Shallow decision trees for explainable $k$-means clustering

Eduardo Laber$^a$, Lucas Murtinho$^{a,*}$, Felipe Oliveira$^b$

$^a$PUC-Rio – Department of Informatics, Rio de Janeiro, RJ, Brazil
$^b$PUC-Rio – Department of Mathematics, Rio de Janeiro, RJ, Brazil

Abstract

A number of recent works have employed decision trees for the construction of explainable partitions that aim to minimize the $k$-means cost function. These works, however, largely ignore metrics related to the depths of the leaves in the resulting tree, which is perhaps surprising considering how the explainability of a decision tree depends on these depths. To fill this gap in the literature, we propose an efficient algorithm that takes into account these metrics. In experiments on 16 datasets, our algorithm yields better results than decision-tree clustering algorithms recently presented in the literature, typically achieving lower or equivalent costs with considerably shallower trees.

1. Introduction

As machine learning models have become used in a wide range of fields, the topic of explainability has grown in importance. Understanding the reasoning behind a model’s decision may be crucial to increase user confidence; to satisfy legal requirements; to conform to moral and ethical expectations; and to verify the model’s work. Since more complex models tend to be harder to interpret but are also more capable of returning good results, there is a trade-off between model performance and explainability. The challenge of navigating this trade-off is increasingly being explored in the machine learning literature.

Although initial efforts towards explainability focused on supervised learning models [1], a number of studies on explainable unsupervised models, and

*Corresponding author.

Email address: lucas.murtinho@gmail.com (Lucas Murtinho)
clustering models in particular, have appeared more recently. One idea that has earned some attention in the literature is to partition the data based on axis-aligned cuts, which can be induced by binary decision trees: at each node $u$ of the tree, a value $\theta$ and a dimension $i$ are selected, so that all data points that have reached $u$ go to one of its two children according to whether their values for dimension $i$ are smaller than $\theta$ or not. In this kind of approach, usually, each cluster is associated with a leaf.

Decision trees are widely considered to be explainable models by machine learning standards. However, the explainability of a decision tree greatly depends on the depths of its leaves, as empirically demonstrated by [2] in a study on how tree structure parameters (the number of leaves, branching factor, tree depth) influence the tree interpretability. The conclusion, based on empirical data from a survey with 98 questions answered by 69 respondents, is that the question depth (the depth of the deepest leaf that is required when answering a question about a classification tree) turns out to be the most important parameter. Explaining leaves that are far from the root involves many tests, which makes it harder to grasp the model’s logic.

There are many possible metrics that can be associated with the depths of the leaves, such as the maximum depth and the average depth. Here, we focus on metrics that consider as equally important the explanation of each data point. More specifically, we consider the Weighted Average Depth (WAD) and the Weighted Average Explanation Size (WAES). The former weighs the depth of each leaf by the number of points of its associated cluster; to minimize it, large clusters shall be associated with shallower leaves (shorter explanations). The latter is a variation of WAD that replaces the depth of a leaf by the number of non-redundant tests in the path from the root to the leaf. These measures are formalized and discussed more thoroughly in Section 1.1.

Figures 1 and 2 show two decision trees that partition the Avila dataset [3] into 12 clusters. Both trees induce the same partition; however, the tree from Figure 1, produced by ExGreedy [4], has WAES $\approx 5.4$, while the one from Figure 2 has WAES $\approx 3.7$, which represents a “gain” of almost 2 conditions on average. For WAD, the gain is even larger, of over 2 conditions ($\approx 6.2$ vs. $\approx 3.8$). This example suggests that there is significant room to improve the explainability of the partitions provided by algorithms available in the literature.

Our contributions. As in [5, 6], we investigate the problem of building
explainable clustering via decision trees. The main difference of our work with respect to the previous ones is our focus on building decision trees that simultaneously yield short explanations and induce partitions of good quality in terms of the \( k \)-means cost function. We understand that one contribution of our paper is the observation that previous approaches overlook the quite important aspect of minimizing measures related to the tree’s depth.

In Section 3 we present a strategy that builds decision trees that induce partitions of low \( k \)-means cost and have low values for WAES and WAD. As other proposals in the literature, we start from the partition provided by some algorithm for the (non-explainable) \( k \)-means clustering problem and
Figure 2: A tree from the ExSHALLOW algorithm, proposed in this paper, for the Avila dataset. It induces the same partition as the tree presented in Figure 1.

build the tree in a top-down fashion, by selecting at each node a cut that is “good” in terms of minimizing our metrics. The key novelties we present here are an effective and efficient way to evaluate the potential of a cut in terms of minimizing the $WAD/WAES$ and how to efficiently trade-off the (potentially conflicting) goals of minimizing both these metrics and the cost of the induced partition.

To evaluate our strategy, in Section 4 we compare its performance against recently proposed algorithms over 16 datasets. Our strategy generated partitions as good as the best of its competitors in terms of the $k$-means cost, while being significantly better in terms of the aforementioned explainability measures. It also compares to the best of these competitors in terms of explainability, while inducing much better partitions than this competitor in terms of the $k$-means cost. Moreover, these gains were obtained without compromising computational efficiency.

1.1. Preliminaries and problem definition

Let $\mathcal{X}$ be a collection of $n$ data points in $\mathbb{R}^d$ and $k \geq 2$ be an integer. In (cost-oriented) hard clustering problems, we want to find a partition of $\mathcal{X}$ that minimizes a given cost function. In the widely studied $k$-means clustering problem, the cost of a partition $\mathcal{P} = \{C_1, \ldots, C_k\}$ is the sum of the squared Euclidean distances between all points in $\mathcal{X}$ and the representatives of the
clusters to which they belong:

\[
\text{cost}(\mathcal{P}) = \sum_{i=1}^{k} \sum_{x \in C_i} ||x - c_i||^2.
\] (1)

In this case, the representative \(c_i\) of cluster \(C_i\) is given by the mean of its points, \(c_i = \frac{\sum_{x \in C_i} x}{|C_i|}\).

In our study we are interested in partitions induced by axis-aligned binary decision trees. A decision tree is axis-aligned if each internal node \(v\) is associated with a test (cut), specified by a coordinate \(i_v \in [d]\) and a real value \(\theta_v\), that partitions the points in \(X\) that reach \(v\) into two sets: those having the coordinate \(i_v\) smaller than or equal to \(\theta_v\) and those having it larger than \(\theta_v\). The leaves induce a partition of \(\mathbb{R}^d\) into axis-aligned regions and, naturally, a partition of \(\mathcal{X}\) into clusters.

For our purposes, it will be convenient to associate a condition to each edge of the tree: the left edge leaving a node \(v\) is associated with the condition \(x_{i_v} \leq \theta_v\) and the right one with the condition \(x_{i_v} > \theta_v\). The explanation of a cluster \(C\) in a decision tree \(D\) is given by the logical AND of the conditions associated with the edges in the path from the root of \(D\) to the leaf associated with \(C\). We say that a condition is redundant with respect to cluster \(C\) if its removal does not change the explanation for \(C\). As an example, if the explanation of cluster \(C\) is \(x_1 > 30\) AND \(x_2 \leq 20\) AND \(x_1 > 70\), then the condition \(x_1 > 30\) is redundant.

We consider two explainability measures for our study: the Weighted Average Explanation Size (WAES) and the Weighted Average Depth (WAD). For a partition \(\mathcal{P} = (C_1, \ldots, C_k)\) induced by a binary decision tree \(D\) with \(k\) leaves, where the cluster \(C_i\) is associated with the leaf \(i\), we have

\[
\text{WAD}(D) = \frac{\sum_{i=1}^{k} |C_i| \ell_i}{n}
\] (2)

and

\[
\text{WAES}(D) = \frac{\sum_{i=1}^{k} |C_i| \ell_{\text{nr}}^i}{n},
\] (3)

where \(\ell_i\) and \(\ell_{\text{nr}}^i\) are, respectively, the number of conditions and non-redundant conditions (w.r.t. \(C_i\)) in the path from the root to leaf \(i\). The WAD is a very natural metric and its relevance was advocated in [2]. The WAES, to the best of our knowledge, has not been considered before.
In terms of explainability, a decision tree is a single structure that allows us to visualize explanations for all clusters (some of them potentially having redundant conditions), and WAD gives the average length (weighted by the cluster’s sizes) of these explanations. For each specific cluster, however, we may derive more compact explanations by removing redundant conditions, and WAES measures the average size of these explanations, again weighted by the cluster’s sizes.

The problem proposed in [5] is that of finding the partition that minimizes (1), among those that can be induced by a decision tree of k leaves. In addition to minimizing (1), we also focus on building trees with low values for WAD (2) and WAES (3).

To accomplish our goal, we note that it is important to take into account both WAD and WAES during the decision tree construction, since the optimization of one metric does not imply on the optimization of the other. For instance, let $X_i$ be the set of $2^{2i} - 1$ points in $R^k$, where the $j$th point has all its $k$ components equal to $2^{2i} + j$. Let $\mathcal{X} = X_1 \cup \ldots \cup X_k$. Clearly the optimal unrestricted $k$-partition for $\mathcal{X}$ is $(X_1, \ldots, X_k)$. This partition can be induced by many decision trees as the trees $D_1$-$D_4$, described below, that have only one internal node per level:

- $D_1$: the cut at level $i$ is $(i, 2^{2i+1})$ so that both $\text{WAD}(D_1)$ and $\text{WAES}(D_1)$ are $\approx k$;
- $D_2$: the cut at level $i$ is $(1, 2^{2(k-i)+1})$ so that both $\text{WAD}(D_2)$ and $\text{WAES}(D_2)$ are $O(1)$;
- $D_3$: the cut at level $i$ is $(i, 2^{2(k-i)+1})$ so that $\text{WAD}(D_3)$ is $O(1)$ and $\text{WAES}(D_3)$ is $\approx k$
- $D_4$: the cut at level $i$ is $(1, 2^{2i+1})$ so that $\text{WAD}(D_4)$ is $\approx k$ and $\text{WAES}(D_4)$ is $O(1)$

Therefore, we shall consider both WAES and WAD while building the tree, otherwise we can end up with a tree that performs poorly with respect to one of the metrics.

We conclude this section by introducing terminologies and notations that will be useful throughout this paper. We use the term explainable clustering to refer to a clustering that is induced by some axis-aligned decision tree. By an $i$-cut we mean a cut associated with component $i$, that is, a cut $x_i \leq \theta$, where...
for some real value $\theta$. If a node in a decision tree is associated with an $i$-cut we say that it is an $i$-node.

2. Related work

[5] presents a poly-time algorithm, IMM, that receives a (non-explainable) partition $P_u$ to the $k$-means clustering problem and builds a decision tree, in top-down fashion, by selecting at each node the cut that, among those that separate at least two representatives in $P_u$, minimizes the number of data points separated from their representatives in $P_u$. In addition, they prove that the cost of the resulting partition is $O(k^2 \text{cost}(P_u))$. A consequence of this result is that the price of explainability, measured by the ratio between the cost of an optimal explainable partition and that of an optimal (non-explainable) one, is $O(k^2)$.

After [5], new algorithms, yielding to improved bounds on the price of explainability, were proposed [4, 7, 8, 9, 10]. The best known upper bound, among those that only depend on $k$, is $O(k \log k)$ from [9]. We note that this bound is nearly tight since the same paper also provides an $\Omega(k)$ lower bound. [11] shows that the $k$-means explainable clustering problem is hard to approximate, thus consolidating the motivation for exploring heuristics for this problem.

Empirical studies with algorithms for building explainable partitions can be found in [6, 4]. The former proposes the ExKMC algorithm and compares it with IMM, CART [12], KDTREE [13], CUBT [14], and CLTREE [15]. One conclusion that can be drawn from this study is that IMM and ExKMC outperform the other competitors when the objective is building trees with exactly $k$ leaves. ExKMC, in contrast to IMM, is not limited to building trees with $k$ leaves, allowing partitions where the same cluster is associated with more than one leaf. This flexibility allows partitions with lower costs (though less explainable). An algorithm with provable guarantees for this scenario was recently obtained in [16].

[4] introduces a simple greedy algorithm, ExGREEDY, and shows that it produces partitions with lower costs than those produced by IMM. We note that neither [6] nor [4] analyze the produced trees in terms of their explainability. In our experiments we compare IMM, ExGREEDY, and ExKMC against our method using different measures of explainability.

The aforementioned papers focus on the $k$-means clustering problem. However, a number of papers [14, 17, 18] propose decision-tree algorithms
to build partitions that optimize other measures.

The importance of building shallow trees for achieving interpretability has been previously discussed in [19], in which clustering and decision trees (constructed with the CART algorithm [12]) are used to locally interpret the results of a black-box model.

3. A strategy for building shallow trees with low cost

Our strategy, denoted by ExSHALLOW, builds a decision tree in a top-down fashion as shown in Algorithm 1. As an input the strategy receives a set of points $X'$ and also a set $S'$ of $k$ representatives (denoted here by reference centers). We say that two cuts are equivalent with respect to set $X' \cup S'$ if they are associated with the same component (both are $i$-cuts for some $i$) and if they induce the same binary partition on $X' \cup S'$. Note that there are at most $|X' \cup S'|d$ pairwise non-equivalent cuts. At each node the strategy evaluates

$$\text{Price}(\gamma, X', S') + \lambda \cdot \text{DExp}(\gamma, X', S')$$

(4)

for each cut $\gamma$ in the set of non-equivalent cuts that separate at least two reference centers from $S'$. Then, it selects the cut $\gamma^*$ for which (4) is minimum.

In Equation (4), $\text{Price}(\gamma, X', S')$ and $\text{DExp}(\gamma, X', S')$ (both detailed further below) estimate how good $\gamma$ is for the goal of building a partition with low cost and with low values for WAES/WAD, respectively; $\lambda$ is a trade-off parameter that we discuss in Section 3.2. We note that DExp stands for Depth Explainability.

After selecting $\gamma^*$, the strategy is recursively performed for each of the groups of the binary partition induced by $\gamma^*$. The recursion stops when $S'$ contains only one reference center. The initial set of reference centers can be built by any algorithm for the (non-explainable) $k$-means clustering problem, such as Lloyd’s algorithm [20].
Algorithm 1 ExShallow($\mathcal{X}'$, $\mathcal{S}'$)

$\mathcal{X}'$: set of points; $\mathcal{S}'$: set of reference centers

1: if $|\mathcal{S}'| = 1$ then
2: Return $\mathcal{X}'$ and the single reference center in $\mathcal{S}'$
3: else
4: $\mathcal{C} \leftarrow$ set of non-equivalent cuts w.r.t. $\mathcal{X}' \cup \mathcal{S}'$ that separate at least two centers in $\mathcal{S}'$
5: $\gamma^* \leftarrow \arg \min_{\gamma \in \mathcal{C}} \{ \text{Price}(\gamma, \mathcal{X}', \mathcal{S}') + \lambda \cdot \text{DExp}(\gamma, \mathcal{X}', \mathcal{S}') \}$
6: $(\mathcal{X}_L^*, \mathcal{X}_R^*) \leftarrow$ partition of $\mathcal{X}'$ induced by $\gamma^*$
7: $(\mathcal{S}_L^*, \mathcal{S}_R^*) \leftarrow$ partition of $\mathcal{S}'$ induced by $\gamma^*$
8: Create a node $u$
9: $u.\text{LeftChild} \leftarrow \text{ExShallow}(\mathcal{X}_L^*, \mathcal{S}_L^*)$
10: $u.\text{RightChild} \leftarrow \text{ExShallow}(\mathcal{X}_R^*, \mathcal{S}_R^*)$
11: Return the tree rooted at $u$
12: end if

Let $\mathcal{X}'$ and $\mathcal{S}'$ be, respectively, the sets of points and centers that reach some given node in the decision tree. In addition, let $\gamma$ be a cut that splits $\mathcal{X}'$ into groups $\mathcal{X}_L^*$ and $\mathcal{X}_R^*$ and splits $\mathcal{S}'$ into groups $\mathcal{S}_L^*$ and $\mathcal{S}_R^*$, each of them containing at least one reference center. $\text{Price}(\gamma, \mathcal{X}', \mathcal{S}')$ is defined as

$$\text{Price}(\gamma, \mathcal{X}', \mathcal{S}') := \frac{\text{InducedCost}(\gamma, \mathcal{X}', \mathcal{S}')}{{\text{CurrentCost}}(\mathcal{X}', \mathcal{S}')}, \quad (5)$$

where

$$\text{CurrentCost}(\mathcal{X}', \mathcal{S}') := \sum_{x \in \mathcal{X}'} \min_{c \in \mathcal{S}'} ||x - c||_2^2 \quad (6)$$

and

$$\text{InducedCost}(\gamma, \mathcal{X}', \mathcal{S}') := \left( \sum_{x \in \mathcal{X}_L^*} \min_{c \in \mathcal{S}_L^*} ||x - c||_2^2 + \sum_{x \in \mathcal{X}_R^*} \min_{c \in \mathcal{S}_R^*} ||x - c||_2^2 \right); \quad (7)$$

that is, $\text{CurrentCost}$ and $\text{InducedCost}$ give, respectively, the cost of the partition before and after applying cut $\gamma$. In both cases, each point is associated with the closest valid reference center. We note that $\text{InducedCost}$ is the cost function used by the ExGreedy algorithm proposed in [4] to select a cut at each node.
To obtain $D_{\text{Exp}}(\gamma, \mathcal{X}', \mathcal{S}')$, we first calculate $\hat{\text{WAD}}(\gamma, \mathcal{X}', \mathcal{S}')$, an estimation of the quality of $\gamma$ for finding a good tree in terms of WAD, and then we adjust $\hat{\text{WAD}}(\gamma, \mathcal{X}', \mathcal{S}')$ to take into account the WAES.

Estimating whether a cut is good or not in terms of WAD is a non-obvious task. For other metrics, such as the maximum depth of a tree, this is much simpler: the more balanced the cut, the better it is. To estimate the quality of the cut $\gamma$ for our task, we efficiently compute (2) for an auxiliary tree that is built specifically for this purpose.

More precisely, $\hat{\text{WAD}}(\gamma, \mathcal{X}', \mathcal{S}')$ is given by the return of the procedure presented in Algorithm 2. EvalWAD($N, K, r_p, r_c$) returns the WAD of a tree with $K$ leaves (corresponding to centers) for a set of $N$ points, where each node in the tree splits the points and the centers in the same proportion as $\gamma$ does, that is, proportionally to $r_p = |\mathcal{X}'_L|/|\mathcal{X}'|$ and $r_c = |\mathcal{S}'_L|/|\mathcal{S}'|$, respectively. We note that these ratios do not change along the algorithm execution and that the resulting decision tree is just a theoretical tree (which may not even be feasible for the instance under consideration), built to estimate how good the cut $\gamma$ is for the goal of minimizing Equation (2).

Algorithm 2 EvalWAD($N, K, r_p, r_c$)

$N$: Current number of points; $K$: Current number of reference centers; $r_p$: Point-split ratio; $r_c$: Center-split ratio

1: if $K = 1$ then
2: Return 0
3: else
4: $K_L \leftarrow K \cdot r_c$
5: $K_R \leftarrow K - K_L$
6: $N_L \leftarrow N \cdot r_p$
7: $N_R \leftarrow N - N_L$
8: Return $1 + (N_L \cdot \text{EvalWAD}(N_L, K_L, r_p, r_c) + N_R \cdot \text{EvalWAD}(N_R, K_R, r_p, r_c))/N$
9: end if

As an example, Figure 3 presents two such trees generated by Algorithm 2 for the same number of centers ($K = 4$) and points ($N = 128$), but different values of $r_c$ and $r_p$. In Figure 3a, $r_c = r_p = 0.5$; as a result, the tree generated by Algorithm 2 has 4 leafs at level 2 with 25 points in each. In Figure 3b, $r_c = r_p = 0.25$; as a result, the tree has 3 levels instead of 2, and most points are in one of the deepest leafs.

The value of $D_{\text{Exp}}(\gamma, \mathcal{X}', \mathcal{S}')$ is given by the return of procedure EvalDExp($\gamma$, ...
Figure 3: Two trees generated by Algorithm 2 with different values of $r_c$ and $r_p$.

$X', S'$ presented in Algorithm 3. To explain the procedure, let $v$ be the current node of the decision tree under construction. Recall that if a cut $\gamma = (i, \theta)$ is applied on $v$ then it induces two edges leaving $v$, one associated with condition $x_i \leq \theta$ and the other with condition $x_i > \theta$. We say that an edge leaving $v$ is killer if its associated condition turns some non-redundant condition in the path, from the root to $v$, into a redundant one. The procedure first determines which edges induced by $\gamma$ on $v$ are killer and, based on that, it adjusts the value of $\hat{\text{WAD}}(\gamma, X', S')$ to take into account the metric WAES. As an example, if only the left edge leaving $v$ is killer then we discount $|X'_L|/|X'|$ from $\hat{\text{WAD}}(\gamma, X', S')$ because one condition in the path from the root to $v$ becomes redundant to explain the clusters of the left subtree of $v$.

By design, EvalDExp prioritizes the choice of cuts at node $v$ that are associated with coordinates that have already been used by some cut in the path from the root to $v$. This way the strategy tends to produce redundant conditions and, therefore, to find good trees in terms of WAES.

To summarize, ExShallow follows the steps of Algorithm 1. At line 5, it calls EvalDExp, presented in Algorithm 3, to evaluate $DExp(\gamma, X', S')$ and the value $\text{Price}(\gamma, X', S')$ is calculated via Equations 5, 6 and 7.
Algorithm 3 EvalDExp($\gamma, \mathcal{X}', S'$)

$\gamma$: cut; $\mathcal{X}'$: set of points; $S'$: set of centers
1. $v \leftarrow$ current node in the decision tree
2. ($\mathcal{X}'_L, \mathcal{X}'_R) \leftarrow$ partition of $\mathcal{X}'$ induced by $\gamma$
3. ($S'_L, S'_R) \leftarrow$ partition of $S'$ induced by $\gamma$
4. $r_p = \mathcal{X}'_L / \mathcal{X}'$
5. $r_c = S'_L / S'$
6. $\hat{\text{WAD}} = \text{EvalWAD}(|\mathcal{X}'|, |S'|, r_p, r_c)$
7. if no edge induced by $\gamma$ on $v$ is killer then
8. Return $\hat{\text{WAD}}$
9. else if only the left edge induced by $\gamma$ on $v$ is killer then
10. Return $\hat{\text{WAD}} - |\mathcal{X}'_L| / |\mathcal{X}'|$
11. else if only the right edge induced by $\gamma$ on $v$ is killer then
12. Return $\hat{\text{WAD}} - |\mathcal{X}'_R| / |\mathcal{X}'|$
13. else
14. Return $\hat{\text{WAD}} - 1$
15. end if

3.1. Implementation details and time-complexity analysis for ExShallow

ExShallow can be implemented in $O(n \cdot k \cdot d \cdot \text{WAD}(D))$ time, where $\text{WAD}(D)$ is the WAD of the decision tree $D$ built by the algorithm. Given the set of points $\mathcal{X}$ and the reference centers $S$, the algorithm first obtains $d$ sorted lists, where the $i$-th list corresponds to the set of points in $\mathcal{X} \cup S$ sorted by component $i$. This initial sorting step takes $O(d(n + k) \log(n + k))$ time and it is only performed in the root of the tree.

Having the $d$ sorted lists at node $v$, it is shown in [3] that [7] can be computed for all valid cuts in $O(dn_v k_v)$ time, where $n_v$ and $k_v$ are, respectively, the number of points and centers that reach $v$. In addition, the computation of $\hat{\text{WAD}}$, via Algorithm 2, takes $O(k_v)$ time per cut and, then, $O(dn_v k_v)$ time for all cuts.

To find out which of the edges are killer in Algorithm 3, we maintain a data structure, namely $A$, with $2d$ entries. For each $i \in [d]$, $A[i]$.left (resp. $A[i]$.right) stores the number of left (resp. right) edges that leave $i$-nodes that lie in the path from the root to the current node. To determine if a left (resp. right) edge leaving an $i$-node is killer we test whether $A[i]$.left > 0 (resp. $A[i]$.right > 0 ) or not. In the positive case the edge is killer, otherwise it is not.

The data structure $A$ can be updated in $O(1)$ time: if the chosen cut at
node \( v \) is an \( i \)-cut, then right before the recursive call at line 9 (resp. line 10) of \text{ExShallow} (Algorithm 1) we increment by one unit \( A[i].\text{left} \) (resp. \( A[i].\text{right} \)), and when we return from the recursion we decrease the respective counter by 1.

After selecting the cut at node \( v \), the \( d \) sorted lists for the children of \( v \) are obtained in \( O(n_vd) \) time from the sorted lists for \( v \).

Thus, the total cost of the algorithm to build a tree \( D \) is proportional to

\[
\sum_{v \in D} n_v \cdot d \cdot k_v \leq \sum_{i=1}^{n} \ell_i \cdot d \cdot k,
\]

where \( \ell_i \) is the depth of data point \( i \) at \( D \). The rightmost term, however, is equal to \( \text{WAD}(D) \cdot n \cdot d \cdot k \).

The \( O(\text{WAD}(D) \cdot n \cdot d \cdot k) \) time complexity suggests that trees with low \( \text{WAD} \) are faster to build – which is good for our purposes, since by design our algorithm tries to build trees with this property.

### 3.2. Setting the trade-off parameter

In a typical case, users are interested in obtaining an explainable clustering with low cost. To achieve this goal they have to properly set the value of \( \lambda \). One possibility is performing a brute-force search over some set of values to find the one that yields the most suitable tree. However, this could be computationally expensive and also non-practical from the users’ perspective, as they would have to analyze many trees. Fortunately, as we explain, we can avoid that.

First we note that a reasonable interpretation for \( \lambda \) is how much we are willing to (locally) give up of cost, in percentage, to reduce by one unit the average size of the explanations. As an example, setting \( \lambda = 0.1 \) means that we accept an additive loss of up to 10% in terms of the partition cost to have explanations one unit shorter on average.

Under this perspective, we shall avoid large values for \( \lambda \), since partitions with high costs are not likely to produce coherent clusters, and making incoherent explainable clusters would be useless. In fact, as we show in our experiments, by setting \( \lambda \) to 0.03 we obtain significant improvements over the existing methods.

A good property of \text{Price} (Equation 5) is that its value for cuts of low \text{InducedCost}, the most relevant ones, lies in the interval \([1, 4k + 1]\), the same one in which both \text{WAES} [3] and \text{WAD} [2] lie, except for a constant
factor. Hence, we are trading off quantities with similar magnitudes, which is beneficial. This is formalized below.

**Lemma 1.** Let \( \mathcal{X}' \) and \( \mathcal{S}' \) be the set of data points and reference centers that reach a given node \( v \). Then, there is a cut \( \gamma' \) that satisfies \( 1 \leq \text{Price}(\gamma', \mathcal{X}', \mathcal{S}') \leq 4|\mathcal{S}'| + 1 \).

*Proof.* The lefthand side follows because any assignment between points and reference centers that is valid after applying a cut is also valid before the cut, so that \( \text{CurrentCost}(\mathcal{X}', \mathcal{S}') \leq \text{InducedCost}(\gamma, \mathcal{X}', \mathcal{S}') \), for every cut \( \gamma \).

For the righthand side, we use the ideas from [5]. Let \( \max_i \) and \( \min_i \) be the maximum and minimum values of the \( i \)-th component among the centers in \( \mathcal{S}' \), respectively. Moreover, let \( b_i = \max_i - \min_i \) and let \( p(\gamma) \) be the number of points in \( \mathcal{X}' \) that are separated from their closest centers in \( \mathcal{S}' \) when a cut \( \gamma \) is employed. We have that

\[
\text{InducedCost}(\gamma, \mathcal{X}', \mathcal{S}') \leq \text{CurrentCost}(\mathcal{X}', \mathcal{S}') + p(\gamma) \sum_{i=1}^{d} b_i^2.
\]

The reason is that \( \sum_{i=1}^{d} b_i^2 \) is an upper bound on the contribution for the \( k \)-means cost of a point that is separated from its closest center.

On the other hand, it follows from Lemma 5.7 of [5] that

\[
\text{CurrentCost}(\mathcal{X}', \mathcal{S}') \geq \frac{p^*}{4|\mathcal{S}'|} \sum_{i=1}^{d} b_i^2,
\]

where \( p^* \) is the number of points separated from their closest centers by the valid cut that separates the minimum number of points.

Thus, if \( \gamma \) is a cut that separates \( p^* \) points from its closest centers, we get that

\[
\text{InducedCost}(\gamma, \mathcal{X}', \mathcal{S}') \leq (4|\mathcal{S}'| + 1)\text{CurrentCost}(\mathcal{X}', \mathcal{S}'),
\]
establishing the result.

\[\square\]

4. Experiments

In this section we report our experimental study. We have two goals: understanding the impact of \( \lambda \) and, most importantly, comparing our strategy with other available proposals for building explainable clustering [5][6][4].
These methods start with the reference centers of a partition for the unrestricted $k$-means clustering problem and, then, build a tree in a top-down fashion by selecting at each node a cut that separates at least two reference centers. What distinguishes them is the strategy employed to choose the cut: IMM [5] selects the cut that minimizes the number of data points separated from their representatives; ExKMC [6] selects the cut that minimizes the overall $k$-means cost of the split when a single center (chosen from the original centers of the unrestricted solution) is assigned to all points in each side of the cut; and ExGreedy [4], as already mentioned, selects the cut that minimizes the InducedCost given by Equation (7).

In our evaluation, we considered 16 datasets of different sizes and characteristics, performing 10 or 30 seeded iterations in each of them, depending on the experiment. For each iteration, we find an unrestricted partition of the data by running Lloyd’s algorithm [20] with the ++ initialization [21], as implemented in Python’s scikit-learn package [22]. This unrestricted partition is provided to IMM and to ExKMC, as implemented in the ExKMC package [6], and to ExGreedy, implemented as an extension of the ExKMC package and available in [4]. Then, we provide the same unrestricted partition to ExShallow, available as ShallowTree in the package of the same name available in [4].

4.1. Dataset summary

Table 1 presents the size, dimension, and number of classes (which we use as the number of clusters) of the datasets in which we perform the experiments. All datasets are available online, and our code includes a script for retrieving and running tests on them. The number of instances, dimensions, and features is that of the final dataset used in our experiments (after removal of missing values and one-hot encoding of categorical variables, for instance). Most datasets are retrieved from OpenML [23] or UCI [24]. All datasets are anonymized and present no offensive content.

4.2. Results

Table 2 shows the main results of our experiments for the 16 datasets and for 4 different explainable clustering algorithms: ExShallow with $\lambda = 0.03$, IMM [5], ExKMC [6], and ExGreedy [4]. For each dataset, we ran 30
Table 1: Dataset summary: \( n \) is the number of data points, \( d \) is the dimension, and \( k \) is the number of desired clusters.

| Dataset                | \( n \)  | \( d \)  | \( k \) | Source          |
|------------------------|---------|---------|-------|----------------|
| Anuran                 | 7,195   | 22      | 10    | UCI            |
| Avila                  | 20,867  | 10      | 12    | UCI \[3\]     |
| Beer                   | 1,514,999 | 5   | 104   | OpenML         |
| BNG (audiology)        | 1,000,000 | 85  | 24    | OpenML         |
| Cifar10                | 60,000  | 3,072   | 10    | UCI \[25\]    |
| Collins                | 1,000   | 19      | 30    | OpenML         |
| Covtype                | 581,012 | 54      | 7     | OpenML \[26\] |
| Digits                 | 1797    | 64      | 10    | UCI \[27\]    |
| Iris                   | 150     | 4       | 3     | UCI \[28\]    |
| Letter                 | 20,000  | 16      | 26    | UCI \[29\]    |
| Mice                   | 552     | 77      | 8     | OpenML \[30\] |
| 20Newsgroups           | 18,846  | 1,069   | 20    | http://qwone.com/~jason/20Newsgroups/ |
| Pendigits              | 10,992  | 16      | 10    | UCI            |
| Poker                  | 1,025,010 | 10  | 10    | UCI            |
| Sensorless             | 58,509  | 48      | 11    | UCI            |
| Vowel                  | 990     | 10      | 11    | UCI            |
Table 2: Full results for experiments for all datasets and algorithms. ExShallow is run with $\lambda = 0.03$. Best results for each dataset are in bold. For the partition cost, WAES, and WAD, values in red (blue) are statistically larger (smaller) than those of ExShallow, with a confidence level of 95%. For the normalized information score, the results from the unexplained partition (via Lloyd’s algorithm) are taken to be the ground truth, and values in red (blue) are statistically smaller (larger) than those of ExShallow, with a confidence level of 95%.

| Dataset     | Median | Normalized Partition Cost | WAES | Normalized Mutual Information |
|-------------|--------|---------------------------|------|-------------------------------|
|             | $\lambda$ | ExShallow | ExGreedy | IMM | KMC | ExShallow | ExGreedy | IMM | KMC | ExShallow | ExGreedy | IMM | KMC |
| Anuran      | 10     | 1.12 | 1.17 | 1.19 | 3.60 | 4.84 | 5.36 | 3.63 | 3.78 | 4.96 | 5.68 | 3.72 | 0.66 | 0.65 | 0.63 | 0.59 |
| Avila       | 12     | 1.05 | 1.07 | 1.18 | 3.87 | 5.58 | 5.25 | 3.26 | 4.60 | 6.64 | 6.61 | 4.47 | 0.73 | 0.72 | 0.73 | 0.68 |
| Beer        | 104    | 1.16 | 1.19 | 1.83 | 1.27 | 7.35 | 8.13 | 7.80 | 6.34 | 10.47 | 15.69 | 54.31 | 7.35 | 0.83 | 0.82 | 0.76 | 0.81 |
| BNG         | 24     | 1.05 | 1.02 | 1.04 | 1.03 | 3.50 | 5.41 | 8.82 | 4.60 | 3.50 | 5.41 | 11.82 | 4.60 | 0.29 | 0.41 | 0.25 | 0.38 |
| Cifar10     | 10     | 1.16 | 1.17 | 1.22 | 1.19 | 3.37 | 3.60 | 5.70 | 3.63 | 3.37 | 3.60 | 5.70 | 3.63 | 0.29 | 0.29 | 0.25 | 0.27 |
| Collins     | 30     | 1.18 | 1.17 | 1.23 | 1.23 | 5.56 | 13.12 | 12.81 | 5.61 | 5.86 | 15.29 | 17.00 | 5.83 | 0.55 | 0.54 | 0.54 | 0.53 |
| Covtype     | 7      | 1.03 | 1.03 | 1.03 | 1.19 | 2.61 | 2.62 | 2.61 | 2.45 | 3.15 | 3.56 | 3.55 | 2.82 | 0.83 | 0.83 | 0.83 | 0.72 |
| Digits      | 10     | 1.19 | 1.21 | 1.23 | 1.22 | 3.96 | 5.65 | 5.36 | 3.80 | 3.96 | 5.65 | 5.36 | 3.80 | 0.58 | 0.55 | 0.55 | 0.54 |
| Iris        | 3      | 1.04 | 1.04 | 1.04 | 1.04 | 1.67 | 1.67 | 1.44 | 1.44 | 1.67 | 1.67 | 1.67 | 1.67 | 0.91 | 0.91 | 0.91 | 0.91 |
| Letter      | 26     | 1.19 | 1.23 | 1.30 | 1.36 | 5.26 | 11.37 | 12.64 | 5.44 | 5.48 | 12.50 | 14.85 | 5.54 | 0.61 | 0.58 | 0.56 | 0.53 |
| Mice        | 8      | 1.07 | 1.09 | 1.12 | 1.15 | 3.17 | 3.32 | 3.53 | 3.12 | 3.24 | 3.58 | 3.76 | 3.13 | 0.72 | 0.71 | 0.71 | 0.65 |
| 20Newsgroups| 29     | 1.05 | 1.01 | 1.01 | 1.01 | 1.12 | 15.61 | 15.53 | 13.80 | 1.22 | 15.63 | 15.53 | 13.80 | 0.10 | 0.55 | 0.56 | 0.53 |
| Pendigits   | 10     | 1.14 | 1.14 | 1.32 | 1.32 | 3.70 | 4.43 | 4.31 | 3.49 | 3.77 | 4.86 | 4.44 | 3.50 | 0.77 | 0.72 | 0.72 | 0.67 |
| Poker       | 10     | 1.10 | 1.10 | 1.10 | 1.12 | 3.35 | 3.37 | 3.37 | 3.23 | 3.35 | 3.37 | 3.37 | 3.23 | 0.41 | 0.41 | 0.41 | 0.40 |
| Sensorless  | 11     | 1.02 | 1.03 | 1.07 | 1.07 | 2.99 | 4.24 | 4.10 | 3.99 | 3.84 | 4.52 | 4.44 | 4.07 | 0.93 | 0.92 | 0.92 | 0.88 |
| Vowel       | 11     | 1.21 | 1.25 | 1.36 | 1.29 | 3.89 | 5.26 | 5.74 | 3.63 | 3.94 | 5.76 | 6.41 | 3.64 | 0.58 | 0.55 | 0.53 | 0.52 |
seeded iterations of Lloyd’s algorithm, and used the resulting (non explainable) partition as a starting point for each explainable clustering algorithm analyzed here.

We also performed statistical tests (one-sided $t$-tests, assuming the same variance for both distributions, and with a confidence level of 95%) to check the statistical significance of the difference between results from ExShallow and each of the other algorithms. Values in red (resp. blue) in Table 2 indicate that results for the algorithm in question are worse (resp. better) on average than those of ExShallow with a confidence level of 95%.

The partition costs are normalized by the cost of the unrestricted partition used as a starting point for the explainable clustering algorithms.

In terms of average cost, ExShallow beats (with 95% confidence) ExGreedy in 9 datasets, IMM in 11 and KMC in 13. It is beaten by at least one algorithm on 4 datasets, in two of them by less than 1%. Only for BNG and 20Newsgroups the partitions generated by ExShallow are clearly worse (by at most 4%), and for both datasets ExShallow returns partitions that are much more explainable (in terms of WAD and WAES) than those of the other algorithms.

In terms of WAES, ExShallow outperforms ExGreedy and IMM on 15 and 14 datasets, respectively. For many datasets it is beaten by KMC by a small margin and, when this happens, it almost always beats KMC in terms of partition cost, frequently by large margins. Observe the median of WAES in the last line of Table 2. Results for WAD are similar, although KMC more frequently outperforms ExShallow in this metric.

We also report the normalized mutual information score (NMI) of the partitions generated by the explainable algorithms, considering that the ground truth is the unrestrained partition from which they are derived; a value of 1 corresponds to a perfect correspondence between partitions. The partition generated by ExShallow is the closest to the unrestrained one for 7 datasets, and it’s as good as those generated by the other explainable algorithms in another one. ExShallow returns the worst partition (in terms of NMI) for a single dataset, 20Newsgroups.

In summary, our experiments suggest that ExShallow is almost always at least close to the best result in terms of both partition cost and explainability, and frequently has a significant advantage in at least one of these dimensions when compared to the other 3 algorithms (as can be seen in the results for Avila, Collins, Letter, and Pendigits, for instance).
Figure 4: Mean WAES per depth factor (for all datasets), normalized by the results for $\lambda = 0$ for each dataset. Error bars (with a confidence interval of 95%) are calculated using Python’s scipy package [32].

To illustrate the last point made above, we refer back to Figures 1 and 2. Both induce the same partition, but ExShallow generates a more balanced tree with a smaller weighted depth. The maximum depth of the ExGreedy tree is 7, against 4 for the ExShallow tree. The largest cluster in the partition (cluster 10, with 5,605 elements) has depth 7 in the ExGreedy tree, and explanation size 6 (condition $EX > 0.440$ at depth 3 is made redundant by condition $EX > 0.574$ at depth 4); in the ExShallow tree, it has both depth and explanation size 4.

4.3. Sensitivity of cost and weighted depth to variations in $\lambda$

Figure 4 shows how the average WAES of the partitions produced by ExShallow changes as $\lambda$ increases. To allow for a comparison between datasets, the values are normalized by those of the tree when $\lambda = 0$ (i.e., when depth is not taken into account by our cost function). For each dataset, we ran 10 seeded iterations of Lloyd’s algorithm and used the resulting partitions as a starting point for each instance of ExShallow with different values of $\lambda$.

ExShallow behaves as expected, with larger values of $\lambda$ associated with trees having lower WAES, on average. (Results for WAD are omitted as they are very similar in terms of correlation with $\lambda$.) We observe a sharp drop for small increments of $\lambda$ when starting from zero. The red value is 0.03, the one employed in the previous experiments.

Figure 5 shows how the mean cost of the partitions produced by our algorithm changes as $\lambda$ increases. To allow for a comparison between datasets, the costs are normalized by the cost of the unrestricted partition generated.
by Lloyd’s algorithm. The behavior is, in general, the expected one, with larger values of $\lambda$ associated with higher costs.

Combining these figures leads to the important, and perhaps surprising, empirical conclusion that working with a small $\lambda$ is very beneficial, as it significantly reduces the average weighted depth and explanation size without increasing the average cost of the partition.

4.4. Calibrating the trade-off between partition quality and explainability

The results presented in Figures 4 and 5 suggest that calibrating $\lambda$ may lead to significant improvements when ExShallow does not initially return partitions that are satisfactory in terms of either quality (cost) or explainability (WAES and/or WAD). We believe the results presented in Table 2 indicate ExShallow “out of the box” is at least competitive with, and arguably superior to, the most recent comparable algorithms presented and evaluated in the literature, but there is some room for improvement. For instance, although the partition induced by ExShallow for the 20Newsgroups dataset is much more explainable than those induced by the competition, the quality of the partition (both in terms of cost and NMI) suffers from it; and in many cases KMC induces partitions that are slightly more explainable, although their quality tends to be worse.

We can use the $\lambda$ parameter to adjust the trade-off between partition quality and explainability in ExShallow, something that is not possible in the other algorithms presented here. To do so, we devised a simple binary search strategy, that starts from our default value of $\lambda = 0.03$ and then, if necessary, decreases it to try and find a partition with smaller cost, or
increases it to try and find a partition with smaller \textit{WAES}. Given a goal cost $c^*$ and a goal \textit{WAES} $w^*$, the binary search aims to find a partition with cost $c \leq c^*$ and \textit{WAES} $w \leq w^*$; if it is unable to do so, it returns the partition with the smallest \textit{WAES} given that its cost does not exceed $c^*$.

We present the results of this binary search, over 30 seeded iterations for each algorithm, in Table 3. Considering that KMC frequently beats \textsc{ExShallow} in terms of \textit{WAES}, we used its results as our goal; the idea being to check if we can “dominate” its results (i.e., induce partitions that have, on average, both smaller costs and explanation sizes) in the datasets under analysis.

In terms of cost, \textsc{ExShallow}* (\textsc{ExShallow} with $\lambda$ optimized by the procedure described above) beats KMC in all but two datasets, where both algorithms are tied; in terms of \textit{WAES}, \textsc{ExShallow}* beats KMC in 13 datasets and is beaten by it in 3. Most notably, in the two datasets (\texttt{BNG} and \texttt{20Newsgroups}) for which KMC induces less costly partitions than \textsc{ExShallow}, \textsc{ExShallow}* induces partitions that beat the ones generated by KMC in both dimensions.

4.5. Running times

Table 4 presents the average running times, over 30 seeded iterations, for each dataset and algorithm – including Lloyd’s algorithm (\textit{KMeans}), which finds the partition used as a starting point for all four algorithms, and \textsc{ExShallow}*. For all explainable algorithms, we add to their running time that of \textit{KMeans}, as an unrestrained partitioned is needed as a starting point for them to find an explainable partition. Disregarding \textsc{ExShallow}*, which can be expensive (as it performs several iterations of \textsc{ExShallow}), \textsc{ExGreedy} is the slowest explainable algorithm for all datasets except \textit{Beer}, for which both IMM and KMC are slower. \textsc{ExShallow}’s running times are typically closer to those of \textsc{ExGreedy} than those of IMM and KMC, which tend to be faster. Overall, we do not perceive running time to be a significant hindrance in choosing \textsc{ExShallow} over the other explainable clustering algorithms analyzed here, particularly due to the overhead imposed by initially running \textit{KMeans}.

5. Conclusions

We discussed how explainable an “explainable partition” actually is, by analyzing the average depth of its underlying decision tree and the average
Table 3: Comparison between results for KMC and ExShallow* (ExShallow with \( \lambda \) optimized via binary search to find a better partition than KMC’s in terms of both cost and WAES). Best results for each dataset are in bold. For the normalized partition cost (NPC), WAES, and WAD, values in red (blue) are statistically larger (smaller) than those of ExShallow*, with a confidence level of 95%. For the normalized information score (NMI), the results from the unexplained partition (via Lloyd’s algorithm) are taken to be the ground truth, and values in red (blue) are statistically smaller (larger) than those of ExShallow*, with a confidence level of 95%.

| Dataset     | NPC  | WAES | WAD  | NMI  |
|-------------|------|------|------|------|
|             | KMC  | ExShallow* | KMC  | ExShallow* | KMC  | ExShallow* | KMC  | ExShallow* |
| Anuran      | 1.32 | 1.20 | 3.41 | 3.19 | 3.41 | 3.33 | 0.64 | 0.68 |
| Avila       | 1.18 | 1.15 | 3.26 | 3.24 | 4.47 | 3.76 | 0.68 | 0.64 |
| Beer        | 1.27 | 1.22 | 6.34 | 7.25 | 7.35 | 10.53 | 0.81 | 0.82 |
| BNG         | 1.03 | 1.02 | 4.60 | 4.50 | 4.60 | 4.50 | 0.38 | 0.38 |
| Cifar10     | 1.19 | 1.16 | 3.63 | 3.37 | 3.63 | 3.37 | 0.27 | 0.29 |
| Collins     | 1.23 | 1.20 | 5.61 | 4.97 | 5.83 | 5.42 | 0.53 | 0.53 |
| Covtype     | 1.13 | 1.12 | 2.45 | 2.44 | 2.82 | 2.65 | 0.72 | 0.75 |
| Digits      | 1.22 | 1.19 | 3.80 | 3.65 | 3.80 | 3.65 | 0.54 | 0.56 |
| Iris        | 1.04 | 1.04 | 1.44 | 1.67 | 1.67 | 1.67 | 0.91 | 0.91 |
| Letter      | 1.36 | 1.24 | 5.44 | 4.81 | 5.54 | 5.02 | 0.53 | 0.58 |
| Mice        | 1.15 | 1.10 | 3.12 | 2.97 | 3.13 | 3.11 | 0.65 | 0.70 |
| 20Newsgroups| 1.01 | 1.01 | 13.80 | 13.45 | 13.80 | 13.78 | 0.53 | 0.53 |
| Pendigits   | 1.32 | 1.15 | 3.49 | 3.28 | 3.50 | 3.37 | 0.67 | 0.75 |
| Poker       | 1.12 | 1.11 | 3.23 | 3.33 | 3.23 | 3.33 | 0.40 | 0.40 |
| Sensorless  | 1.07 | 1.02 | 3.99 | 2.99 | 4.07 | 3.84 | 0.88 | 0.91 |
| Vowel       | 1.29 | 1.24 | 3.63 | 3.41 | 3.64 | 3.50 | 0.52 | 0.56 |
| Median      | 1.17 | 1.15 | 3.53 | 3.35 | 3.72 | 3.58 | 0.59 | 0.61 |
Table 4: Average running times (in seconds) for each algorithm and dataset, including Lloyd’s algorithm (KMEANS). Experiments were performed on 8 484 Intel Core i7-4790 processors @3.60GHz with 32 GB of RAM, running Ubuntu 20.04.3 LTS.

| Dataset      | K     | K-MEANS | ExShallow | ExShallow* | ExGreedy | IMM   | KMC  |
|--------------|-------|---------|-----------|------------|----------|-------|------|
| Anuran       | 10    | 0.46    | 0.75      | 0.74       | 0.74     | 0.56  | 0.62 |
| Avila        | 12    | 1.44    | 1.84      | 3.81       | 1.93     | 1.60  | 1.77 |
| Beer         | 104   | 722.65  | 730.90    | 938.72     | 731.75   | 752.97| 760.48 |
| BNG          | 24    | 896.97  | 1033.53   | 1669.02    | 1068.84  | 930.79| 956.54 |
| Cifar10      | 10    | 348.71  | 550.68    | 550.83     | 562.80   | 416.71| 437.85 |
| Collins      | 30    | 0.34    | 0.46      | 0.50       | 0.52     | 0.38  | 0.41 |
| Covtype      | 7     | 36.88   | 59.30     | 58.01      | 61.63    | 42.51 | 48.76 |
| Digits       | 10    | 0.27    | 0.44      | 0.44       | 0.48     | 0.32  | 0.36 |
| Iris         | 3     | 0.02    | 0.02      | 0.06       | 0.02     | 0.02  | 0.02 |
| Letter       | 26    | 4.15    | 4.83      | 4.84       | 5.55     | 4.40  | 4.57 |
| Mice         | 8     | 0.14    | 0.24      | 0.41       | 0.24     | 0.16  | 0.18 |
| 20Newsgroups | 20    | 47.38   | 57.51     | 311.83     | 107.11   | 54.56 | 63.59 |
| Pendigits    | 10    | 0.74    | 1.01      | 1.02       | 1.04     | 0.83  | 0.92 |
| Poker        | 10    | 83.67   | 97.32     | 328.40     | 97.30    | 87.32 | 95.67 |
| Sensorless   | 11    | 3.11    | 7.21      | 7.22       | 7.69     | 4.04  | 5.00 |
| Vowel        | 11    | 0.18    | 0.20      | 0.24       | 0.21     | 0.19  | 0.19 |
| Median       | 10.50 | 2.28    | 2.82      | 3.81       | 3.22     | 2.48  | 2.65 |

number of rules needed to explain each cluster in the partition (both metrics being weighted by the number of points assigned to each leaf/cluster). In most of the previous work on explainable clustering via decision trees, measures related to the depths of the leaves were largely ignored.

We present ExShallow, a simple and efficient algorithm that seeks to minimize both the cost of the resulting explainable partition and the aforementioned metrics. The algorithm has a tunable parameter $\lambda$ that allows to trade-off cost and explainability. Our experiments suggest that by working with a (fixed) small value for $\lambda$, ExShallow produces partitions that are at least as good as, and many times significantly better than, those obtained by the available methods in the literature. Thus, we understand that it is a valuable tool for those interested in explainable partitions that optimize the quite popular $k$-means cost.

References

[1] N. Burkart, M. F. Huber, A survey on the explainability of supervised machine learning, J. Artif. Intell. Res. 70 (2021) 245–317. doi:10.1613/
[2] R. Piltaver, M. Lustrek, M. Gams, S. Martincic-Ipsic, What makes classification trees comprehensible?, Expert Syst. Appl. 62 (2016) 333–346. doi:10.1016/j.eswa.2016.06.009
URL https://doi.org/10.1016/j.eswa.2016.06.009

[3] C. De Stefano, M. Maniaci, F. Fontanella, A. Scotto di Freca, Reliable writer identification in medieval manuscripts through page layout features: The “Avila” Bible case, Engineering Applications of Artificial Intelligence 72 (2018) 99–110. doi:https://doi.org/10.1016/j.engappai.2018.03.023
URL https://www.sciencedirect.com/science/article/pii/S0952197618300721

[4] E. S. Laber, L. Murtinho, On the price of explainability for some clustering problems, in: M. Meila, T. Zhang (Eds.), Proceedings of the 38th International Conference on Machine Learning, Vol. 139 of Proceedings of Machine Learning Research, PMLR, 2021, pp. 5915–5925.
URL https://proceedings.mlr.press/v139/laberalber21a.html

[5] M. Moshkovitz, S. Dasgupta, C. Rashtchian, N. Frost, Explainable k-means and k-medians clustering, in: H. D. III, A. Singh (Eds.), Proceedings of the 37th International Conference on Machine Learning, Vol. 119 of Proceedings of Machine Learning Research, PMLR, 2020, pp. 7055–7065.
URL https://proceedings.mlr.press/v119/moshkovitz20a.html

[6] N. Frost, M. Moshkovitz, C. Rashtchian, ExKMC: Expanding explainable k-means clustering, arXiv (2020). doi:10.48550/ARXIV.2006.02399
URL https://arxiv.org/abs/2006.02399

[7] K. Makarychev, L. Shan, Near-optimal algorithms for explainable k-medians and k-means, in: M. Meila, T. Zhang (Eds.), Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event, Vol. 139 of Proceedings of Machine Learning Research, PMLR, 2021, pp. 7358–7367.
URL http://proceedings.mlr.press/v139/makarychev21a.html
[8] M. Charikar, L. Hu, Near-optimal explainable $k$-means for all dimensions, in: Proceedings of the 2022 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), 2022, pp. 2580–2606. arXiv: https://epubs.siam.org/doi/pdf/10.1137/1.9781611977073.101, doi:10.1137/1.9781611977073.101. URL https://epubs.siam.org/doi/abs/10.1137/1.9781611977073.101

[9] H. Esfandiari, V. Mirrokni, S. Narayanan, Almost tight approximation algorithms for explainable clustering, in: Proceedings of the 2022 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), 2022, pp. 2641–2663. arXiv: https://epubs.siam.org/doi/pdf/10.1137/1.9781611977073.103, doi:10.1137/1.9781611977073.103. URL https://epubs.siam.org/doi/abs/10.1137/1.9781611977073.103

[10] B. Gamlath, X. Jia, A. Polak, O. Svensson, Nearly-tight and oblivious algorithms for explainable clustering, in: M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, J. W. Vaughan (Eds.), Advances in Neural Information Processing Systems, Vol. 34, Curran Associates, Inc., 2021, pp. 28929–28939. URL https://proceedings.neurips.cc/paper/2021/file/f24ad6f72d6cc4cb51464f2b29ab69d3-Paper.pdf

[11] E. S. Laber, The computational complexity of some explainable clustering problems (2022). doi:10.48550/ARXIV.2208.09643. URL https://arxiv.org/abs/2208.09643

[12] L. Breiman, J. H. Friedman, R. A. Olshen, C. J. Stone, Classification and Regression Trees, Wadsworth and Brooks, Monterey, CA, 1984. URL https://doi.org/10.1201/9781315139470

[13] J. L. Bentley, Multidimensional binary search trees used for associative searching, Commun. ACM 18 (9) (1975) 509–517. doi:10.1145/361002.361007. URL https://doi.org/10.1145/361002.361007

[14] R. Fraiman, B. Ghattas, M. Svarc, Interpretable clustering using unsupervised binary trees Adv. Data Anal. Classif. 7 (2) (2013) 125–145.
[15] B. Liu, Y. Xia, P. Yu, Clustering via decision tree construction in: W. Chu, T. Young Lin (Eds.), Foundations and Advances in Data Mining, Springer Berlin Heidelberg, Berlin, Heidelberg, 2005, pp. 97–124. doi:10.1007/11362197_5.
URL https://doi.org/10.1007/11362197_5

[16] K. Makarychev, L. Shan, Explainable k-means: Don’t be greedy, plant bigger trees! in: Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2022, Association for Computing Machinery, New York, NY, USA, 2022, p. 1629–1642. doi:10.1145/3519935.3520056.
URL https://doi.org/10.1145/3519935.3520056

[17] D. Bertsimas, A. Orfanoudaki, H. Wiberg, Interpretable clustering via optimal trees, arXiv (2018). doi:10.48550/ARXIV.1812.00539.
URL https://arxiv.org/abs/1812.00539

[18] S. Saisubramanian, S. Galhotra, S. Zilberstein, Balancing the trade-off between clustering value and interpretability in: Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society, Association for Computing Machinery, New York, NY, USA, 2020, p. 351–357.
URL https://doi.org/10.1145/3375627.3375843

[19] A. Blanco-Justicia, J. Domingo-Ferrer, S. Martínez, D. Sánchez, Machine learning explainability via microaggregation and shallow decision trees, Knowledge-Based Systems 194 (2020) 105532. doi:https://doi.org/10.1016/j.knosys.2020.105532.
URL https://www.sciencedirect.com/science/article/pii/S0950705120300368

[20] S. Lloyd, Least squares quantization in PCM, IEEE Transactions on Information Theory 28 (2) (1982) 129–137. doi:10.1109/TIT.1982.1056489.
URL https://doi.org/10.1109/TIT.1982.1056489

[21] D. Arthur, S. Vassilvitskii, K-means++: The advantages of careful seeding in: Proceedings of the Eighteenth Annual ACM-SIAM Symposium
on Discrete Algorithms, SODA ’07, Society for Industrial and Applied Mathematics, USA, 2007, p. 1027–1035.
URL https://dl.acm.org/doi/10.5555/1283383.1283494

[22] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay, Scikit-learn: Machine Learning in Python. J. Mach. Learn. Res. 12 (2011) 2825–2830.
URL https://dl.acm.org/doi/10.5555/1953048.2078195

[23] J. Vanschoren, J. N. van Rijn, B. Bischl, L. Torgo, OpenML: Networked science in machine learning. SIGKDD Explor. Newsl. 15 (2) (2014) 49–60. doi:10.1145/2641190.2641198.
URL https://doi.org/10.1145/2641190.2641198

[24] D. Dua, C. Graff, UCI machine learning repository (2017).
URL http://archive.ics.uci.edu/ml

[25] A. Krizhevsky, Learning multiple layers of features from tiny images (2009) 32–33.
URL https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf

[26] R. Collobert, S. Bengio, Y. Bengio, A parallel mixture of SVMs for very large scale problems, in: T. Dietterich, S. Becker, Z. Ghahramani (Eds.), Advances in Neural Information Processing Systems, Vol. 14, MIT Press, 2001, pp. 633–640.
URL https://proceedings.neurips.cc/paper/2001/file/36ac8e558ac7690b6f44e2cb5ef93322-Paper.pdf

[27] E. Alpaydin, C. Kaynak, Cascading classifiers. Kybernetika 34 (4) (1998) 369–374.
URL http://www.kybernetika.cz/content/1998/4/369

[28] R. A. Fisher, The use of multiple measurements in taxonomic problems. Annals of Eugenics 7 (2) (1936) 179–188.
URL https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1469-1809.1936.tb02137.x
[29] C.-W. Hsu, C.-J. Lin, A comparison of methods for multiclass support vector machines, IEEE Transactions on Neural Networks 13 (2) (2002) 415–425. doi:10.1109/72.991427. URL https://doi.org/10.1109/72.991427

[30] C. Higuera, K. J. Gardiner, K. J. Cios, Self-organizing feature maps identify proteins critical to learning in a mouse model of down syndrome, PLOS ONE 10 (6) (2015) 1–28. doi:10.1371/journal.pone.0129126. URL https://doi.org/10.1371/journal.pone.0129126

[31] A. Strehl, J. Ghosh, Cluster ensembles — a knowledge reuse framework for combining multiple partitions, J. Mach. Learn. Res. 3 (2003) 583–617. doi:10.1162/153244303321897735. URL https://doi.org/10.1162/153244303321897735

[32] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, I. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, A. Vijaykumar, A. P. Bardelli, A. Rothberg, A. Hilboll, A. Kloeckner, A. Scopatz, A. Lee, A. Rokem, C. N. Woods, C. Fulton, C. Masson, C. Häggström, C. Fitzgerald, D. A. Nicholson, D. R. Hagen, D. V. Pasechnik, E. Olivetti, E. Martin, E. Wieser, F. Silva, F. Lenders, F. Wilhelm, G. Young, G. A. Price, G.-L. Ingold, G. E. Allen, G. R. Lee, H. Audren, I. Probst, J. P. Dietrich, J. Silterra, J. T. Webber, J. Slavič, J. Nothman, J. Buchner, J. Kulick, J. L. Schönberger, J. V. de Miranda Cardoso, J. Reimer, J. Harrington, J. L. C. Rodríguez, J. Nunez-Iglesias, J. Kuczynski, K. Fritz, M. Thoma, M. Newville, M. Kümmerer, M. Bolingbroke, M. Tartre, M. Pak, N. J. Smith, N. Nowaczyk, N. Shebanov, O. Pavlyk, P. A. Brodtkorb, P. Lee, R. T. McGibbon, R. Feldbauer, S. Lewis, S. Tygier, S. Sievert, S. Vigna, S. Peterson, S. More, T. Pudlik, T. Oshima, T. J. Pingel, T. P. Robitaille, T. Spura, T. R. Jones, T. Cera, T. Leslie, T. Zito, T. Krauss, U. Upadhyay, Y. O. Halchenko, Y. Vázquez-Baeza, S. 1.0 Contributors, Scipy 1.0: fundamental algorithms for scientific computing in Python, Nature Methods 17 (3) (2020) 261–272.
