overall distributions via the Freedman–Diaconis rule, with a shared global range $[\min(Y) - \delta,\ \max(Y) + \delta]$ for a small margin δ; the overall histogram receives Laplace smoothing ($+1$ per bin), while the subgroup histogram receives a Bayesian correction using the overall histogram as a Dirichlet prior with total pseudo-count equal to the number of bins; KL divergence is estimated by the sample-wise Monte Carlo average $\widehat{\mathrm{KL}} = \frac{1}{n_Q} \sum_{i=1}^{n_Q} [\log \hat{p}(y_i) - \log \hat{q}(y_i)]$, where $\hat{p}$ and $\hat{q}$ are the smoothed subgroup and overall histogram densities and the n_Q evaluation points $\{y_i\}$ are the subgroup members in the current split.

For each subgroup we record the standard deviation of $\widehat{\mathrm{KL}}$ across the 300 bootstrap replicates, computed separately for the training and test splits. Subgroups are grouped into five size buckets by training set membership: <50, 50–100, 100–200, 200–300, and 300+.

Notably, evaluating $\widehat{\mathrm{KL}}$ on a held-out test set does address the *bias* of the training-set estimate: because the density $\hat{p}$ is fitted on training data and evaluated on independent test points, the optimistic bias from overfitting the histogram to the same samples is removed. However, the train/test split does not reduce the *variance* caused by small sample sizes.

As shown in Figure 3, the bootstrap standard deviation of $\widehat{\mathrm{KL}}_{\mathrm{test}}$ is comparable to—and in several cases larger than—that of $\widehat{\mathrm{KL}}_{\mathrm{train}}$ for small subgroups. This is expected: for a subgroup with n training members, the histogram density $\hat{p}$ is estimated from only n points regardless of the evaluation split, and the test subgroup is typically even smaller than the training subgroup, introducing additional Monte Carlo noise into the KL evaluation. In other words, the train/test split corrects *what* is being estimated (removing the positive bias from evaluating on the training data), but it cannot stabilize the estimate when the underlying density is poorly determined due to insufficient data.

C.3 Constrained Subgroup Discovery with support thresholds given as ratios

When the support thresholds are given as ratios, our method shows strong performance as well, as shown in Table 3.

Support thresholds: absolute vs. ratio. We report results under two conventions for the support threshold. In the main text (Table 1), we use absolute counts $\alpha \in \{50, 100, 150, 200, 300\}$, treating the threshold as a fixed minimum number of training instances regardless of dataset size. In the appendix here, we use ratios $\alpha' \in \{0.05, 0.10, 0.20, 0.30\}$, treating the threshold as a fraction of the training set. The two conventions probe different use cases: absolute counts correspond to an analyst requiring a minimum sample size for downstream analysis (e.g., a clinical-trial power requirement), while ratios correspond to an analyst requiring a minimum fractional coverage of the population. For a dataset with N training instances, an absolute threshold of α corresponds to a ratio of $\alpha' = \alpha/N$, and the two conventions therefore impose identical constraints at a per-dataset level when $\alpha = \alpha' N$. Because N varies across the 19 datasets by more than an order of magnitude (see Table 2), a fixed absolute count α maps to very different ratios across datasets, and vice versa; the two regimes are not directly interchangeable.

As Appendix C.2 suggests that subgroups with less than 50 training instances will lead to estimated KL-divergence with prohibitively large variance, we raise the threshold to $\max(\alpha' N, 50)$ training instances in this comparison; this matters for datasets with small sample sizes together with small α', whose rows then coincide with the $\alpha = 50$ rows of Table 1.

Results. Table 3 reports results under support thresholds expressed as ratios of the dataset size ($\alpha' \in \{0.05, 0.10, 0.20, 0.30\}$), complementing the absolute-count thresholds in the main text. The overall conclusion is unchanged: our method dominates, winning 62 of 76 (dataset, α') cells and achieving the best average rank at every threshold (1.42, 1.47, 1.11, 1.26 across the four ratios, versus 2.47–4.00 for all competitors).

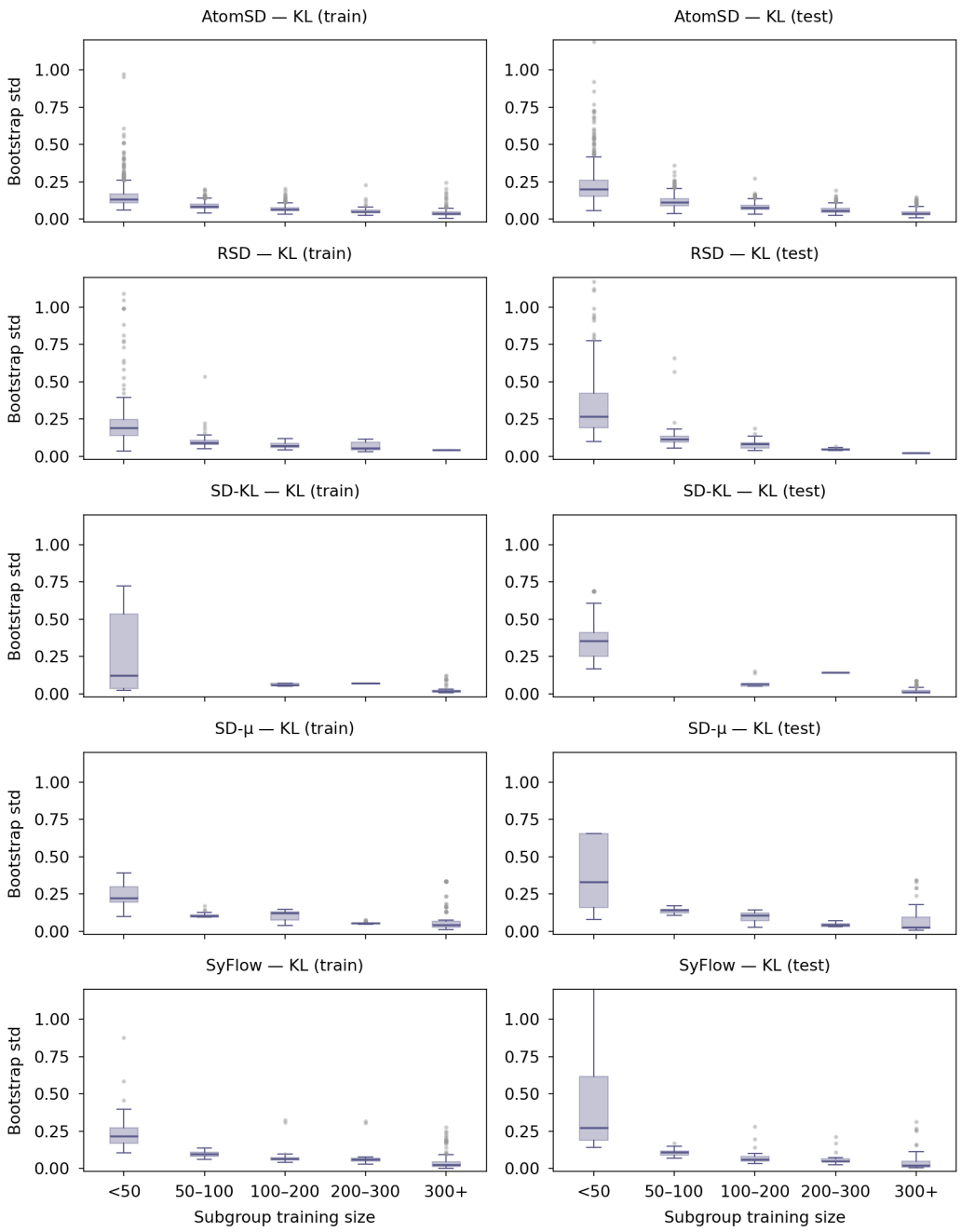

Figure 3: Bootstrap stability of KL divergence estimates across subgroup discovery algorithms. For each discovered subgroup, we perform 300 bootstrap replicates of the dataset and re-estimate $\widehat{\mathrm{KL}}$ using the same histogram-based procedure (Freedman–Diaconis binning with Bayesian smoothing). Each point in the box plot represents one subgroup's bootstrap standard deviation of $\widehat{\mathrm{KL}}$, grouped by training set size. **Left column:** KL estimated on the training split. **Right column:** KL estimated on the test split. Across all five algorithms, subgroups with fewer than 50 training instances exhibit substantially higher estimation variability, indicating that their KL-based ranking scores are unreliable. The pattern is consistent for both train and test estimates.

The gap widens as the support constraint increases. At $\alpha' = 0.20$ and $\alpha' = 0.30$, RSD, SD-KL and SD-μ collapse to the whole-dataset default (KL $= 0$) on the majority of datasets. Specifically, RSD

Table 3: Average held-out KL divergence (higher is better) across 3-fold CV for support ratio threshold α'. **Bold** indicates the best method per row. If a method returns no subgroup satisfying the support threshold on none of the three folds, KL is set to 0 (the default, trivial subgroup covering all instances); otherwise, the reported value is averaged only over folds on which the method returns a valid subgroup.

Dataset	α'	**AtomSD**	RSD	SD-KL	SD-μ	SyFlow	Dataset	α'	**AtomSD**	RSD	SD-KL	SD-μ	SyFlow
Ailerons	0.05	**1.491**	0.000	0.981	1.053	1.312	SkillCraft	0.05	**0.759**	0.000	-0.458	0.699	0.592
	0.10	**1.347**	0.000	0.981	1.053	1.312		0.10	**0.715**	0.000	0.000	0.699	0.592
	0.20	**1.069**	0.000	0.981	1.053	1.055		0.20	**0.723**	0.000	0.000	0.475	0.642
	0.30	1.051	0.000	0.000	1.053	**1.065**		0.30	**0.693**	0.000	0.000	0.000	0.551
Bikesharing	0.05	**2.180**	0.000	1.290	0.000	0.704	SML2010	0.05	**1.867**	0.000	0.000	1.180	1.063
	0.10	**1.557**	0.000	1.290	0.000	0.704		0.10	**1.384**	0.000	0.000	1.180	1.014
	0.20	**1.048**	0.000	0.000	0.000	0.704		0.20	**1.045**	0.000	0.000	0.000	0.639
	0.30	**0.757**	0.000	0.000	0.000	0.694		0.30	**0.728**	0.000	0.000	0.000	0.000
California	0.05	**1.645**	0.000	0.000	0.000	0.747	Supercond	0.05	**1.744**	0.000	0.839	0.773	0.665
	0.10	**1.292**	0.000	0.000	0.000	0.747		0.10	**1.278**	0.000	0.839	0.773	0.665
	0.20	**0.904**	0.000	0.000	0.000	0.444		0.20	**0.972**	0.000	0.839	0.773	0.665
	0.30	**0.635**	0.000	0.000	0.000	0.251		0.30	**0.798**	0.000	0.000	0.000	0.665
Concrete	0.05	**1.278**	1.174	0.000	1.220	1.213	Support2	0.05	**0.145**	0.000	-0.181	0.000	-0.163
	0.10	0.593	0.996	0.000	0.993	**1.213**		0.10	**0.055**	0.000	-0.181	0.000	-0.199
	0.20	**0.478**	0.000	0.000	0.000	0.186		0.20	**0.036**	0.000	-0.181	0.000	-0.110
	0.30	**0.382**	0.000	0.000	0.000	0.000		0.30	**0.038**	0.000	-0.181	0.000	-0.110
Elevators	0.05	0.318	0.000	-0.268	-0.100	**0.447**	Synchronous	0.05	1.375	**1.525**	0.000	1.204	0.965
	0.10	0.318	0.000	-0.240	-0.100	**0.447**		0.10	1.375	**1.525**	0.000	1.204	0.965
	0.20	**0.318**	0.000	0.000	-0.100	0.258		0.20	**1.157**	0.000	0.000	0.000	0.673
	0.30	**0.318**	0.000	0.000	-0.100	0.263		0.30	**0.824**	0.000	0.000	0.000	0.789
Energy	0.05	**1.783**	0.000	1.088	1.250	0.844	Thermography	0.05	**0.946**	0.554	0.000	0.000	0.451
	0.10	**1.544**	0.000	1.088	1.250	0.844		0.10	**0.700**	0.636	0.000	0.000	0.406
	0.20	**1.138**	0.000	1.056	0.000	0.844		0.20	**0.548**	0.349	0.000	0.000	0.000
	0.30	**0.909**	0.000	0.000	0.000	0.844		0.30	**0.319**	0.289	0.000	0.000	0.000
ForestFires	0.05	0.000	0.131	0.000	0.000	**0.579**	Toxicity	0.05	**0.897**	0.465	0.000	0.000	0.617
	0.10	0.000	0.131	0.000	0.000	**0.579**		0.10	**0.842**	0.621	0.000	0.000	0.617
	0.20	0.000	0.000	0.000	0.000	0.000		0.20	**0.498**	0.000	0.000	0.000	0.000
	0.30	0.000	0.000	0.000	0.000	0.000		0.30	**0.238**	0.000	0.000	0.000	0.000
House	0.05	**1.186**	0.000	0.000	0.000	0.447	Treasury	0.05	1.605	1.282	0.000	**1.923**	0.817
	0.10	**0.943**	0.000	0.000	0.000	0.447		0.10	**1.458**	0.000	0.000	0.000	0.817
	0.20	**0.710**	0.000	0.000	0.000	0.369		0.20	**1.258**	0.000	0.000	0.000	0.817
	0.30	**0.549**	0.000	0.000	0.000	0.000		0.30	**0.988**	0.000	0.000	0.000	0.817
NavalProp	0.05	**1.551**	0.000	0.000	0.120	0.003	Wizmir	0.05	1.516	**1.801**	1.629	1.072	1.016
	0.10	**1.344**	0.000	0.000	0.000	0.003		0.10	1.565	1.226	**1.629**	1.072	1.016
	0.20	**1.128**	0.000	0.000	0.000	0.003		0.20	**1.151**	0.000	0.000	1.072	1.016
	0.30	**0.871**	0.000	0.000	0.000	0.003		0.30	0.841	0.000	0.000	0.000	**0.920**
Puma32h	0.05	**2.365**	0.000	0.574	1.913	0.439							
	0.10	**1.783**	0.000	0.574	0.000	0.439							
	0.20	**1.132**	0.000	0.568	0.000	0.439							
	0.30	**0.760**	0.000	0.000	0.000	0.439							
#Best	0.05	**14**	2	0	1	2	Avg. Rank	0.05	**1.42**	3.53	4.00	3.05	3.00
	0.10	**14**	1	1	0	3		0.10	**1.47**	3.50	3.74	3.50	2.79
	0.20	**18**	0	0	0	0		0.20	**1.11**	3.92	3.68	3.68	2.61
	0.30	**16**	0	0	0	2		0.30	**1.26**	3.66	3.89	3.71	2.47
Total	–	**62**	3	1	1	7	*Overall*	–	**1.32**	3.65	3.83	3.49	2.72

returns no subgroup meeting the constraint on all but one dataset (Thermography) at $\alpha' = 0.20$ or 0.30, and SD-KL and SD-μ similarly degenerate on most large datasets. Our method, by contrast, returns a non-trivial subgroup on every dataset except ForestFires at every threshold, with KL-divergence values that remain substantial even at $\alpha' = 0.30$ (e.g., 0.871 on NavalProp, 0.798 on Supercond, 0.760 on Puma32h). This pattern mirrors the absolute-threshold results and reflects the same underlying property: the structure theorem guarantees that a Pareto-optimal subgroup exists at every support level, and our method locates it rather than falling back to the trivial solution.

C.4 Subgroup Improvement with support thresholds given as ratios

Figure 4 reports the support-constrained subgroup improvement experiment under support thresholds expressed as ratios, complementing the absolute-count thresholds in the main text. For each dataset and each competitor, we apply our method as a post-processing step to the competitor's returned subgroup and report the difference in held-out KL divergence; positive values indicate that our method yields a more deviating subgroup within the same atoms. Similarly, as Appendix C.2 suggests that subgroups with less than 50 training instances will lead to estimated KL-divergence with prohibitively

large variance, we skip subgroups for post-processing whose support (number of training instances) is smaller than 50.

The pattern is consistent with the main-text results: the distribution of KL differences is predominantly positive across all four competitors and all four support ratios. Medians are above zero for SyFlow, SD-KL and SD-μ at every ratio; for RSD, which returns a subgroup satisfying the constraint on only a few datasets beyond $\alpha' = 0.10$ (one or two per box), the median is around zero. This reinforces the interpretation advanced in the main text: 1) the structural condition identified by our theorem (union of whole atoms plus at most one fractional atom) is empirically necessary, and 2) existing methods routinely return subgroups violating it, leaving KL divergence on the table that our post-processing step reliably recovers.

The magnitude of improvement varies systematically with the competitor. Gains over SyFlow are largest at small α' and narrow as α' grows, while gains over RSD are small or absent. RSD often returns no subgroup at moderate-to-large ratios (see Table 3), so there is less structural slack to exploit when it does. The consistency of the effect across competitors is the key observation: the improvement is not an artifact of any particular baseline but a systematic consequence of the structural condition being violated in practice.

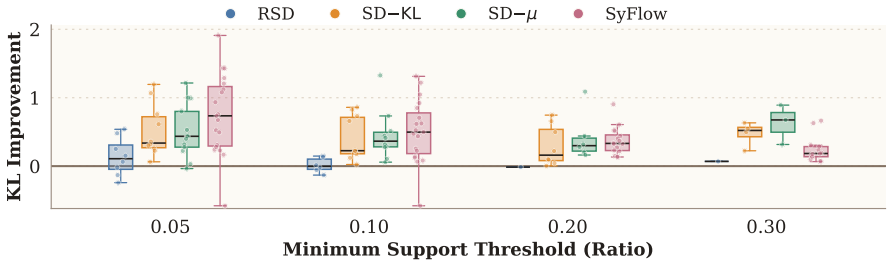

Figure 4: Held-out KL-divergence difference between AtomSD's refined subgroup and each competitor's original subgroup, across 19 datasets and four support ratios α'. Positive values indicate that AtomSD's refinement improves the given subgroups.

C.5 CD-diagrams and significance testing

Figure 5 shows critical-difference diagrams [9] corresponding to the constrained subgroup discovery results in Tables 1 and 3. At every support threshold in both regimes, the Friedman test (Iman–Davenport F statistic) rejects the null hypothesis of equal performance at $p < 10^{-5}$, and AtomSD is ranked first in every setting with a mean rank of 1.11–1.47. Its separation from the runner-up exceeds the critical difference in six of the nine settings; in the remaining three ($\alpha = 50$, $\alpha' = 0.10$, $\alpha' = 0.30$) the gap (1.19–1.32) falls slightly short of the critical difference of 1.40. The relative ordering of the remaining methods is less stable across thresholds (e.g., RSD is the second-best at $\alpha = 50$ but drops substantially at larger α), reflecting the baselines' sensitivity to the implicit support–deviation trade-off embedded in their respective quality measures. AtomSD's ranking, by contrast, is consistent across both absolute and ratio regimes, which we view as further evidence that the theory-directed atom-selection approach is not tuned to a particular support scale.

C.6 Empirically validating Theorem 3.3: searching over the leaves/atomic subgroups for unconstrained subgroup discovery

We empirically validate Theorem 3.3 as a sanity check: since Theorem 3.3 suggests that at the population level, the atomic subgroup with the largest KL-divergence is the most deviating subgroup, we empirically examine the KL-divergence of the atomic subgroups (i.e., the leaves) learned by CDTree. As our experiments in Appendix C.2 suggests that subgroups with less than 50 training instances will lead to estimated KL-divergence with prohibitively large variance, we exclude any atomic subgroups that has less than 50 training instances in this comparison. Accordingly, all subgroups with less than 50 training instances from competitor algorithms are also removed.

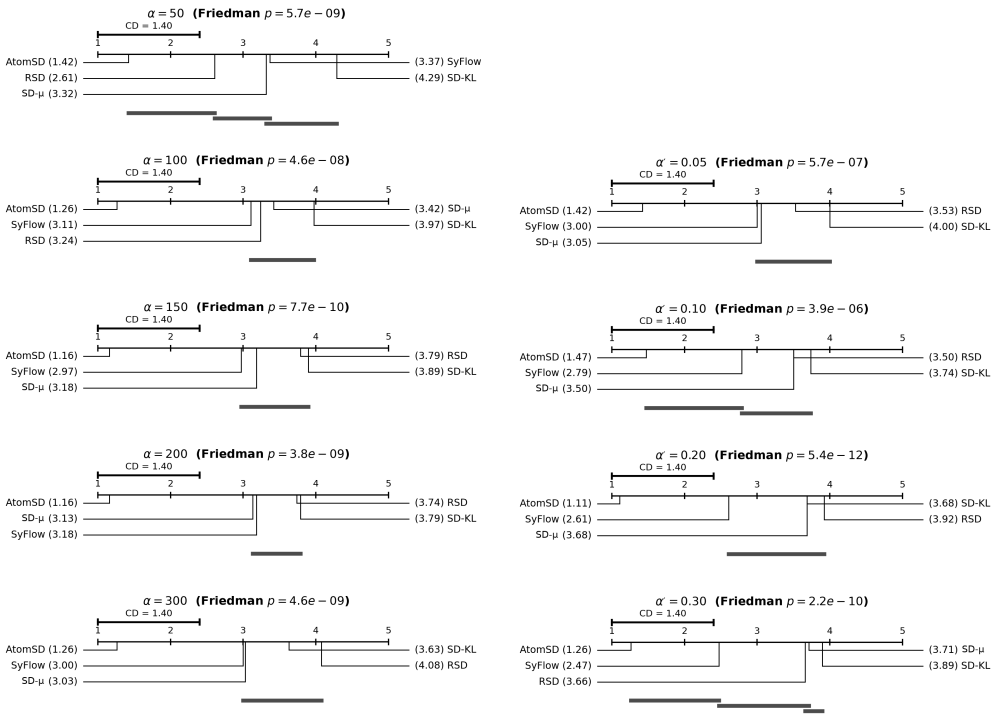

Figure 5: Critical-difference diagrams for constrained subgroup discovery across all support thresholds. **Left column:** absolute thresholds $\alpha \in \{50, 100, 150, 200, 300\}$ (Table 1). **Right column:** ratio thresholds $\alpha' \in \{0.05, 0.10, 0.20, 0.30\}$ (Table 3). Each diagram reports the Friedman test p-value (Iman–Davenport F statistic) and the Nemenyi critical difference (CD $= 1.40$ at significance level 0.05). Methods connected by a horizontal bar are not significantly different; methods separated by more than the CD are significantly different.

Table 4 shows that our method delivers the strongest overall performance on unconstrained subgroup discovery. (Here AtomSD returns a single leaf, so its values can be below the constrained ones in Table 1, where GAP may return unions of leaves.) It achieves the highest mean KL divergence on 10 of 19 datasets and obtains the best average rank of 1.789, ahead of all competitors. The margins are often substantial as well and it scales gracefully from low-dimensional problems (e.g., *Energy*, 4 features) to high-dimensional ones (e.g., *Supercond*, 81 features). RSD is the most consistent runner-up, winning on five datasets, but collapses on larger or harder instances such as *California*, *Elevators*, and *House*, where it returns a degenerate subgroup (KL ≤ 0) or none at all. SD-KL and SD-μ are competitive only on a handful of low-dimensional datasets and produce degenerate KL values elsewhere, while SyFlow's performance is uneven and rarely leading. Taken together, the results indicate that our approach finds more deviating subgroups than existing baselines and, crucially, does so reliably across a wide range of dataset sizes and dimensionalities.

D Additional Algorithmic Details

We provide further details of the proposed optimization algorithms.

SearchFull. Algorithm 2 specifies the SEARCHFULL subroutine, which refines the index set $\mathcal{I}$ of fully-included atoms while keeping the partial component p fixed. The procedure is a standard local search over the power set of candidate atoms: at each iteration it considers every single-atom toggle — adding an atomic subgroup S_i currently absent from $\mathcal{I}$, or removing one currently present — and commits to the best improving move, subject to the minimum-support constraint α. Candidates whose support overlaps with the current partial component p are excluded (line 4) to avoid double-counting

Table 4: Results for unconstrained subgroup discovery. Each cell reports the mean maximum KL divergence over three held-out folds (standard deviation in parentheses), measured between the discovered subgroup and the marginal distribution of the full held-out split. Per-dataset best means are bold. #Best counts datasets on which each algorithm attains the highest mean; Avg. Rank is the mean rank across datasets (lower is better).

Dataset	# rows	dim.	**AtomSD**	RSD	SD-KL	SD-μ	SyFlow
Ailerons	13750	40	**1.446** (±0.442)	1.363 (±0.168)	0.981 (±0.000)	1.053 (±0.005)	1.312 (±0.390)
Bikesharing	17379	17	**3.056** (±0.104)	0.891 (±0.300)	1.290 (±0.000)	2.889 (±0.229)	1.246 (±0.386)
California	20640	8	**1.962** (±0.209)	-0.694 (±0.000)	0.000 (±0.000)	1.316 (±0.108)	0.747 (±0.072)
Concrete	1030	8	1.147 (±0.163)	1.174 (±0.201)	0.000 (±0.000)	**1.220** (±0.224)	1.213 (±0.190)
Elevators	16599	18	**1.509** (±0.096)	0.000 (±0.000)	-0.268 (±0.244)	-0.100 (±0.063)	0.447 (±0.209)
Energy	9568	4	2.105 (±0.083)	**2.229** (±0.138)	1.088 (±0.035)	1.250 (±0.020)	0.844 (±0.067)
ForestFires	517	12	0.000 (±0.000)	0.131 (±0.000)	0.000 (±0.000)	0.000 (±0.000)	**0.579** (±0.000)
House	22784	16	**1.697** (±0.291)	0.000 (±0.000)	0.000 (±0.000)	1.501 (±0.107)	0.447 (±0.112)
NavalProp	11934	16	**1.687** (±0.093)	1.242 (±0.283)	0.878 (±0.542)	0.449 (±0.582)	0.952 (±0.358)
Puma32h	8192	32	2.769 (±0.230)	**3.088** (±0.292)	0.574 (±0.008)	1.913 (±0.068)	0.439 (±0.009)
SkillCraft	3338	18	0.387 (±0.362)	**1.035** (±0.000)	-0.458 (±0.647)	0.699 (±0.042)	0.592 (±0.170)
SML2010	2764	20	**2.152** (±0.314)	2.029 (±0.000)	0.000 (±0.000)	1.180 (±0.186)	1.253 (±0.432)
Supercond	21263	81	**3.094** (±0.183)	0.855 (±0.089)	0.839 (±0.017)	0.773 (±0.019)	0.665 (±0.028)
Support2	9105	26	**0.729** (±0.077)	0.119 (±0.513)	-0.181 (±0.005)	-0.175 (±0.137)	-0.163 (±0.070)
Synchronous	557	4	1.205 (±0.000)	**1.525** (±0.000)	0.000 (±0.000)	1.204 (±0.297)	0.965 (±0.453)
Thermography	1003	25	**0.911** (±0.350)	0.554 (±0.185)	0.000 (±0.000)	0.000 (±0.000)	0.451 (±0.219)
Toxicity	908	6	0.576 (±0.179)	0.465 (±0.201)	0.000 (±0.000)	0.000 (±0.000)	**0.617** (±0.372)
Treasury	1049	15	1.454 (±0.310)	1.282 (±0.000)	0.000 (±0.000)	**1.923** (±0.090)	0.817 (±0.121)
Wizmir	1461	9	1.656 (±0.122)	**1.801** (±0.204)	1.629 (±0.000)	1.072 (±0.088)	1.016 (±0.171)
#Best			**10**	5	0	2	2
Avg. Rank			**1.789**	2.395	4.342	3.211	3.263

instances across the full and partial parts of the subgroup. The loop terminates when no single-atom toggle improves the objective, yielding a locally optimal selection of whole atoms given the current p.

The move set is deliberately restricted to single-atom flips. Larger move sets (e.g., pairwise swaps) would enlarge the neighbourhood at the cost of per-iteration complexity, and we found empirically that the single-flip neighbourhood already yields competitive solutions when alternated with the partial-refinement step SEARCHPARTIAL. By Theorem 3.5, the globally optimal subgroup has the form $(\bigcup_{i\in\mathcal{I}} S_i) \cup S'_k$ for some index set $\mathcal{I}$ and at most one partial atom S'_k; SEARCHFULL targets the $\mathcal{I}$ component of this decomposition, and SEARCHPARTIAL targets the S'_k component. Alternating the two (Algorithm 1) therefore operates directly in the solution class guaranteed by the theorem.

Algorithm 2 SearchFull: Search for Unions of Full Atomic Subgroups

Input: Current index set $\mathcal{I} \subseteq [K]$ of whole atoms and partial subgroup p; candidate atoms $\mathcal{S}$, training data, minimum support α, SCORE and SUPP as in Algorithm 1.
Output: The index set $\mathcal{I}$ after a local search over single-atom toggles, with p unchanged.

function SEARCHFULL($\mathcal{I}, p$)
  repeat
    $\mathcal{I}_{\text{best}} \leftarrow \mathcal{I}$
    for all $i \in [K]$ such that $p \cap S_i = \varnothing$ **do** ▷ Consider toggling atom i
      $\mathcal{I}' \leftarrow \mathcal{I} \setminus \{i\}$ if $i \in \mathcal{I}$, otherwise $\mathcal{I} \cup \{i\}$
      if SUPP($\mathcal{I}', p$) $< \alpha$ **then continue** ▷ Support constraint violated
      end if
      if SCORE($\mathcal{I}', p$) $>$ SCORE($\mathcal{I}_{\text{best}}, p$) **then** $\mathcal{I}_{\text{best}} \leftarrow \mathcal{I}'$
      end if
    end for
    improved $\leftarrow$ SCORE($\mathcal{I}_{\text{best}}, p$) $>$ SCORE($\mathcal{I}, p$)
    if *improved* **then** $\mathcal{I} \leftarrow \mathcal{I}_{\text{best}}$ ▷ Commit the best improving toggle
    end if
  until $\neg$ *improved*
  return $(\mathcal{I}, p)$
end function

SearchPartial. Algorithm 3 specifies the SEARCHPARTIAL subroutine, which refines the partial component p while keeping the index set $\mathcal{I}$ fixed in structure. For each candidate atom S_i, the procedure uses S_i as the partial atom: i is removed from the index set if present (line 7) and the bounding box of S_i is shrunk into a partial region $p' \subseteq S_i$. The best resulting $(\mathcal{I}', p')$ across all candidate atoms is returned.

The core of the procedure is a best-improvement shrinkage of S_i's bounding box (lines 9–19). A *cut* tightens exactly one bound of one feature: it raises the lower bound, or lowers the upper bound, of that feature to a value observed within the current box. Features fixed by equality constraints in S_i are excluded. Starting from the full bounds $[\boldsymbol{\ell}, \boldsymbol{u}]$ of S_i, each iteration scores every admissible cut and commits the one that most improves score on the union with the atoms in $\mathcal{I}'$, subject to the support constraint. If no cut is ever applied, S_i is kept as a whole atom (line 21). The loop terminates when no cut improves the objective.

Two design choices deserve comment. First, restricting cuts to observed values rather than arbitrary thresholds keeps the candidate pool finite and ensures every partial region corresponds to a valid rule on the training data. Second, cuts are chosen by best improvement rather than first improvement: all candidate cuts across all features are scored before the box changes, so the result does not depend on the order in which features are visited. The search stops at a local optimum with respect to such cuts and does not guarantee a globally optimal partial region. The output $p' = [\boldsymbol{\ell}, \boldsymbol{u}]$ is a rectangular partial region within S_i, exactly the form of S'_k in the decomposition guaranteed by Theorem 3.5.

Algorithm 3 SearchPartial: Search for the Optional Partial Atomic Subgroup

Input: Current index set $\mathcal{I} \subseteq [K]$ of whole atoms and partial subgroup p; candidate atoms $\mathcal{S}$, training data, minimum support α, SCORE and SUPP as in Algorithm 1; BOX(S_i): the bounding box $[\boldsymbol{\ell}, \boldsymbol{u}]$ of S_i over its non-equality-constrained features; CONVERT$(\boldsymbol{\ell}, \boldsymbol{u})$: the rule (sub-region of S_i) with these bounds.

Output: The best-scoring pair among the current $(\mathcal{I}, p)$, the pruned pair $(\mathcal{I}, \varnothing)$, and one candidate per atom i: either $(\mathcal{I} \setminus \{i\},\ p')$ with a shrunk $p' \subsetneq S_i$, or $(\mathcal{I} \cup \{i\}, \varnothing)$ if no cut is applied; all subject to the minimum support α.

1: **function** SEARCHPARTIAL($\mathcal{I}, p$)
2: $(\mathcal{I}_{\text{best}}, p_{\text{best}}) \leftarrow (\mathcal{I}, p)$
3: **if** SUPP$(\mathcal{I}, \varnothing) \geq \alpha$ **and** SCORE$(\mathcal{I}, \varnothing) >$ SCORE$(\mathcal{I}, p)$ **then**
4: $(\mathcal{I}_{\text{best}}, p_{\text{best}}) \leftarrow (\mathcal{I}, \varnothing)$ ▷ Dropping p is better
5: **end if**
6: **for all** $i \in [K]$ **do** ▷ Try S_i as the partial atom
7: $\mathcal{I}' \leftarrow \mathcal{I} \setminus \{i\}$ ▷ Unchanged if $i \notin \mathcal{I}$
8: $[\boldsymbol{\ell}, \boldsymbol{u}] \leftarrow$ BOX(S_i) ▷ Start from the full extent of S_i
9: **repeat** ▷ Best-improvement shrinkage: each iteration tightens one bound of one feature
10: $p' \leftarrow$ CONVERT$(\boldsymbol{\ell}, \boldsymbol{u})$; $c_{\text{best}} \leftarrow \varnothing$; $s_{\text{best}} \leftarrow$ SCORE$(\mathcal{I}', p')$
11: **for all** cuts $c \in \{\ell_f \leftarrow v,\ u_f \leftarrow v\}$, v a value of feature f observed in p' **do**
12: $p_c \leftarrow$ CONVERT($[\boldsymbol{\ell}, \boldsymbol{u}]$ with cut c applied)
13: **if** SUPP$(\mathcal{I}', p_c) \geq \alpha$ **and** SCORE$(\mathcal{I}', p_c) > s_{\text{best}}$ **then**
14: $(c_{\text{best}}, s_{\text{best}}) \leftarrow (c,$ SCORE$(\mathcal{I}', p_c))$
15: **end if**
16: **end for**
17: **if** $c_{\text{best}} \neq \varnothing$ **then** apply c_{best} to $[\boldsymbol{\ell}, \boldsymbol{u}]$
18: **end if**
19: **until** $c_{\text{best}} = \varnothing$
20: $p' \leftarrow$ CONVERT$(\boldsymbol{\ell}, \boldsymbol{u})$
21: **if** $p' = S_i$ **then** $(\mathcal{I}', p') \leftarrow (\mathcal{I}' \cup \{i\}, \varnothing)$ ▷ No cut applied: keep S_i as a whole atom
22: **end if**
23: **if** SUPP$(\mathcal{I}', p') \geq \alpha$ **and** SCORE$(\mathcal{I}', p') >$ SCORE$(\mathcal{I}_{\text{best}}, p_{\text{best}})$ **then**
24: $(\mathcal{I}_{\text{best}}, p_{\text{best}}) \leftarrow (\mathcal{I}', p')$
25: **end if**
26: **end for**
27: **return** $(\mathcal{I}_{\text{best}}, p_{\text{best}})$
28: **end function**

E NP-hardness of finding the global-optimal union of atomic subgroups

Theorem E.1 (Worst-case NP-hardness)**.** *Let $\{S_i\}_{i=1}^m$ partition $\mathcal{X}$. Given a threshold $\alpha \in (0,1)$, the problem*

$$\max_{I\subseteq[m]} \mathrm{KL}\Big(P(Y \mid X \in \bigcup_{i\in I} S_i) \,\Big\|\, P(Y)\Big) \quad \textit{subject to} \quad P_X\big(\bigcup_{i\in I} S_i\big) \geq \alpha \tag{4}$$

is NP-hard in general. This hardness holds even when Y is continuous and each $P(Y \mid X \in S_i)$ is a uniform distribution on an interval disjoint from all others.

Proof. We prove the claim by a polynomial-time reduction from SUBSET-SUM. The key idea is to construct an instance where the KL divergence of any union of atoms depends only on its total weight, so that maximizing KL subject to a support constraint is equivalent to finding a subset of atoms with a prescribed total weight, which is precisely SUBSET-SUM.

Step 1: Recall the source problem. An instance of SUBSET-SUM consists of positive integers $a_1, \ldots, a_m$ and a target $T \in \mathbb{N}$, and asks whether there exists $I \subseteq [m]$ such that $\sum_{i\in I} a_i = T$. This problem is NP-complete.

Step 2: Construct the reduction. Given a SUBSET-SUM instance $(a_1, \ldots, a_m, T)$, let $A := \sum_{i=1}^m a_i$ and set $w_i := a_i/A$ and $\alpha := T/A$. Let X be uniform on $[0,1]$ and construct a partition $\{S_i\}_{i=1}^m$ of $[0,1]$ into consecutive intervals with $P_X(S_i) = w_i$, and assign conditional distributions $P(Y \mid X \in S_i) = U([i, i+1])$. These uniform distributions have pairwise disjoint supports by construction, which will be the key property enabling the KL computation below.

Step 3: Compute the KL divergence. Fix any $I \subseteq [m]$, write $A_I := \bigcup_{i\in I} S_i$, and let $W_I := P_X(A_I) = \sum_{i\in I} w_i$. By the law of total probability, $P(Y \mid X \in A_I) = \sum_{i\in I}(w_i/W_I)\, U([i, i+1])$ and $P(Y) = \sum_{i=1}^m w_i\, U([i, i+1])$. Since the supports $[i, i+1]$ are pairwise disjoint, for $y \in [i, i+1]$ with $i \in I$ the densities simplify to $p_I(y) = w_i/W_I$ and $p(y) = w_i$, so the likelihood ratio $p_I(y)/p(y) = 1/W_I$ is constant across the entire support of p_I. This gives the clean expression

$$\mathrm{KL}(P(Y \mid X \in A_I) \,\|\, P(Y)) = \log \frac{1}{W_I}.$$

The KL divergence therefore depends on I only through its total weight W_I and, more importantly, not through which specific atoms are selected.

Step 4: Reduce to SUBSET-SUM. Since $w \mapsto \log(1/w)$ is strictly decreasing, the optimization problem (4) reduces to finding I with $W_I \geq \alpha$ that minimizes W_I. The decision version with threshold $\tau := \log(1/\alpha)$ therefore asks whether there exists I with $W_I \geq \alpha$ and $\log(1/W_I) \geq \log(1/\alpha)$, i.e., $W_I \leq \alpha$. Together these two conditions force $W_I = \alpha$ exactly. Translating back via $w_i = a_i/A$ and $\alpha = T/A$, we have

$$W_I = \alpha \quad \Longleftrightarrow \quad \sum_{i\in I} a_i = T,$$

which is precisely the SUBSET-SUM decision problem. The reduction runs in polynomial time, so the decision version of (4) is NP-hard, and hence so is the optimization problem itself. Since the construction uses only continuous Y with uniform conditional distributions on pairwise disjoint intervals, the hardness holds already for this restricted class of instances. □

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 three contributions: (i) a structural characterization of KL-optimal subgroups under support constraints (Theorem 3.5, Corollary 3.6); (ii) a modular two-step recipe reducing the search to a finite combinatorial problem; and (iii) an instantiation (AtomSD) that empirically dominates existing methods across 19 UCI datasets and improves subgroups returned by competing methods as a post-processing step. Each claim is supported by the corresponding theorem in Section 3 or experiment in Section 5. The scope of the theoretical results is stated explicitly (partition-based generative model; distinct generating distributions for the necessity direction), and the introduction flags that real data deviates from this model to varying degrees.

   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 contains a dedicated "Limitations and future work" paragraph that addresses (i) the partition-based generative model is an approximation of real data and formal robustness to distributional misspecification remains open, and (ii) the natural extension to subgroup-set discovery with diversity considerations. In addition, the experiments section explicitly discusses the finite-sample regime where the partition is learned rather than given (Section 5).

   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 theoretical results (Theorem 3.1, Theorem 3.3, Theorem 3.5, Corollary 3.6, and Theorem E.1) are numbered, cross-referenced, and stated with their full set of assumptions. The partition-based generative model is formally introduced in Definition 3.2, and results requiring additional assumptions (a unique maximizing atom in Theorem 3.3; distinct generating distributions in Corollary 3.6) state them explicitly. Complete proofs for all theorems are provided in Appendix A.

   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 5 specifies the datasets (19 benchmark datasets), competitor algorithms, evaluation protocol (3-fold cross-validation with held-out KL estimation), and dataset preprocessing. Full reproducibility details including hyperparameter choices, KL estimation procedure, and dataset summary are in Appendix B. Pseudocode for all algorithmic components (GAP, SearchFull, SearchPartial) is given in the main text (Algorithm 1) and Appendix D (Algorithms 2 and 3). Source code is provided at an anonymous URL.

   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: Source code is available at `https://anonymous.4open.science/r/SubgroupDiscovery`. All datasets are publicly available, mostly from the UCI Machine Learning Repository [1].

   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: Data splits (3-fold cross-validation), hyperparameter settings (default values for all competitor methods; no hyperparameter tuning for AtomSD), evaluation metric (held-out KL divergence via histogram density estimation with Freedman–Diaconis

binning and Bayesian smoothing), and support threshold ranges (absolute counts $\alpha \in \{50, 100, 150, 200, 300\}$ in the main text; ratios $\alpha' \in \{0.05, 0.10, 0.20, 0.30\}$ in Appendix C.3) are all specified.

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: Table 1 reports mean KL divergence across 3-fold cross-validation. A Friedman test is applied to test the null hypothesis of equal performance across methods, with $p < 10^{-5}$ at every support threshold; the corresponding Nemenyi critical-difference diagrams are provided in Appendix C.5. Additionally, Appendix C.2 reports a bootstrap stability analysis ($B = 300$ replicates) characterizing the estimation variability of KL divergence for subgroups of varying sizes.

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 B states that all experiments were conducted on a cluster equipped with AMD EPYC 7662 64-Core Processors. The method is CPU-only and does not require GPU resources.

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 only publicly available benchmark datasets, does not involve human subjects, and does not release models or data with misuse risk.

   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: [N/A]

    Justification: This is a foundational theoretical/methodological paper on subgroup discovery. The theoretical contribution (structural characterization of KL-optimal subgroups under support constraints) and algorithmic instantiation are domain-agnostic tools for exploratory data analysis. We do not anticipate direct positive or negative societal impact beyond the general benefits and risks of data-analysis tools, which are not specific to this work.

    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 paper does not release data or models that pose a high risk for misuse. The released code is an implementation of a domain-agnostic subgroup discovery method applied to publicly available benchmarks.

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: All datasets used are publicly available, mostly from the UCI Machine Learning Repository [1], and are cited accordingly. Competitor methods (SyFlow [34], RSD [29], SD-μ and SD-KL via pysubgroup [24], CDTree [35]) are used with their original implementations and cited at each point of use.

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: The AtomSD implementation is released at the anonymous repository `https://anonymous.4open.science/r/SubgroupDiscovery`, with documentation including usage instructions and reproduction scripts for the experiments reported in the paper.

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: The paper does not involve crowdsourcing or research with human subjects.

    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: The paper does not involve research with human subjects.

    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: All theoretical results, algorithmic design, experimental protocols, and scientific interpretations were developed entirely by the authors. LLMs did not contribute to the core methodology, scientific rigor, or originality of the research.

    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.