Explainable k-Means and k-Medians Clustering

Sanjoy Dasgupta [1] **Nave Frost** [2] **Michal Moshkovitz** [1] **Cyrus Rashtchian** [1]

Abstract

Many clustering algorithms lead to cluster assignments that are hard to explain, partially because they depend on all the features of the data in a complicated way. To improve interpretability, we consider using a small decision tree to partition a data set into clusters, so that clusters can be characterized in a straightforward manner. We study this problem from a theoretical viewpoint, measuring cluster quality by the k-means and k-medians objectives. In terms of negative results, we show that popular top-down decision tree algorithms may lead to clusterings with arbitrarily large cost, and any clustering based on a tree with k leaves must incur an $\Omega(\log k)$ approximation factor compared to the optimal clustering. On the positive side, for two means/medians, we show that a single threshold cut can achieve a constant factor approximation, and we give nearly-matching lower bounds; for general $k \geq 2$, we design an efficient algorithm that leads to an $O(k)$ approximation to the optimal k-medians and an $O(k^2)$ approximation to the optimal k-means. Prior to our work, no algorithms were known with provable guarantees independent of dimension and input size.

1. Introduction

A central direction in machine learning is understanding the reasoning behind decisions made by learned models (Lipton, 2018; Molnar, 2019; Murdoch et al., 2019). Prior work on AI explainability focuses on the interpretation of a black-box model, known as *post-modeling* explainability (Baehrens et al., 2010; Deutch & Frost, 2019). While methods such as LIME (Ribeiro et al., 2016) or Shapley explanations (Lundberg & Lee, 2017) have made progress in this direction, they do not provide direct insight into the underlying data set, and the explanations depend heavily on the given model. This has raised concerns about the applicability of current solutions, leading researchers to consider more principled approaches to interpretable methods (Rudin, 2019).

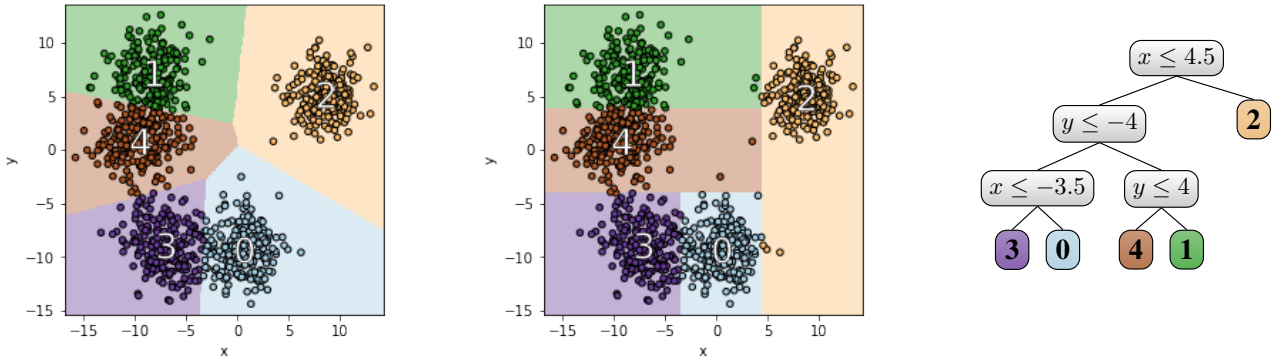
We address the challenge of developing machine learning systems that are explainable by design, starting from an *unlabeled* data set. Specifically, we consider *pre-modeling* explainability in the context of clustering. A common use of clustering is to identify patterns or discover structural properties in a data set by quantizing the unlabeled points. For instance, k-means clustering may be used to discover coherent groups among a supermarket's customers. While there are many good clustering algorithms, the resulting cluster assignments can be hard to understand because the clusters may be determined using all the features of the data, and there may be no concise way to explain the inclusion of a particular point in a cluster. This limits the ability of users to discern the commonalities between points within a cluster or understand why points ended up in different clusters.

Our goal is to develop accurate, efficient clustering algorithms with concise explanations of the cluster assignments. There should be a simple procedure using a few features to explain why any point belongs to its cluster. Small decision trees have been identified as a canonical example of an easily explainable model (Molnar, 2019; Murdoch et al., 2019), and previous work on explainable clustering uses an unsupervised decision tree (Bertsimas et al., 2018; Fraiman et al., 2013; Geurts et al., 2007; Ghattas et al., 2017; Liu et al., 2005). Each node of the binary tree iteratively partitions the data by thresholding on a single feature. We focus on finding k clusters, and hence, we use trees with k leaves. Each leaf corresponds to a cluster, and the tree is as small as possible. We refer to such a tree as a *threshold tree*.

There are many benefits of using a small threshold tree to produce a clustering. Any cluster assignment is explained by computing the thresholds along the root-to-leaf path. By restricting to k leaves, we ensure that each such path accesses at most $k-1$ features, independent of the data dimension. In general, a threshold tree provides an initial quantization of the data set, which can be combined with other methods for future learning tasks. While we consider static data sets, new data points can be easily clustered by using the tree, leading to explainable assignments.

[1]University of California, San Diego [2]Tel Aviv University. Correspondence to: Nave Frost <navefrost@mail.tau.ac.il>, Michal Moshkovitz <mmoshkovitz@eng.ucsd.edu>, Cyrus Rashtchian <crashtchian@eng.ucsd.edu>.

Proceedings of the 37^{th} International Conference on Machine Learning, Online, PMLR 119, 2020.



(a) Optimal 5-means clusters (b) Tree based 5-means clusters (c) Threshold tree

Figure 1: The optimal 5-means clustering (left) determines uses combinations of both features. The explainable clustering (middle) uses axis-aligned rectangles summarized by the threshold tree (right). Because the clusters contain nearby points, a small threshold tree makes very few mistakes and leads to a good approximation.

To analyze clustering quality, we consider the k-means and k-medians objectives (MacQueen, 1967; Steinhaus, 1956). The goal is to efficiently determine a set of k centers that minimize either the squared ℓ_2 or the ℓ_1 distance, respectively, of the input vectors to their closest center.

Figure 1 provides an example of standard and explainable k-means clustering on the same data set. The left figure shows an optimal 5-means clustering. The figure in the middle shows an explainable 5-means clustering, determined by the tree on the right. The tree has five leaf nodes, and vectors are assigned to clusters based on the thresholds. Geometrically, the tree defines a set of axis-aligned cuts that determine the clusters. While the two clusterings are very similar, using the threshold tree leads to easy explanations, whereas using a standard k-means clustering algorithm leads to more complicated clusters. The difference between the two approaches becomes more evident in higher dimensions, because standard algorithms will likely determine clusters based on all of the feature values.

To reap the benefits of explainable clusters, we must ensure that the data partition is a good approximation of the optimal clustering. While many efficient algorithms have been developed for k-means/medians clustering, the resulting clusters are often hard to interpret (Arthur & Vassilvitskii, 2007; Kanungo et al., 2002; Ostrovsky et al., 2013; Shalev-Shwartz & Ben-David, 2014). For example, Lloyd's algorithm alternates between determining the best center for the clusters and reassigning points to the closest center (Lloyd, 1982). The resulting set of centers depends in a complex way to the other points in the data set. Therefore, the relationship between a point and its nearest center may be the result of an opaque combination of many feature values. This issue persists even after dimension reduction or feature selection, because a non-explainable clustering algorithm is often invoked on the modified data set. As our focus is on pre-modeling explanability, we aim for simple explanations that use the original feature vectors.

Figure 1 depicts a situation where the optimal clustering is approximated by one that is induced by a tree. But it is not clear whether this is possible in general. Our first technical challenge is to understand the *price of explainability* in the context of clustering: that is, the multiplicative blowup in k-means (or k-medians) cost that is inevitable if we force our final clustering to have an interpretable form. The second challenge is to actually find such a tree *efficiently*. This is non-trivial because it requires a careful, rather than random or exhaustive, choice of a subset of features. As we will see, the analysis that is ultimately needed is quite novel even given the vast existing literature on clustering.

1.1. Our contributions

We provide several new theoretical results on explainable k-means and k-medians clustering. Our new algorithms and lower bounds are summarized in Table 1.

Basic limitations. A partition into k clusters can be realized by a binary threshold tree with $k-1$ internal splits. This uses at most $k-1$ features, but is it possible to use even fewer, say $\log k$ features? In Section 3, we demonstrate a simple data set that requires $\Omega(k)$ features to achieve a explainable clustering with bounded approximation ratio compared to the optimal k-means/medians clustering. The depth of the tree might need to be $k-1$ in the worst case.

One idea for building a tree is to begin with a good k-means (or k-medians) clustering, use it to label all the points, and then apply a supervised decision tree algorithm that

attempts to capture this labeling. In Section 3, we show that standard decision tree algorithms, such as ID3, may produce clusterings with arbitrarily high cost. Thus, existing splitting criteria are not suitable for finding a low-cost clustering, and other algorithms are needed.

New algorithms. On the positive side, we provide efficient algorithms to find a small threshold tree that comes with provable guarantees on the cost. We note that using a small number of clusters is preferable for easy interpretations, and therefore k is often relatively small. For the special case of two clusters ($k = 2$), we show (Theorem 1) that a single threshold cut provides a constant-factor approximation to the optimal 2-medians/means clustering, with a closely-matching lower bound (Theorem 2), and we provide an efficient algorithm for finding the best cut. For general k, we show how to approximate any clustering by using a threshold tree with k leaves (Algorithm 1) . The main idea is to minimize the number of mistakes made at each node in the tree, where a mistake occurs when a threshold separates a point from its original center. Overall, the cost of the explainable clustering will be close to the original cost up to a factor that depends on the tree depth (Theorem 3). In the worst-case, we achieve an approximation factor of $O(k^2)$ for k-means and $O(k)$ for k-medians compared to the cost of any clustering (e.g., the optimal cost). These results do not depend on the dimension or input size, and hence, we get a constant factor approximation when k is a constant.

Approximation lower bounds. Since our upper bounds depend on k, it is natural to wonder whether it is possible to achieve a constant-factor approximation, or whether the cost of explainability grows with k. On the negative side, we identify a data set such that any threshold tree with k leaves must incur an $\Omega(\log k)$-approximation for both k-medians and k-means (Theorem 4). For this data set, our algorithm achieves a nearly matching bound for k-medians.

Table 1: Summary of our new lower and upper bounds on approximating k-medians/means with explainable clusters.

	k-medians		k-means	
	$k = 2$	$k > 2$	$k = 2$	$k > 2$
Lower	$2 - \frac{1}{d}$	$\Omega(\log k)$	$3\left(1 - \frac{1}{d}\right)^2$	$\Omega(\log k)$
Upper	2	$O(k)$	4	$O(k^2)$

1.2. Related work

It is NP-hard to find the optimal k-means clustering (Aloise et al., 2009; Dasgupta, 2008) or even a very close approximation (Awasthi et al., 2015). Previous algorithms for k-medians/means use iterative algorithms to produce a good approximate clustering, but this leads to complicated clusters that depend on subtle properties of the data set (Aggarwal et al., 2009; Arthur & Vassilvitskii, 2007; Kanungo et al., 2002; Ostrovsky et al., 2013). Several papers have considered the use of decision trees for explainable clustering (Bertsimas et al., 2018; Fraiman et al., 2013; Geurts et al., 2007; Ghattas et al., 2017; Liu et al., 2005). However, all prior work on this topic is empirical, without any theoretical analysis of quality compared to the optimal clustering.

One way to cluster based on few features is to use dimensionality reduction. Two main types of dimensionality reduction methods have been investigated for k-medians/means. Work on *feature selection* shows that it is possible to cluster based on $\Theta(k)$ features and obtain a constant factor approximation for k-means/medians (Boutsidis et al., 2009; Cohen et al., 2015). However, after selecting the features, these methods employ existing approximation algorithms to find a good clustering, and hence, the cluster assignments are not explainable. Work on *feature extraction* shows that it is possible to use the Johnson-Lindenstrauss transform to $\Theta(\log k)$ dimensions, while preserving the clustering cost (Becchetti et al., 2019; Makarychev et al., 2019). Again, this relies on running a k-means/medians algorithm after projecting to the low dimensional subspace. The resulting clusters are not explainable, and moreover, the features are arbitrary linear combinations of the original features.

Besides explainability, many other clustering variants have received recent attention, such as fair clustering (Backurs et al., 2019; Bera et al., 2019; Huang et al., 2019; Kleindessner et al., 2019; Mahabadi & Vakilian, 2020; Schmidt et al., 2019), online clustering (Bhaskara & Rwanpathirana, 2020; Cohen-Addad et al., 2019; Hess & Sabato, 2019; Liberty et al., 2016; Moshkovitz, 2019), and the use of same-cluster queries (Ailon et al., 2018; Ashtiani et al., 2016; Huleihel et al., 2019; Mazumdar & Saha, 2017).

2. Preliminaries

Throughout we use bold variables for vectors, and we use non-bold for scalars such as feature values. Given a set of points $\mathcal{X} = \{\mathbf{x}^1, \dots, \mathbf{x}^n\} \subseteq \mathbb{R}^d$ and an integer k the goal of k-medians and k-means clustering is to partition $\mathcal{X}$ into k subsets and minimize the distances of the points to the centers of the clusters.

It is known that the optimal centers correspond to means or medians of the clusters, respectively. Denoting the centers as $\boldsymbol{\mu}^1, \dots, \boldsymbol{\mu}^k$, the aim of k-means is to find a clustering that minimizes the following objective

$$\mathrm{cost}_2(\boldsymbol{\mu}^1, \dots, \boldsymbol{\mu}^k) = \sum_{\mathbf{x} \in \mathcal{X}} \|\mathbf{x} - c_2(\mathbf{x})\|_2^2 ,$$

where $c_2(\mathbf{x}) = \arg\min_{\boldsymbol{\mu} \in \{\boldsymbol{\mu}^1, \dots, \boldsymbol{\mu}^k\}} \|\boldsymbol{\mu} - \mathbf{x}\|_2$.

Similarly, the goal of k-medians is to minimize

$$\mathrm{cost}_1(\boldsymbol{\mu}^1, \dots, \boldsymbol{\mu}^k) = \sum_{\mathbf{x} \in \mathcal{X}} \|\mathbf{x} - c_1(\mathbf{x})\|_1 ,$$

where $c_1(\mathbf{x}) = \arg\min_{\boldsymbol{\mu} \in \{\boldsymbol{\mu}^1, \ldots, \boldsymbol{\mu}^k\}} \|\boldsymbol{\mu} - \mathbf{x}\|_1$.

As it will be clear from context whether we are talking about k-medians or k-means, we often abuse notation and simply write cost and $c(\mathbf{x})$, and we let *opt* denote the optimal clustering.

2.1. Clustering using threshold trees

Perhaps the simplest way to define two clusters is to use a *threshold cut*, which partitions the data based on a threshold for a single feature. More formally, the two clusters can be written as $\widehat{C}^{\theta,i} = (\widehat{C}^1, \widehat{C}^2)$, which is defined using a coordinate i and a threshold $\theta \in \mathbb{R}$ in the following way. For each input point $\mathbf{x} \in \mathcal{X}$, we place $\mathbf{x} = [x_1, \ldots, x_d]$ in the first cluster $\widehat{C}^1$ if $x_i \leq \theta$, and otherwise $\mathbf{x} \in \widehat{C}^2$. A threshold cut can be used to explain 2-means or 2-medians clustering because a single feature and threshold determines the division of points into two clusters.

For $k > 2$ clusters, we consider iteratively using threshold cuts as the basis for the cluster explanations. More precisely, we construct a binary *threshold tree*. This tree is an unsupervised variant of a decision tree. Each internal node contains a single feature and threshold, which iteratively partitions the data, leading to clusters determined by the vectors that reach the leaves. We focus on trees with exactly k leaves, one for each cluster $\{1, 2, \ldots, k\}$, which also limits the depth and total number of features to at most $k - 1$.

When clustering using such a tree, it is easy to understand why $\mathbf{x}$ was assigned to its cluster: we may simply inspect the threshold conditions on the root-to-leaf path for $\mathbf{x}$. This also ensures the number of conditions for the cluster assignment is rather small, which is crucial for interpretability. Notice that these tree-based explanations are especially useful in high-dimensional space, when the number of clusters is much smaller than the input dimension ($k \ll d$).

More formally, a threshold tree T with k leaves induces a k-clustering of the data. If we denote these clusters as $\widehat{C}^j \subseteq \mathcal{X}$, the k-medians/means cost of the tree is defined as

$$\text{cost}_1(T) = \sum_{j=1}^{k} \sum_{x \in \widehat{C}^j} \|x - \text{median}(\widehat{C}^j)\|_1$$

$$\text{cost}_2(T) = \sum_{j=1}^{k} \sum_{x \in \widehat{C}^j} \|x - \text{mean}(\widehat{C}^j)\|_2^2$$

Our goal is to understand when it is possible to efficiently produce a tree T such that $\text{cost}(T)$ is not too large compared to the optimal k-medians/means cost. Specifically, we say that an algorithm is an *a-approximation*, if the cost is at most a times the optimal cost, i.e., if the algorithm returns threshold tree T then we have $\text{cost}(T) \leq a \cdot \text{cost}(opt)$.

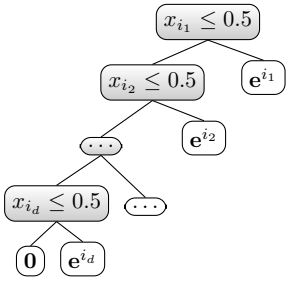


Figure 2: Optimal threshold tree for the data set in $\mathbb{R}^{k-1}$ consisting of the $k-1$ standard basis vectors and the all zeros vector. Any optimal tree must use all $k-1$ features and have depth $k-1$.

3. Motivating Examples

Using $k-1$ features may be necessary. We start with a simple but important bound showing that trees with depth less than k (or fewer than $k-1$ features) can be arbitrarily worse than the optimal clustering. Consider the data set consisting of the $k-1$ standard basis vectors $\mathbf{e}^1, \ldots, \mathbf{e}^{k-1} \in \mathbb{R}^{k-1}$ along with the all zeros vector. As this data set has k points, the optimal k-median/means cost is zero, putting each point in its own cluster. Unfortunately, it is easy to see that for this data, depth $k-1$ is necessary for clustering with a threshold tree. Figure 2 depicts an optimal tree for this data set. Shorter trees do not work because projecting onto any $k-2$ coordinates does not separate the data, as at least two points will have all zeros in these coordinates. Therefore, any tree with depth at most $k-2$ will put two points in the same cluster, leading to non-zero cost, whereas the optimal cost is zero. In other words, for this data set, caterpillar trees such as Figure 2 are necessary and sufficient for an optimal clustering. This example also shows that $\Theta(k)$ features are tight for feature selection (Cohen et al., 2015) and provides a separation with feature extraction methods that use a linear map to only a logarithmic number of dimensions (Becchetti et al., 2019; Makarychev et al., 2019).

Standard top-down decision trees do not work A natural approach to building a threshold tree is to (1) find a good k-medians or k-means clustering using a standard algorithm, then (2) use it to label all the points, and finally (3) apply a supervised decision tree learning procedure, such as ID3 (Quinlan, 1986; 2014) to find a threshold tree that agrees with these cluster labels as much as possible. ID3, like other common decision tree algorithms, operates in a greedy manner, where at each step it finds the best split in terms of *entropy* or *information gain*. We will show that this is not a suitable strategy for clustering and that the resulting tree can have cost that is arbitrarily bad. In what follows, denote by $\text{cost}(ID3_\ell)$ the cost of the decision tree with ℓ leaves returned by ID3 algorithm.

Figure 3 depicts a data set $\mathcal{X} \subseteq \mathbb{R}^2$ partitioned into three

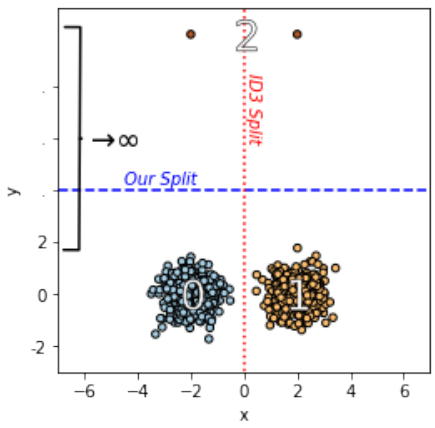


Figure 3: Comparison between the first split of an entropy-based decision tree (e.g. ID3) versus the optimal first threshold. By placing the top two points in separate clusters, the ID3 split results in a 3-means/medians clustering with arbitrarily worse cost.

clusters $\mathcal{X} = \mathcal{X}_0 \cup \mathcal{X}_1 \cup \mathcal{X}_2$. We define two centers $\boldsymbol{\mu}^0 = (-2, 0)$ and $\boldsymbol{\mu}^1 = (2, 0)$ and for each $i \in \{0, 1\}$, we define $\mathcal{X}_i$ as 500 i.i.d. points $\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}^i, \epsilon)$ for some small $\epsilon > 0$. Then, $\mathcal{X}_2 = \{(-2, v), (2, v)\}$ where $v \to \infty$. With high probability, we have that the optimal 3-means clustering is $(\mathcal{X}_0, \mathcal{X}_1, \mathcal{X}_2)$, i.e. $\mathbf{x} \in \mathcal{X}$ gets label $y \in \{0, 1, 2\}$ such that $\mathbf{x} \in \mathcal{X}_y$. The ID3 algorithm minimizes the entropy at each step. In the first iteration, it splits between the two large clusters. As a result $(-2, v)$ and $(2, v)$ will also be separated from one another. Since $ID3_3$ outputs a tree with exactly three leaves, one of the leaves must contain a point from $\mathcal{X}_2$ together with points from either $\mathcal{X}_0$ or $\mathcal{X}_1$, this means that $\text{cost}(ID3_3) = \Omega(v) \to \infty$. Note that $\text{cost}((\mathcal{X}_1, \mathcal{X}_2, \mathcal{X}_3))$ does not depend on v, and hence, it is substantially smaller than $\text{cost}(ID3_3)$. Unlike ID3, the optimal threshold tree first separates $\mathcal{X}_2$ from $\mathcal{X}_0 \cup \mathcal{X}_1$, and in the second split it separates $\mathcal{X}_0$ and $\mathcal{X}_1$. In other words, putting the outliers in a separate cluster is necessary for an optimal clustering. We note that it is easy to extend this example to larger numbers of clusters or when we allow ID3 to use more leaves.

4. Two Clusters Using a Single Threshold Cut

In this section, we consider the case of $k = 2$ clusters, and we study how well a single threshold cut can approximate the optimal partition into two clusters.

4.1. Algorithm for $k = 2$

We present an algorithm to efficiently minimize the cost using a single threshold cut. We begin by considering a single feature i and determining the value of the best threshold $\theta \in \mathbb{R}$ for this feature. Then, we minimize over all features $i \in [d]$ to output the best threshold cut. We focus on the 2-means algorithm; the 2-medians case is similar.

For feature i, we first sort the input points according to this feature, i.e., assume that the vectors are indexed as $x_i^1 \leq \ldots \leq x_i^n$. Notice that when restricting to this feature, there are only $n - 1$ possible partitions of the data set into two non-empty clusters. In particular, we can calculate the cost of all threshold cuts for the ith feature by scanning the values in this feature from smallest to largest. Then, we compute for each position $p \in [n - 1]$

$$\text{cost}(p) = \sum_{j=1}^{p} \left\|\mathbf{x}^j - \boldsymbol{\mu}^1(p)\right\|_2^2 + \sum_{j=p+1}^{n} \left\|\mathbf{x}^j - \boldsymbol{\mu}^2(p)\right\|_2^2,$$

where we denote the optimal centers for these clusters as $\boldsymbol{\mu}^1(p) = \frac{1}{p}\sum_{j=1}^{p} \mathbf{x}^j$ and $\boldsymbol{\mu}^2(p) = \frac{1}{n-p}\sum_{j=p+1}^{n} \mathbf{x}^j$ because these are the means of the first p and last $n - p$ points, respectively. Because there are $O(nd)$ possible thresholds, and naively computing the cost of each requires time $O(nd)$, this would lead to a running time of $O(n^2d^2)$. We can improve the time to $O(nd^2 + nd \log n)$ by using dynamic programming. Pseudo-code for the algorithm and description of the dynamic programming are in Appendix F.1.

4.2. Theoretical guarantees for $k = 2$

We prove that there always exists a threshold cut with low cost. Since our algorithm from the previous section finds the *best* cut, it achieves the guarantees of this theorem.

Theorem 1. *For any data set $\mathcal{X} \subseteq \mathbb{R}^d$, there is a threshold cut $\widehat{C}$ such that the 2-medians cost satisfies*

$$\text{cost}(\widehat{C}) \leq 2 \cdot \text{cost}(opt),$$

and there is a threshold cut $\widehat{C}$ such that the 2-means cost satisfies

$$\text{cost}(\widehat{C}) \leq 4 \cdot \text{cost}(opt),$$

where opt is the optimal 2-medians or means clustering.

The key idea of the analysis is to bound the cost of the threshold clustering in terms of the number of points on which it disagrees with an optimal clustering. Intuitively, if any threshold cut must lead to a fairly different clustering, then the cost of the optimal 2-medians/means clustering must also be large.

We note that it is possible to prove a slightly weaker bound by using the midpoint (for each feature) between the centers. When there are t changes, using the midpoint shows that $\text{cost}(opt)$ is at least t times *half* of the distance between the two centers. In other words, this argument only captures half of the cost. Using Hall's theorem, we show that each change corresponds to a pair in the matching, and each such pair contributes to $\text{cost}(opt)$ the distance between the

centers (not half as before). This improves the bound by a factor of two. The proof for 2-means is in Appendix D.

Notation. We denote the optimal clusters as C^1 and C^2 with optimal centers $\boldsymbol{\mu}^1$ and $\boldsymbol{\mu}^2$. Notice that we can assume $\mu_i^1 \leq \mu_i^2$ for each coordinate i because negating the ith coordinate for all points in the dataset does not change the 2-medians/means cost. Assume that a single threshold partitions $\mathcal{X}$ into $\widehat{C}^1, \widehat{C}^2$ such that $t = \min(|C^1 \Delta \widehat{C}^1|, |C^1 \Delta \widehat{C}^2|)$. We refer to these t points as *changes*, and we assume that t is the minimum possible over all threshold cuts.

If C^1, C^2 is an optimal 2-medians clustering, then we prove that the cost of $\widehat{C}^1, \widehat{C}^2$ is at most twice the optimal 2-medians cost. Similarly, if C^1, C^2 is an optimal 2-means clustering, then we prove that the cost of $\widehat{C}^1, \widehat{C}^2$ is at most four times the optimal 2-means cost. We simply need that the threshold cut $\widehat{C} = (\widehat{C}^1, \widehat{C}^2)$ minimizes the number of changes t compared to the optimal clusters.

We begin with a structural claim regarding the best threshold cut. This will allow us to obtain a tighter bound on the optimal 2-medians/means cost, compared to the general $k > 2$ case, in terms of the necessary number of changes. We utilize Hall's theorem on perfect matchings.

Proposition 1 (Hall's Theorem)**.** *Let (P, Q) be a bipartite graph. If all subsets $P' \subseteq P$ have at least $|P'|$ neighbors in Q, then there is a matching of size $|P|$.*

Lemma 1. *Let C^1 and C^2 be the optimal clustering of $\mathcal{X} \subseteq \mathbb{R}^d$, and assume that any threshold cut requires t changes. For each $i \in [d]$, there are t disjoint pairs of vectors $(\mathbf{p}^j, \mathbf{q}^j)$ in $\mathcal{X}$ such that $\mathbf{p}^j \in C^1$ and $\mathbf{q}^j \in C^2$ and $q_i^j \leq p_i^j$ for every $j \in [t]$.*

Proof. Let $\boldsymbol{\mu}^1$ and $\boldsymbol{\mu}^2$ be the centers for the optimal clusters C^1 and C^2. Focus on index $i \in [d]$, and assume without loss of generality that $\mu_i^1 \leq \mu_i^2$. The t pairs correspond to a matching the following bipartite graph (P, Q). Let $Q = C^2$ and define $P \subseteq C^1$ as the t points in C^1 with largest value in their ith coordinate. Connect $\mathbf{p} \in P$ and $\mathbf{q} \in Q$ by an edge if only if $q_i \leq p_i$. By construction, a matching with t edges implies our claim. By Hall's theorem, we just need to prove that $P' \subseteq P$ has at least $|P'|$ neighbors.

Index $P = \{\mathbf{p}^1, \ldots, \mathbf{p}^t\}$ by ascending value of ith coordinate, $p_i^1 \leq \cdots \leq p_i^t$. Now, notice that vertices in P have nested neighborhoods: for all $j > j'$, the neighborhood of $\mathbf{p}^{j'}$ is a subset of the neighborhood of $\mathbf{p}^j$. It suffices to prove that $\mathbf{p}^j$ has at least j neighbors, because this implies that any subset $P' \subseteq P$ has at least $|P'|$ neighbors, guaranteeing a matching of size $|P| = t$. Indeed, if $|P'| = b$ then we know that $\mathbf{p}^j \in P'$ for some $j \geq b$, implying that P' has at least $j \geq b = |P'|$ neighbors.

Assume for contradiction that $\mathbf{p}^j$ has at most $j-1$ neighbors. We argue that the threshold cut $x_i \leq p_i^j$ has fewer than t changes, which contradicts the fact that all threshold cuts must make at least t changes. By our assumption, there are at most $j-1$ points that are smaller than p_i^j and belong to the second cluster. By the definition of P, there are exactly $t - j$ points with a larger ith coordinate than p_i^j in the first cluster. Therefore, the threshold cut $x_i \leq p_i^j$ makes at most $(t-j) + (j-1) < t$ changes, a contradiction. □

Proof for 2-medians. Suppose $\boldsymbol{\mu}^1, \boldsymbol{\mu}^2$ are optimal 2-medians centers for clusters C^1 and C^2, and that the threshold cut $\widehat{C}$ makes t changes, which is the minimum possible. Applying Lemma 4 (see Appendix B) in the special case of $k = 2$ implies that

$$\text{cost}(\widehat{C}) \leq \text{cost}(\textit{opt}) + t \left\| \boldsymbol{\mu}^1 - \boldsymbol{\mu}^2 \right\|_1 .$$

We simply need to prove that $t \left\| \boldsymbol{\mu}^1 - \boldsymbol{\mu}^2 \right\|_1 \leq \text{cost}(\textit{opt})$.

Applying Lemma 1 for each coordinate $i \in [d]$ guarantees t pairs of vectors $(\mathbf{p}^1, \mathbf{q}^1), \ldots, (\mathbf{p}^t, \mathbf{q}^t)$ with the following properties. Each p_i^j corresponds to the ith coordinate of some point in C^1 and q_i^j corresponds to the ith coordinate of some point in C^2. Furthermore, for each coordinate, the t pairs correspond to $2t$ distinct points in $\mathcal{X}$. Finally, we can assume without loss of generality that $\mu_i^1 \leq \mu_i^2$ and $q_i^j \leq p_i^j$, which implies that

$$\begin{aligned}
\text{cost}(\textit{opt}) &\geq \sum_{i=1}^{d} \sum_{j=1}^{t} |\mu_i^2 - q_i^j| + |p_i^j - \mu_i^1| \\
&\geq \sum_{i=1}^{d} \sum_{j=1}^{t} (\mu_i^2 - q_i^j) + (p_i^j - \mu_i^1) \\
&\geq \sum_{i=1}^{d} \sum_{j=1}^{t} (\mu_i^2 - q_i^j) + (q_i^j - p_i^j) + (p_i^j - \mu_i^1) \\
&= t \cdot \sum_{i=1}^{d} (\mu_i^2 - \mu_i^1) = t \left\| \boldsymbol{\mu}^2 - \boldsymbol{\mu}^1 \right\|_1 .
\end{aligned}$$

4.3. Lower bounds for $k = 2$

We next show that optimal clustering is not, in general, realizable with a single threshold cut, except in a small number of dimensions (e.g., $d = 1$). Our lower bounds on the approximation ratio increase with the dimension, approaching two for 2-medians or three for 2-means.

The two lower bounds are based on a data set $\mathcal{X} \subseteq \mathbb{R}^d$ consisting of $2d$ points, split into two optimal clusters each with d points. The first cluster contains the d vectors of the form $\mathbf{1} - \mathbf{e}^i$, where $\mathbf{e}^i$ is the ith coordinate vector and $\mathbf{1}$ is the all-ones vector. The second cluster contains their negations, $-\mathbf{1} + \mathbf{e}^i$. Due to the zero-valued coordinate in each vector, any threshold cut must separate at least one

vector from its optimal center. In the case of 2-medians, each incorrect cluster assignment incurs a cost of $2d$. The optimal cost is roughly $2d$, while the threshold cost is roughly $4d$ (correct assignments contribute $\approx 2d$, plus $2d$ from the error), leading to an approximation ratio of nearly two. A similar result holds for 2-means. The proof of these two lower bounds is in Appendix E.

Theorem 2. *For any integer $d \geq 1$, define data set $\mathcal{X} \subseteq \mathbb{R}^d$ as above. Any threshold cut $\widehat{C}$ must have 2-medians cost*

$$\text{cost}(\widehat{C}) \geq \left(2 - \frac{1}{d}\right) \cdot \text{cost}(opt)$$

and 2-means cost

$$\text{cost}(\widehat{C}) \geq 3\left(1 - \frac{1}{d}\right)^2 \cdot \text{cost}(opt),$$

where opt is the optimal 2-medians or means clustering.

5. Threshold Trees with $k > 2$ Leaves

We provide an efficient algorithm to produce a threshold tree with k leaves that constitutes an approximate k-medians or k-means clustering of a data set $\mathcal{X}$. Our algorithm, Iterative Mistake Minimization (IMM), starts with a reference set of cluster centers, for instance from a polynomial-time constant-factor approximation algorithm for k-medians/means (Aggarwal et al., 2009), or from a domain-specific clustering heuristic. These centers may use all the features and produce complicated clusters in the d-dimensional space.

We then begin the process of finding an explainable approximation to this reference clustering, in the form of a threshold tree with k leaves, whose internal splits are based on single features. The way we do this is almost identical for k-medians and k-means, and the analysis is also nearly the same. Our algorithm is deterministic and its run time is only $O(kdn \log n)$, after finding the initial centers.

As discussed in Section 3, existing decision tree algorithms use greedy criteria that are not suitable for our tree-building process. However, we show that an alternative greedy criterion—minimizing the number of *mistakes* at each split (the number of points separated from their corresponding cluster center)—leads to a favorable approximation ratio to the optimal k-medians or k-means cost.

5.1. Our algorithm

Algorithm 1 takes as input a data set $\mathcal{X} \subseteq \mathbb{R}^d$. The first step is to obtain a reference set of k centers $\{\boldsymbol{\mu}^1, \ldots, \boldsymbol{\mu}^k\}$, for instance from a standard clustering algorithm. We assign each data point $\mathbf{x}^j$ the label y^j of its closest center. We then call the `build_tree` procedure, which looks for a tree-induced clustering that fits these labels.

Algorithm 1 ITERATIVE MISTAKE MINIMIZATION

Input : $\mathbf{x}^1, \ldots, \mathbf{x}^n$ – vectors in $\mathbb{R}^d$
k – number of clusters
Output : root of the threshold tree

$\boldsymbol{\mu}^1, \ldots \boldsymbol{\mu}^k \leftarrow$ `k-Means`$(\mathbf{x}^1, \ldots, \mathbf{x}^n, k)$
foreach $j \in [1, \ldots, n]$ **do**
  $y^j \leftarrow \arg\min_{1 \leq \ell \leq k} \|\mathbf{x}^j - \boldsymbol{\mu}^\ell\|$
end
return `build_tree`$(\{\mathbf{x}^j\}_{j=1}^n, \{y^j\}_{j=1}^n, \{\boldsymbol{\mu}^j\}_{j=1}^k)$

build_tree$(\{\mathbf{x}^j\}_{j=1}^m, \{y^j\}_{j=1}^m, \{\boldsymbol{\mu}^j\}_{j=1}^k)$:
  if $\{y^j\}_{j=1}^m$ *is homogeneous* **then**
    leaf.$cluster \leftarrow y^1$
    return leaf
  end
  foreach $i \in [1, \ldots, d]$ **do**
    $\ell_i \leftarrow \min_{1 \leq j \leq m} \mu_i^{y^j}$
    $r_i \leftarrow \max_{1 \leq j \leq m} \mu_i^{y^j}$
  end
  $i, \theta \leftarrow \arg\min_{i, \ell_i \leq \theta < r_i} \sum_{j=1}^m$ `mistake`$(\mathbf{x}^j, \boldsymbol{\mu}^{y^j}, i, \theta)$
  $\mathsf{M} \leftarrow \{j \mid$ `mistake`$(\mathbf{x}^j, \boldsymbol{\mu}^{y^j}, i, \theta) = 1\}_{j=1}^m$
  $\mathsf{L} \leftarrow \{j \mid (x_i^j \leq \theta) \wedge (j \notin \mathsf{M})\}_{j=1}^m$
  $\mathsf{R} \leftarrow \{j \mid (x_i^j > \theta) \wedge (j \notin \mathsf{M})\}_{j=1}^m$
  node.$condition \leftarrow$ "$x_i \leq \theta$"
  node.$lt \leftarrow$ `build_tree`$(\{\mathbf{x}^j\}_{j \in \mathsf{L}}, \{y^j\}_{j \in \mathsf{L}}, \{\boldsymbol{\mu}^j\}_{j=1}^k)$
  node.$rt \leftarrow$ `build_tree`$(\{\mathbf{x}^j\}_{j \in \mathsf{R}}, \{y^j\}_{j \in \mathsf{R}}, \{\boldsymbol{\mu}^j\}_{j=1}^k)$
  return node

mistake$(\mathbf{x}, \boldsymbol{\mu}, i, \theta)$:
  return $(x_i \leq \theta) \neq (\mu_i \leq \theta)$? 1 : 0

The tree is built top-down, using binary splits. Each node u of the tree can be associated with the portion of the input space that passes through that node, a hyper-rectangular region cell$(u) \subseteq \mathbb{R}^d$. If this cell contains two or more of the centers $\boldsymbol{\mu}^j$, then it needs to be split. We do so by picking the feature $i \in [d]$ and threshold value $\theta \in \mathbb{R}$ such that the resulting split $x_i \leq \theta$ sends at least one center to each side and moreover produces the fewest *mistakes*: that is, separates the fewest points in $\mathcal{X} \cap$ cell(u) from their corresponding centers in $\{\boldsymbol{\mu}^j : 1 \leq j \leq k\} \cap$ cell(u). We do not count points whose centers lie outside cell(u), since they are associated with mistakes in earlier splits. Figure 4 in Appendix A shows a step-by-step example.

We find the optimal split (i, θ) by searching over all pairs efficiently using dynamic programming. We then add this node to the tree, and discard the mistakes (the points that got split from their centers) before recursing on the left and right children. We terminate at a leaf node whenever all points have the same label (i.e., the subset of the data is *homogeneous*). Because there were k different labels to begin with, the resulting tree has exactly k leaves.

We next describe how to speed-up the algorithm using dynamic programming. In Section 5.2, we provide the main ideas for analyzing the IMM approximation guarantees.

Time analysis of the tree-building algorithm. We sketch how to execute the algorithm in time $O(kdn \log n)$ for an n-point data set. At each step of the top-down procedure,

we find a coordinate and threshold pair that minimizes the mistakes at this node (line 10 in `build_tree` procedure). We use dynamic programming to avoid recomputing the cost from scratch for each potential threshold. For each coordinate $i \in [d]$, we first sort the data points and centers. Then, we iterate over the possible thresholds. We claim that each internal node can be processed in time $O(dn \log n)$. The key observation is that each point will affect the number of mistakes at most two times. Indeed, when the threshold moves, either a data point or a center moves to the other side of the threshold. Since we know the number of mistakes from the previous threshold, we efficiently count the new mistakes as follows. If a single data point changes sides, then the number of mistakes changes by at most one. If a center switches sides, which happens at most once, then we can update the mistakes for this center. Overall, each data point affects the mistakes at most twice (once when changing sides, and once when its center changes sides). Thus, the running time for each internal node is $O(dn \log n)$. As the tree has $k-1$ internal nodes, the total time is $O(kdn \log n)$.

5.2. Approximation guarantee for the IMM algorithm

Our main theoretical result of this section is the following.

Theorem 3. *Suppose that IMM takes centers $\boldsymbol{\mu}^1, \ldots, \boldsymbol{\mu}^k$ and returns a tree T of depth H. Then,*

1. *The k-medians cost is at most*

$$\text{cost}(T) \leq (2H+1) \cdot \text{cost}(\boldsymbol{\mu}^1, \ldots, \boldsymbol{\mu}^k)$$

2. *The k-means cost is at most*

$$\text{cost}(T) \leq (8Hk+2) \cdot \text{cost}(\boldsymbol{\mu}^1, \ldots, \boldsymbol{\mu}^k)$$

In particular, IMM achieves worst case approximation factors of $O(k)$ and $O(k^2)$ using any $O(1)$ approximation to k-means or k-medians, respectively.

We state the theorem in terms of the depth of the tree to highlight that the approximation guarantee may depend on the structure of the input data. If the optimal clusters can be easily identified by a small number of salient features, then the tree may have depth $O(\log k)$. In the next section we provide a lower bound showing that an $\Omega(\log k)$ approximation factor is necessary for k-medians and k-means. For this data set, our algorithm produces a threshold tree with depth $O(\log k)$, and therefore, the analysis is tight for k-medians. We leave it as an intriguing open question whether the bound can be improved for k-means.

The proof of the approximation bound rests upon a simple characterization of the excess clustering cost induced by the tree. For any internal node u of the final tree T, let $\text{cell}(u) \subseteq \mathbb{R}^d$ denote the region of the input space that ends up in that node, and let $B(u)$ be the bounding box of the centers that lie in this node, $\{\boldsymbol{\mu}^j : 1 \leq j \leq k\} \cap \text{cell}(u)$. We will be interested in the diameter of this bounding box, measured either by ℓ_1 or squared ℓ_2 norm, and denoted $\text{diam}_1(B(u))$ and $\text{diam}_2^2(B(u))$, respectively.

Lemma 2. *If IMM takes centers $\boldsymbol{\mu}^1, \ldots, \boldsymbol{\mu}^k$ and returns a tree T that incurs t_u mistakes at node $u \in T$, then*

1. *The k-medians cost of T satisfies*

$$\text{cost}(T) \leq \text{cost}(\boldsymbol{\mu}^1, \ldots, \boldsymbol{\mu}^k) + \sum_{u \in T} t_u \text{diam}_1(B(u))$$

2. *The k-means cost of T satisfies*

$$\text{cost}(T) \leq 2\text{cost}(\boldsymbol{\mu}^1, \ldots, \boldsymbol{\mu}^k) + 2 \sum_{u \in T} t_u \text{diam}_2^2(B(u))$$

A detailed proof is given in Appendix B. Briefly, any point $\mathbf{x}$ that ends up in a different leaf from its correct center $\boldsymbol{\mu}^j$ incurs some extra cost. To bound this, consider the internal node u at which $\mathbf{x}$ is separated from $\boldsymbol{\mu}^j$. Node u also contains the center $\boldsymbol{\mu}^i$ that ultimately ends up in the same leaf as $\mathbf{x}$. For k-medians, the excess cost for $\mathbf{x}$ can then be bounded by $\|\boldsymbol{\mu}^i - \boldsymbol{\mu}^j\|_1 \leq \text{diam}_1(B(u))$. The argument for k-means is similar.

These $\sum_u t_u \text{diam}(B(u))$ terms can in turn be bounded in terms of the cost of the reference clustering.

Lemma 3. *If IMM takes centers $\boldsymbol{\mu}^1, \ldots, \boldsymbol{\mu}^k$ and returns a tree T of depth H that makes t_u mistakes at node $u \in T$,*

1. *The k-medians cost satisfies*

$$\sum_{u \in T} t_u \text{diam}_1(B(u)) \leq 2H \cdot \text{cost}(\boldsymbol{\mu}^1, \ldots, \boldsymbol{\mu}^k).$$

2. *The k-means cost satisfies*

$$\sum_{u \in T} t_u \text{diam}_2^2(B(u)) \leq 4Hk \cdot \text{cost}(\boldsymbol{\mu}^1, \ldots, \boldsymbol{\mu}^k).$$

The proof for this lemma is significantly more complicated, and it contains the main new techniques in our analysis. We provide a sketch of the proof; full details are in Appendix B.

The core challenge is that we aim to lower bound the cost of the given centers using only information about the number of mistakes at each internal node. Moreover, the IMM algorithm only minimizes the *number* of mistakes, and not the *cost* of each mistake. Therefore, we must show that if every axis-aligned cut in $B(u)$ separates at least t_u points $\mathbf{x}$ from their centers, then there must be a considerable distance between the points in $\text{cell}(u)$ and their centers.

To prove this, we analyze the structure of points in each cell. Specifically, we consider the single-coordinate projection of points in the box $B(u)$, and we order the centers in $B(u)$ from smallest to largest for the analysis. If there are k' centers in node u, we consider the partition of $B(u)$ into $2(k'-1)$ disjoint segments, splitting at the centers and at the midpoints between consecutive centers. Since t_u is the minimum number of mistakes, we must in particular have at least t_u mistakes from the threshold cut at each midpoint. We argue that each of these segments is covered at least t_u times by a certain set of intervals. Specifically, we consider the intervals between mistake points and their true centers, and we say that an interval *covers* a segment if the segment is contained in the interval. This allows us to capture the cost of mistakes at different distance scales. For example, if a point is very far from its true center, then it covers many disjoint segments, and we show that it also implies a large contribution to the cost. Claim 2 in Appendix B provides our main covering result, and we use this to argue that the cost of the given centers can be lower bounded in terms of the distance between consecutive centers in $B(u)$. For k-medians, we can directly derive a lower bound on the cost in terms of the ℓ_1 diameter $\mathrm{diam}_1(B(u))$. For k-means, however, we employ Cauchy-Schwarz, which incurs an extra factor of k in the bound with $\mathrm{diam}_2^2(B(u))$. Overall, we sum these bounds over the height H of the tree, leading to the claimed upper bounds in the above lemma.

5.3. Approximation lower bound

To complement our upper bounds, we show that a threshold tree with k leaves cannot, in general, yield better than an $\Omega(\log k)$ approximation to the optimal k-medians or k-means clustering.

Theorem 4. *For any $k \geq 2$, there exists a data set with k clusters such that any threshold tree T with k leaves must have k-medians and k-means cost at least*

$$\mathrm{cost}(T) \geq \Omega(\log k) \cdot \mathrm{cost}(opt),$$

where opt is the optimal k-medians or means clustering.

The data set is produced by first picking k random centers from the hypercube $\{-1, 1\}^d$, for large enough d, and then using each of these to produce a cluster consisting of the d points that can be obtained by replacing one coordinate of the center by zero. Thus the clusters have size d and radius $O(1)$. To prove the lower bound, we use ideas from the study of pseudo-random binary vectors, showing that projecting the centers to any subset of $m \lesssim \log_2 k$ coordinates take on all 2^m possible values, with each occurring roughly equally often. Then, we show that (i) the threshold tree must be essentially a complete binary tree with depth $\Omega(\log_2 k)$ to achieve a clustering with low cost, and (ii) any such tree incurs a cost of $\Omega(\log k)$ times more than the optimal for this data set (for both k-medians and k-means). The proof of Theorem 4 appears in Appendix C.

It would be interesting to improve our upper bounds on explainable clustering for well-separated data. Our lower bound of $\Omega(\log k)$ utilizes clusters with diameter $O(1)$ and separation $\Omega(d)$, where the hardness stems from the randomness of the centers. In this case, the approximation factor $\Theta(\log k)$ is tight because our upper bound proof actually provides a bound in terms of the tree depth (which is about $\log k$, see Appendix C.5). Therefore, an open question is whether a $\Theta(\log k)$ approximation is possible for any well-separated clusters (e.g., mixture of Gaussians with separated means and small variance).

6. Conclusion

In this paper we discuss the capabilities and limitations of explainable clusters. For the special case of two clusters ($k = 2$), we provide nearly matching upper and lower bounds for a single threshold cut. For general $k > 2$, we present the IMM algorithm that achieves an $O(H)$ approximation for k-medians and an $O(Hk)$ approximation for k-means when the threshold tree has depth H and k leaves. We complement our upper bounds with a lower bound showing that any threshold tree with k leaves must have cost at least $\Omega(\log k)$ more than the optimal for certain data sets. Our theoretical results provide the first approximation guarantees on the quality of explainable unsupervised learning in the context of clustering. Our work makes progress toward the larger goal of explainable AI methods with precise objectives and provable guarantees.

An immediate open direction is to improve our results for k clusters, either on the upper or lower bound side. One option is to use larger threshold trees with more than k leaves (or allowing more than k clusters). It is also an important goal to identify natural properties of the data that enable explainable, accurate clusters. Beyond k-medians/means, it would be interesting to develop other clustering methods using a small number of features (e.g., hierarchical clustering). Finally, we believe our algorithms would be useful in practice, as they are efficient and easily implementable.

Acknowledgements

Sanjoy Dasgupta has been supported by NSF CCF-1813160. Nave Frost has been funded by the European Research Council (ERC) under the European Unions Horizon 2020 research and innovation programme (Grant agreement No. 804302). The contribution of Nave Frost is part of a Ph.D. thesis research conducted at Tel Aviv University.

References

Aggarwal, A., Deshpande, A., and Kannan, R. Adaptive sampling for k-means clustering. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pp. 15–28. Springer, 2009.

Ailon, N., Bhattacharya, A., and Jaiswal, R. Approximate correlation clustering using same-cluster queries. In *Latin American Symposium on Theoretical Informatics*, pp. 14–27. Springer, 2018.

Aloise, D., Deshpande, A., Hansen, P., and Popat, P. NP-hardness of Euclidean sum-of-squares clustering. *Machine learning*, 75 (2):245–248, 2009.

Arthur, D. and Vassilvitskii, S. k-means++: The advantages of careful seeding. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete algorithms*, pp. 1027–1035. Society for Industrial and Applied Mathematics, 2007.

Ashtiani, H., Kushagra, S., and Ben-David, S. Clustering with same-cluster queries. In *Advances in neural information processing systems*, pp. 3216–3224, 2016.

Awasthi, P., Charikar, M., Krishnaswamy, R., and Sinop, A. K. The hardness of approximation of Euclidean k-means. In *31st International Symposium on Computational Geometry, SoCG 2015*, pp. 754–767. Schloss Dagstuhl-Leibniz-Zentrum fur Informatik GmbH, Dagstuhl Publishing, 2015.

Backurs, A., Indyk, P., Onak, K., Schieber, B., Vakilian, A., and Wagner, T. Scalable fair clustering. In *International Conference on Machine Learning*, pp. 405–413, 2019.

Baehrens, D., Schroeter, T., Harmeling, S., Kawanabe, M., Hansen, K., and MÃžller, K.-R. How to explain individual classification decisions. *Journal of Machine Learning Research*, 11(Jun): 1803–1831, 2010.

Becchetti, L., Bury, M., Cohen-Addad, V., Grandoni, F., and Schwiegelshohn, C. Oblivious dimension reduction for k-means: beyond subspaces and the Johnson-Lindenstrauss lemma. In *STOC*, 2019.

Bera, S., Chakrabarty, D., Flores, N., and Negahbani, M. Fair algorithms for clustering. In *Advances in Neural Information Processing Systems*, pp. 4955–4966, 2019.

Bertsimas, D., Orfanoudaki, A., and Wiberg, H. Interpretable clustering via optimal trees. *arXiv preprint arXiv:1812.00539*, 2018.

Bhaskara, A. and Rwanpathirana, A. K. Robust algorithms for online k-means clustering. In *Proceedings of the 31st International Conference on Algorithmic Learning Theory*, pp. 148–173, 2020.

Boutsidis, C., Drineas, P., and Mahoney, M. W. Unsupervised feature selection for the k-means clustering problem. In *NIPS*, pp. 153–161, 2009.

Cohen, M. B., Elder, S., Musco, C., Musco, C., and Persu, M. Dimensionality reduction for k-means clustering and low rank approximation. In *STOC*, 2015.

Cohen-Addad, V., Guedj, B., Kanade, V., and Rom, G. Online k-means clustering. *arXiv preprint arXiv:1909.06861*, 2019.

Dasgupta, S. The hardness of k-means clustering. In *Technical Report*. University of California, San Diego (Technical Report), 2008.

Deutch, D. and Frost, N. Constraints-based explanations of classifications. In *2019 IEEE 35th International Conference on Data Engineering (ICDE)*, pp. 530–541. IEEE, 2019.

Fraiman, R., Ghattas, B., and Svarc, M. Interpretable clustering using unsupervised binary trees. *Advances in Data Analysis and Classification*, 7(2):125–145, 2013.

Geurts, P., Touleimat, N., Dutreix, M., and d'Alché Buc, F. Inferring biological networks with output kernel trees. *BMC Bioinformatics*, 8(2):S4, 2007.

Ghattas, B., Michel, P., and Boyer, L. Clustering nominal data using unsupervised binary decision trees: Comparisons with the state of the art methods. *Pattern Recognition*, 67:177–185, 2017.

Hess, T. and Sabato, S. Sequential no-substitution k-median-clustering. *arXiv preprint arXiv:1905.12925*, 2019.

Huang, L., Jiang, S., and Vishnoi, N. Coresets for clustering with fairness constraints. In *Advances in Neural Information Processing Systems*, pp. 7587–7598, 2019.

Huleihel, W., Mazumdar, A., Médard, M., and Pal, S. Same-cluster querying for overlapping clusters. In *Advances in Neural Information Processing Systems*, pp. 10485–10495, 2019.

Kanungo, T., Mount, D. M., Netanyahu, N. S., Piatko, C. D., Silverman, R., and Wu, A. Y. A local search approximation algorithm for k-means clustering. In *Proceedings of the Eighteenth Annual Symposium on Computational Geometry*, pp. 10–18, 2002.

Kleindessner, M., Awasthi, P., and Morgenstern, J. Fair k-center clustering for data summarization. In *International Conference on Machine Learning*, pp. 3448–3457, 2019.

Liberty, E., Sriharsha, R., and Sviridenko, M. An algorithm for online k-means clustering. In *2016 Proceedings of the eighteenth workshop on algorithm engineering and experiments (ALENEX)*, pp. 81–89. SIAM, 2016.

Lipton, Z. C. The mythos of model interpretability. *Queue*, 16(3): 31–57, 2018.

Liu, B., Xia, Y., and Yu, P. S. Clustering via decision tree construction. In *Foundations and Advances in Data Mining*, pp. 97–124. Springer, 2005.

Lloyd, S. Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28(2):129–137, 1982.

Lundberg, S. M. and Lee, S.-I. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*, pp. 4765–4774, 2017.

MacQueen, J. Some methods for classification and analysis of multivariate observations. In *Proc. 5th Berkeley Symp. Mathematical Statist. Probability*, pp. 281–297, 1967.

Mahabadi, S. and Vakilian, A. (individual) fairness for k-clustering. *arXiv preprint arXiv:2002.06742*, 2020.

Makarychev, K., Makarychev, Y., and Razenshteyn, I. Performance of Johnson-Lindenstrauss transform for k-means and k-medians clustering. In *STOC*, 2019.

Mazumdar, A. and Saha, B. Clustering with noisy queries. In *Advances in Neural Information Processing Systems*, pp. 5788–5799, 2017.

Molnar, C. *Interpretable Machine Learning*. Lulu. com, 2019. `https://christophm.github.io/interpretable-ml-book/`.

Moshkovitz, M. Unexpected effects of online k-means clustering. *arXiv preprint arXiv:1908.06818*, 2019.

Murdoch, W. J., Singh, C., Kumbier, K., Abbasi-Asl, R., and Yu, B. Interpretable machine learning: definitions, methods, and applications. *arXiv preprint arXiv:1901.04592*, 2019.

Ostrovsky, R., Rabani, Y., Schulman, L. J., and Swamy, C. The effectiveness of Lloyd-type methods for the k-means problem. *Journal of the ACM*, 59(6):1–22, 2013.

Quinlan, J. R. Induction of decision trees. *Machine learning*, 1(1): 81–106, 1986.

Quinlan, J. R. *C4. 5: programs for machine learning*. Elsevier, 2014.

Ribeiro, M. T., Singh, S., and Guestrin, C. Why should I trust you?: Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1135–1144. ACM, 2016.

Rudin, C. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5):206–215, 2019.

Schmidt, M., Schwiegelshohn, C., and Sohler, C. Fair coresets and streaming algorithms for fair k-means. In *International Workshop on Approximation and Online Algorithms*, pp. 232–251. Springer, 2019.

Schütze, H., Manning, C. D., and Raghavan, P. Introduction to information retrieval. In *Proceedings of the International Communication of Association for Computing Machinery Conference*, volume 4, 2008.

Shalev-Shwartz, S. and Ben-David, S. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

Steinhaus, H. Sur la division des corp materiels en parties. *Bull. Acad. Polon. Sci*, 1:801–804, 1956.