arXiv:2002.12538v1 [cs.LG] 28 Feb 2020

Explainable k-Means and k-Medians Clustering

Sanjoy Dasgupta*
University of California, San Diego
dasgupta@eng.ucsd.edu

Nave Frost
Tel Aviv University
navefrost@mail.tau.ac.il

Michal Moshkovitz
University of California, San Diego
mmoshkovitz@eng.ucsd.edu

Cyrus Rashtchian
University of California, San Diego
crashtchian@eng.ucsd.edu

Abstract

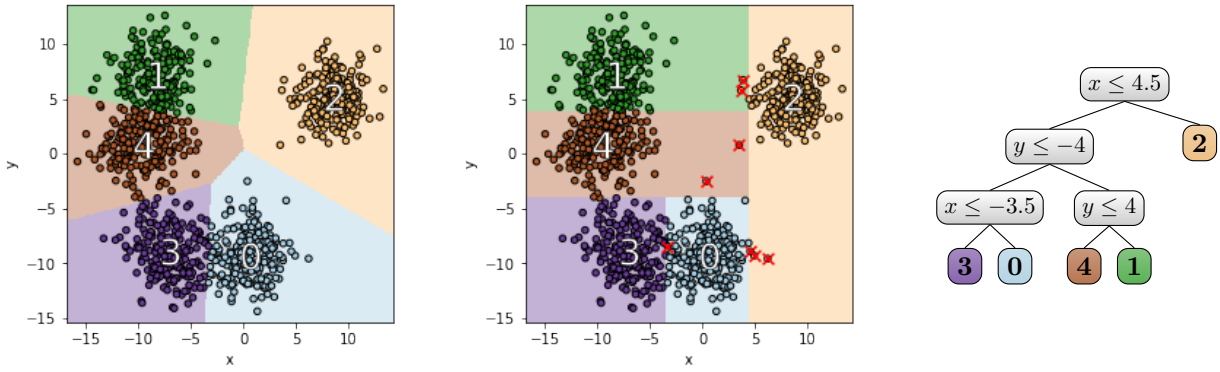
Clustering is a popular form of unsupervised learning for geometric data. Unfortunately, many clustering algorithms lead to cluster assignments that are hard to explain, partially because they depend on all the features of the data in a complicated way. To improve interpretability, we consider using 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: Must there exist a tree-induced clustering whose cost is comparable to that of the best unconstrained clustering, and if so, how can it be found? In terms of negative results, we show, first, that popular top-down decision tree algorithms may lead to clusterings with arbitrarily large cost, and second, that any tree-induced clustering must in general incur an $\Omega(\log k)$ approximation factor compared to the optimal clustering. On the positive side, we design an efficient algorithm that produces explainable clusters using a tree with k leaves. For two means/medians, we show that a single threshold cut suffices to achieve a constant factor approximation, and we give nearly-matching lower bounds. For general $k \geq 2$, our algorithm is 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 [26, 34, 36]. Prior work on AI explainability focuses on the interpretation of a black-box model, known as *post-modeling* explainability [8, 41]. While methods such as LIME [40] or Shapley explanations [29] 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 [42].

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

*Supported by NSF CCF-1813160.



(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. The benefit of explainability would be more apparent in higher dimensions.

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 [34, 36], and previous work on explainable clustering uses an unsupervised decision tree [27, 17, 18, 19, 11]. 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 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.

To analyze clustering quality, we consider the k-means and k-medians objectives [30, 46]. 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 [4, 23, 37, 45]. For example, Lloyd's algorithm alternates between determining the best center for the clusters and reassigning points to the closest center [28]. 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 in which the optimal clustering is very well approximated by one that is induced by a tree. But it is not clear whether this would in general be possible. 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 a highly constrained, 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, choice of a subset of features. As we will see, the kind of 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. In particular, 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.

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

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

1.2 Related work

It is NP-hard to find the optimal k-means clustering [3, 16] or even a very close approximation [6]. 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 [4, 1, 23, 37]. Several papers have considered the use of decision trees for explainable clustering [27, 17, 18, 19, 11]. 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 [13, 14]. 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 [9, 32]. 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 [7, 10, 21, 24, 31, 43], online clustering [12, 15, 20, 25, 35], and the use of same-cluster queries [2, 5, 22, 33].

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 $\mathcal{X} = \{\mathbf{x}^1, \ldots, \mathbf{x}^n\} \subseteq \mathbb{R}^d$ 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 $\boldsymbol{\mu}^1, \ldots, \boldsymbol{\mu}^k$, the aim of k-means is to find a clustering that minimizes the following objective

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

where $c_2(\mathbf{x}) = \arg\min_{\boldsymbol{\mu} \in \{\boldsymbol{\mu}^1, \ldots, \boldsymbol{\mu}^k\}} \|\boldsymbol{\mu} - \mathbf{x}\|_2$. Similarly, the goal of k-medians is to minimize

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

where $c_1(\mathbf{x}) = \arg\min_{\boldsymbol{\mu} \in \{\boldsymbol{\mu}^1, \ldots, \boldsymbol{\mu}^k\}} \|\boldsymbol{\mu} - \mathbf{x}\|_1$. We refer to the optimal set of k-means centers as opt^2 and the optimal k-medians centers as opt^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, opt, and $c(\mathbf{x})$.

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 $\text{cost}(T) \leq a \cdot \text{cost}(opt)$.

3 Motivating Examples

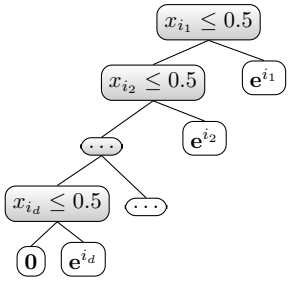
Using $k-1$ features may be necessary. We start with a simple but important bound showing that trees with depth less than k (or fewer than $k-1$ features) can be arbitrarily worse than the optimal clustering. Consider the data set consisting of the $k-1$ standard basis vectors $\mathbf{e}^1, \ldots, \mathbf{e}^{k-1} \in \mathbb{R}^{k-1}$ 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 2a 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 2a are necessary and sufficient for an optimal clustering. This example also shows that $\Theta(k)$ features are tight for feature selection [14] and provides a separation with feature extraction methods that use a linear map to only a logarithmic number of dimensions [9, 32].

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 [38, 39] 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.

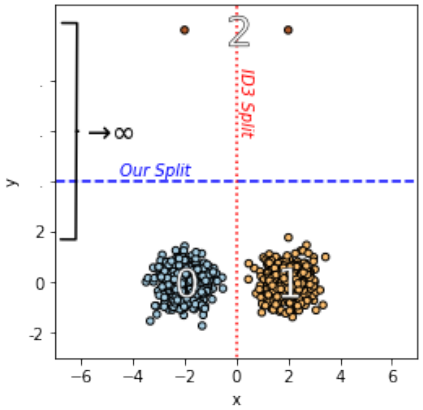
Figure 2b depicts a data set $\mathcal{X} \subseteq \mathbb{R}^2$ partitioned into three clusters $\mathcal{X} = \mathcal{X}_0 \cup \mathcal{X}_1 \cup \mathcal{X}_2$. We define two centers $\boldsymbol{\mu}^0 = (-2, 0)$ and $\boldsymbol{\mu}^1 = (2, 0)$ and for each $i \in \{0, 1\}$, we define $\mathcal{X}_i$ 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 special case of $k = 2$ clusters, and we study how well a single threshold cut can approximate the optimal partition into two clusters. Our algorithm will be the basis of our explainable k-medians/means result, and we achieve tighter bounds for two clusters.



(a) 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 $k-1$.



(b) The ID3 split results in a 3-means/medians clustering with arbitrarily worse cost than the optimal because it places the top two points in separate clusters. Our algorithm, described in Section 5, instead starts with the optimal first split.

Figure 2: Motivating examples showing that (a) threshold trees may need depth $k-1$ and (b) algorithms such as ID3 may perform badly.

4.1 Our 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 E.

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

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

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. We will think of these points as *mistakes*. More formally, a point $\mathbf{x}$ is a mistake for the threshold cut (i, θ) if

$$\text{sign}(\theta - x_i) \neq \text{sign}(\theta - c(\mathbf{x})_i),$$

where $\text{sign}(y) = 1 \Leftrightarrow y \geq 0$. We show that if the number of mistakes is large, so is the optimal cost. Intuitively, there are data sets where the optimal clustering may be very different from any axis-aligned clustering, but in these cases, the cost of moving the points cannot be too large compared to the optimal cost.

We note that it is possible to prove a slightly weaker bound by using the midpoint between the centers as the threshold. To improve the constant in the upper bound, we use a refined analysis using a matching result based on Hall's theorem. The proof of the theorem appears in Appendix C.

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.

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

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

The proof of these two lower bounds is in Appendix D.

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 or k-means [1], or from a domain-specific clustering heuristic. These centers may well 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.

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.

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.

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`$(\mathbf{x}^j, \boldsymbol{\mu}^{y^j}, i, \theta)$
  M $\leftarrow \{j \mid$ `mistake`$(\mathbf{x}^j, \boldsymbol{\mu}^{y^j}, i, \theta) = 1\}_{j=1}^m$
  L $\leftarrow \{j \mid (x_i^j \leq \theta) \wedge (j \notin \mathsf{M})\}_{j=1}^m$
  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

We analyze the approximation guarantees of IMM in Section 5.2 and the running time in Section 5.3.

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 1. *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*

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

A detailed proof is given in Appendix A. 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 2. *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*

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

2. *The k-means cost satisfies*

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

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 A.

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 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 A 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 Time analysis of the tree-building algorithm

While we have analyzed the approximation ratio of our algorithm, we have yet to justify its efficiency. Here 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 $O(kdn \log n)$.

5.4 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*

$$\text{cost}(T) \geq \Omega(\log k) \cdot \text{cost}(opt).$$

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 B.

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 B.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.

References

[1] Ankit Aggarwal, Amit Deshpande, and Ravi Kannan. Adaptive sampling for k-means clustering. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 15–28. Springer, 2009.

[2] Nir Ailon, Anup Bhattacharya, and Ragesh Jaiswal. Approximate correlation clustering using same-cluster queries. In *Latin American Symposium on Theoretical Informatics*, pages 14–27. Springer, 2018.

[3] Daniel Aloise, Amit Deshpande, Pierre Hansen, and Preyas Popat. NP-hardness of Euclidean sum-of-squares clustering. *Machine learning*, 75(2):245–248, 2009.

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

[5] Hassan Ashtiani, Shrinu Kushagra, and Shai Ben-David. Clustering with same-cluster queries. In *Advances in neural information processing systems*, pages 3216–3224, 2016.

[6] Pranjal Awasthi, Moses Charikar, Ravishankar Krishnaswamy, and Ali Kemal Sinop. The hardness of approximation of Euclidean k-means. In *31st International Symposium on Computational Geometry, SoCG 2015*, pages 754–767. Schloss Dagstuhl-Leibniz-Zentrum fur Informatik GmbH, Dagstuhl Publishing, 2015.

[7] Arturs Backurs, Piotr Indyk, Krzysztof Onak, Baruch Schieber, Ali Vakilian, and Tal Wagner. Scalable fair clustering. In *International Conference on Machine Learning*, pages 405–413, 2019.

[8] David Baehrens, Timon Schroeter, Stefan Harmeling, Motoaki Kawanabe, Katja Hansen, and Klaus-Robert MÃžller. How to explain individual classification decisions. *Journal of Machine Learning Research*, 11(Jun):1803–1831, 2010.

[9] Luca Becchetti, Marc Bury, Vincent Cohen-Addad, Fabrizio Grandoni, and Chris Schwiegelshohn. Oblivious dimension reduction for k-means: beyond subspaces and the Johnson-Lindenstrauss lemma. In *STOC*, 2019.

[10] Suman Bera, Deeparnab Chakrabarty, Nicolas Flores, and Maryam Negahbani. Fair algorithms for clustering. In *Advances in Neural Information Processing Systems*, pages 4955–4966, 2019.

[11] Dimitris Bertsimas, Agni Orfanoudaki, and Holly Wiberg. Interpretable clustering via optimal trees. *arXiv preprint arXiv:1812.00539*, 2018.

[12] Aditya Bhaskara and Aravinda Kanchana Rwanpathirana. Robust algorithms for online k-means clustering. In *Proceedings of the 31st International Conference on Algorithmic Learning Theory*, pages 148–173, 2020.

[13] Christos Boutsidis, Petros Drineas, and Michael W Mahoney. Unsupervised feature selection for the k-means clustering problem. In *NIPS*, pages 153–161, 2009.

[14] Michael B Cohen, Sam Elder, Cameron Musco, Christopher Musco, and Madalina Persu. Dimensionality reduction for k-means clustering and low rank approximation. In *STOC*, 2015.

[15] Vincent Cohen-Addad, Benjamin Guedj, Varun Kanade, and Guy Rom. Online k-means clustering. *arXiv preprint arXiv:1909.06861*, 2019.

[16] Sanjoy Dasgupta. The hardness of k-means clustering. University of California, San Diego (Technical Report), 2008.

[17] Ricardo Fraiman, Badih Ghattas, and Marcela Svarc. Interpretable clustering using unsupervised binary trees. *Advances in Data Analysis and Classification*, 7(2):125–145, 2013.

[18] Pierre Geurts, Nizar Touleimat, Marie Dutreix, and Florence d'Alché Buc. Inferring biological networks with output kernel trees. *BMC Bioinformatics*, 8(2):S4, 2007.

[19] Badih Ghattas, Pierre Michel, and Laurent Boyer. Clustering nominal data using unsupervised binary decision trees: Comparisons with the state of the art methods. *Pattern Recognition*, 67:177–185, 2017.

[20] Tom Hess and Sivan Sabato. Sequential no-substitution k-median-clustering. *arXiv preprint arXiv:1905.12925*, 2019.

[21] Lingxiao Huang, Shaofeng Jiang, and Nisheeth Vishnoi. Coresets for clustering with fairness constraints. In *Advances in Neural Information Processing Systems*, pages 7587–7598, 2019.

[22] Wasim Huleihel, Arya Mazumdar, Muriel Médard, and Soumyabrata Pal. Same-cluster querying for overlapping clusters. In *Advances in Neural Information Processing Systems*, pages 10485–10495, 2019.

[23] Tapas Kanungo, David M Mount, Nathan S Netanyahu, Christine D Piatko, Ruth Silverman, and Angela Y Wu. A local search approximation algorithm for k-means clustering. In *Proceedings of the Eighteenth Annual Symposium on Computational Geometry*, pages 10–18, 2002.

[24] Matthäus Kleindessner, Pranjal Awasthi, and Jamie Morgenstern. Fair k-center clustering for data summarization. In *International Conference on Machine Learning*, pages 3448–3457, 2019.

[25] Edo Liberty, Ram Sriharsha, and Maxim Sviridenko. An algorithm for online k-means clustering. In *2016 Proceedings of the eighteenth workshop on algorithm engineering and experiments (ALENEX)*, pages 81–89. SIAM, 2016.

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

[27] Bing Liu, Yiyuan Xia, and Philip S Yu. Clustering via decision tree construction. In *Foundations and Advances in Data Mining*, pages 97–124. Springer, 2005.

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

[29] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*, pages 4765–4774, 2017.

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

[31] Sepideh Mahabadi and Ali Vakilian. (individual) fairness for k-clustering. *arXiv preprint arXiv:2002.06742*, 2020.

[32] Konstantin Makarychev, Yury Makarychev, and Ilya Razenshteyn. Performance of Johnson-Lindenstrauss transform for k-means and k-medians clustering. In *STOC*, 2019.

[33] Arya Mazumdar and Barna Saha. Clustering with noisy queries. In *Advances in Neural Information Processing Systems*, pages 5788–5799, 2017.

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

[35] Michal Moshkovitz. Unexpected effects of online k-means clustering. *arXiv preprint arXiv:1908.06818*, 2019.

[36] W James Murdoch, Chandan Singh, Karl Kumbier, Reza Abbasi-Asl, and Bin Yu. Interpretable machine learning: definitions, methods, and applications. *arXiv preprint arXiv:1901.04592*, 2019.

[37] Rafail Ostrovsky, Yuval Rabani, Leonard J Schulman, and Chaitanya Swamy. The effectiveness of Lloyd-type methods for the k-means problem. *Journal of the ACM*, 59(6):1–22, 2013.

[38] J. Ross Quinlan. Induction of decision trees. *Machine learning*, 1(1):81–106, 1986.

[39] J Ross Quinlan. *C4. 5: programs for machine learning*. Elsevier, 2014.

[40] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 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*, pages 1135–1144. ACM, 2016.

[41] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Anchors: High-precision model-agnostic explanations. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

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

[43] Melanie Schmidt, Chris Schwiegelshohn, and Christian Sohler. Fair coresets and streaming algorithms for fair k-means. In *International Workshop on Approximation and Online Algorithms*, pages 232–251. Springer, 2019.

[44] Hinrich Schütze, Christopher D Manning, and Prabhakar Raghavan. Introduction to information retrieval. In *Proceedings of the International Communication of Association for Computing Machinery Conference*, volume 4, 2008.

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

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

A Upper Bound: Threshold Tree with k leaves

We prove Theorem 3 regarding the approximation ratio of the IMM algorithm. The proof proceeds in three main steps. First, we rewrite the cost of IMM in terms of the minimum number of mistakes made between the output clustering and the clustering based on the given centers. Second, we provide a lemma that relates the cost of any clustering to the number of changes required by a threshold clustering. Finally, we put these two together to show that the output cost is at most an $O(H)$ factor larger than the k-medians cost and at most an $O(Hk)$ factor larger than the k-means cost, respectively, where H is the depth of the IMM tree, and the cost is relative to $\text{cost}(\boldsymbol{\mu}^1, \ldots, \boldsymbol{\mu}^k)$. In particular, when IMM starts with a constant factor approximation to the optimal centers, it achieves cost $O(H \cdot \text{cost}(opt^1))$ for k-medians or $O(Hk \cdot \text{cost}(opt^2))$ for k-means.

Notation and Preliminaries. Let T be the IMM tree that is built using the given centers $\boldsymbol{\mu}^1, \ldots, \boldsymbol{\mu}^k$. Each node u in the tree corresponds to a value $\theta_u \in \mathbb{R}$ and a coordinate $i \in [d]$. The tree T defines a partition of $\mathcal{X}$ into k clusters $\widehat{C}_1, \ldots, \widehat{C}_k$ based on the points that reach the k leaves in T, where we index the clusters so that leaf j contains the centers $\boldsymbol{\mu}^j$ and $\widehat{\boldsymbol{\mu}}^j$, where $\widehat{\boldsymbol{\mu}}^j$ is the mean of $\widehat{C}_j$ for k-means and the median of $\widehat{C}_j$ for k-medians. This provides a bijection between old and new centers (and clusters). Recall that the map $c : \mathcal{X} \to \{\boldsymbol{\mu}^1, \ldots, \boldsymbol{\mu}^k\}$ associates each point to its nearest center (i.e., $c(\mathbf{x})$ corresponds to the cluster assignment given by the centers $\{\boldsymbol{\mu}^1, \ldots, \boldsymbol{\mu}^k\}$).

For a node $u \in T$, we let $\mathcal{X}_u$ denote the surviving data set vectors at node $u \in T$ based on the thresholds from the root to u. We also define $J_u \subseteq [k]$ be the set of surviving centers at node u from the set $\{\boldsymbol{\mu}^1, \ldots, \boldsymbol{\mu}^k\}$, where these centers satisfy the thresholds from the root to u. Define $\boldsymbol{\mu}^{L,u}$ and $\boldsymbol{\mu}^{R,u}$ to be the maximal (smallest and largest) coordinate-wise values of the centers in J_u, that is, for $i \in [d]$, we set

$$\mu_i^{L,u} = \min_{j \in J_u} \mu_i^j, \qquad \text{and} \qquad \mu_i^{R,u} = \max_{j \in J_u} \mu_i^j.$$

In other words, using the previous notation, we have that

$$\text{diam}_1(B(u)) = \|\mu_i^{L,u} - \mu_i^{R,u}\|_1 \qquad \text{and} \qquad \text{diam}_2^2(B(u)) = \|\mu_i^{L,u} - \mu_i^{R,u}\|_2^2,$$

where we recall that $B(u) = \{\boldsymbol{\mu}^1, \ldots, \boldsymbol{\mu}^k\} \cap \text{cell}(u)$.

Recall that t_u for node $u \in T$ denotes the number of *mistakes* incurred during the threshold cut defined by u, where a point $\mathbf{x}$ is a mistake at node u if x reaches u, it was not a mistake before, and exactly one of the following two events occurs:

$$\{c(\mathbf{x})_i \le \theta_u \ \text{ and } \ x_i > \theta_u\} \qquad \text{or} \qquad \{c(\mathbf{x})_i > \theta_u \ \text{ and } \ x_i \le \theta_u\}.$$

Let $\mathcal{X} = \mathcal{X}^{\mathsf{cor}} \cup \mathcal{X}^{\mathsf{mis}}$ be a partition of the input data set into two parts, where $\mathbf{x}$ is in $\mathcal{X}^{\mathsf{cor}}$ if it reaches the same leaf node in T as its center $c(\mathbf{x})$, and otherwise, $\mathbf{x}$ is in $\mathcal{X}^{\mathsf{mis}}$. In other words, $\mathcal{X}^{\mathsf{mis}}$ contains all points $\mathbf{x} \in \mathcal{X}$ that are a mistake at any node u in T, and the rest of the points are in $\mathcal{X}^{\mathsf{cor}}$.

We also need a standard inequality to analyze the k-means cost.

Claim 1. *For any $a_1, \ldots, a_m \in \mathbb{R}$, it holds that $\sum_{i=1}^k a_i^2 \ge \frac{1}{k}\left(\sum_{i=1}^k a_i\right)^2$.*

Proof. Denote by a the vector $(a_1, \ldots, a_m)$ and by b the vector $(1/\sqrt{k}, \ldots, 1/\sqrt{k})$. By the CauchySchwarz inequality $\frac{1}{k}\left(\sum_{i=1}^k a_i\right)^2 = \langle a, b\rangle^2 \le \sum_{i=1}^k a_i^2$ □

We also need two facts, which state the optimal center for a cluster corresponds to mean or median of the points in the cluster, respectively.

Fact 1. *For any set of points $S = \{\mathbf{x}^1, \ldots, \mathbf{x}^n\} \subseteq \mathbb{R}^d$, the optimal center under the ℓ_2^2 cost is the mean $\boldsymbol{\mu} = \frac{1}{n}\sum_{\mathbf{x}\in S}\mathbf{x}$.*

Fact 2. *For any set of points $S = \{\mathbf{x}^1, \ldots, \mathbf{x}^n\} \subseteq \mathbb{R}^d$, the optimal center $\boldsymbol{\mu}$ under the ℓ_1 cost is the median $\mu_i = \mathsf{median}(x_i^1, \ldots, x_i^n)$ for $i \in [d]$.*

The proofs of these facts can be found in standard texts [44].

A.1 Proof of Theorem 3

To prove the theorem, we state two lemmas that aid in analyzing the cost of the given clustering versus the IMM clustering. These lemmas are simply a restatement of Lemmas 1 and 2, this time using the new notation. The theorem will follow from these lemmas, and we will prove the lemmas in the proceeding subsections. We start with the lemma relating the number of mistakes t_u at each node u and the distance between $\boldsymbol{\mu}^{L,u}$ and $\boldsymbol{\mu}^{R,u}$ to the cost incurred by the given centers.

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

1. *The k-medians cost of the IMM tree satisfies*

$$\mathrm{cost}(T) \leq \mathrm{cost}(\boldsymbol{\mu}^1, \ldots, \boldsymbol{\mu}^k) + \sum_{u\in T} t_u \|\boldsymbol{\mu}^{L,u} - \boldsymbol{\mu}^{R,u}\|_1$$

2. *The k-means cost of the IMM tree satisfies*

$$\mathrm{cost}(T) \leq 2 \cdot \mathrm{cost}(\boldsymbol{\mu}^1, \ldots, \boldsymbol{\mu}^k) + 2 \cdot \sum_{u\in T} t_u \|\boldsymbol{\mu}^{L,u} - \boldsymbol{\mu}^{R,u}\|_2^2$$

We next bound the cost of the given centers in the terms of the number of mistakes in the tree. The key idea is that if there must be many mistakes at each node, then the cost of the given centers must actually be fairly large.

Lemma 4. *If IMM takes centers $\boldsymbol{\mu}^1, \ldots, \boldsymbol{\mu}^k$ and returns a tree T of depth H that incurs t_u mistakes at node $u \in T$, then*

1. *The k-medians cost satisfies*

$$\sum_{u\in T} t_u \|\boldsymbol{\mu}^{L,u} - \boldsymbol{\mu}^{R,u}\|_1 \leq 2H \cdot \mathrm{cost}(\boldsymbol{\mu}^1, \ldots, \boldsymbol{\mu}^k).$$

2. *The k-means cost satisfies*

$$\sum_{u\in T} t_u \|\boldsymbol{\mu}^{L,u} - \boldsymbol{\mu}^{R,u}\|_2^2 \leq 4Hk \cdot \mathrm{cost}(\boldsymbol{\mu}^1, \ldots, \boldsymbol{\mu}^k).$$

Combining these two lemmas immediately implies Theorem 3.

A.2 Proof of Lemma 3

We begin with the k-medians proof (the k-means proof will be similar). Notice that the cost can only increase when measuring the distance to the (suboptimal) center $\boldsymbol{\mu}^j$ instead of the (optimal) center $\widehat{\boldsymbol{\mu}}^j$ for cluster $\widehat{C}_j$, and hence,

$$\text{cost}(T) = \sum_{j=1}^{k} \sum_{\mathbf{x} \in \widehat{C}_j} \|\mathbf{x} - \widehat{\boldsymbol{\mu}}^j\|_1 \leq \sum_{j=1}^{k} \sum_{\mathbf{x} \in \widehat{C}_j} \|\mathbf{x} - \boldsymbol{\mu}^j\|_1.$$

We can rewrite this sum using the partition $\mathcal{X}^{\mathsf{cor}}$ and $\mathcal{X}^{\mathsf{mis}}$ of $\mathcal{X}$, using the fact that whenever $\mathbf{x} \in \mathcal{X}^{\mathsf{cor}}$, then the distance is computed with respect to the true center $c(\mathbf{x})$,

$$\begin{aligned}
\sum_{j=1}^{k} \sum_{\mathbf{x} \in \widehat{C}_j} \|\mathbf{x} - \boldsymbol{\mu}^j\|_1 &= \sum_{j=1}^{k} \sum_{\mathbf{x} \in \mathcal{X}^{\mathsf{cor}} \cap \widehat{C}_j} \|\mathbf{x} - \boldsymbol{\mu}^j\|_1 + \sum_{j=1}^{k} \sum_{\mathbf{x} \in \mathcal{X}^{\mathsf{mis}} \cap \widehat{C}_j} \|\mathbf{x} - \boldsymbol{\mu}^j\|_1 \\
&= \sum_{\mathbf{x} \in \mathcal{X}^{\mathsf{cor}}} \|\mathbf{x} - c(\mathbf{x})\|_1 + \sum_{j=1}^{k} \sum_{\mathbf{x} \in \mathcal{X}^{\mathsf{mis}} \cap \widehat{C}_j} \|\mathbf{x} - \boldsymbol{\mu}^j\|_1
\end{aligned}$$

Starting with the above cost bound, and using the triangle inequality, we see

$$\begin{aligned}
\text{cost}(T) &\leq \sum_{\mathbf{x} \in \mathcal{X}^{\mathsf{cor}}} \|\mathbf{x} - c(\mathbf{x})\|_1 + \sum_{j=1}^{k} \sum_{\mathbf{x} \in \mathcal{X}^{\mathsf{mis}} \cap \widehat{C}_j} \|\mathbf{x} - \boldsymbol{\mu}^j\|_1 \\
&\leq \sum_{\mathbf{x} \in \mathcal{X}^{\mathsf{cor}}} \|\mathbf{x} - c(\mathbf{x})\|_1 + \sum_{j=1}^{k} \sum_{\mathbf{x} \in \mathcal{X}^{\mathsf{mis}} \cap \widehat{C}_j} (\|\mathbf{x} - c(\mathbf{x})\|_1 + \|c(\mathbf{x}) - \boldsymbol{\mu}^j\|_1) \\
&= \text{cost}(\boldsymbol{\mu}^1, \ldots, \boldsymbol{\mu}^k) + \sum_{j=1}^{k} \sum_{\mathbf{x} \in \mathcal{X}^{\mathsf{mis}} \cap \widehat{C}_j} \|c(\mathbf{x}) - \boldsymbol{\mu}^j\|_1
\end{aligned}$$

To control the second term in the final line, we must bound the cost of the mistakes. We decompose $\mathcal{X}^{\mathsf{mis}}$ based on the node u where $\mathbf{x} \in \mathcal{X}^{\mathsf{mis}}$ is first separated from its true center $c(\mathbf{x})$ due to the threshold at node u. To this end, consider some point $\mathbf{x} \in \mathcal{X}^{\mathsf{mis}} \cap \widehat{C}_j$, where its distance is measured to the incorrect center $\boldsymbol{\mu}^j \neq c(\mathbf{x})$. Both centers $c(\mathbf{x})$ and $\boldsymbol{\mu}^j$ have survived until node u in the threshold tree T, and hence, both vectors are part of the definitions of $\boldsymbol{\mu}^{L,u}$ and $\boldsymbol{\mu}^{R,u}$. In particular, we can use the upper bound

$$\|c(\mathbf{x}) - \boldsymbol{\mu}^j\|_1 \leq \|\boldsymbol{\mu}^{L,u} - \boldsymbol{\mu}^{R,u}\|_1.$$

There are t_u points in $\mathcal{X}^{\mathsf{mis}}$ caused by the threshold at node u, and we have that

$$\sum_{j=1}^{k} \sum_{\mathbf{x} \in \mathcal{X}^{\mathsf{mis}} \cap \widehat{C}_j} \|c(\mathbf{x}) - \boldsymbol{\mu}^j\|_1 \leq \sum_{u \in T} t_u \cdot \|\boldsymbol{\mu}^{L,u} - \boldsymbol{\mu}^{R,u}\|_1.$$

Therefore, we have, as desired

$$\begin{aligned}
\text{cost}(T) &\leq \text{cost}(\boldsymbol{\mu}^1, \ldots, \boldsymbol{\mu}^k) + \sum_{j=1}^{k} \sum_{\mathbf{x} \in \mathcal{X}^{\mathsf{mis}} \cap \widehat{C}_j} \|\mathbf{x} - \boldsymbol{\mu}^j\|_1 \\
&\leq \text{cost}(\boldsymbol{\mu}^1, \ldots, \boldsymbol{\mu}^k) + \sum_{u \in T} t_u \|\boldsymbol{\mu}^{L,u} - \boldsymbol{\mu}^{R,u}\|_1.
\end{aligned}$$

The analysis for k-means is essentially same, except we incur a factor of two by using Claim 1 instead of the triangle inequality:

$$\begin{aligned}
\text{cost}(T) &\leq \sum_{\mathbf{x}\in\mathcal{X}^{\mathsf{cor}}} \|\mathbf{x}-c(\mathbf{x})\|_2^2 + 2\sum_{j=1}^{k}\sum_{\mathbf{x}\in\mathcal{X}^{\mathsf{mis}}\cap\widehat{C}_j} (\|\mathbf{x}-c(\mathbf{x})\|_2^2 + \|c(\mathbf{x})-\boldsymbol{\mu}^j\|_2^2) \\
&\leq 2\cdot\text{cost}(\boldsymbol{\mu}^1,\ldots,\boldsymbol{\mu}^k) + 2\cdot\sum_{j=1}^{k}\sum_{\mathbf{x}\in\mathcal{X}^{\mathsf{mis}}\cap\widehat{C}_j} \|c(\mathbf{x})-\boldsymbol{\mu}^j\|_2^2 \\
&\leq 2\cdot\text{cost}(\boldsymbol{\mu}^1,\ldots,\boldsymbol{\mu}^k) + 2\cdot\sum_{u\in T} t_u\|\boldsymbol{\mu}^{L,u}-\boldsymbol{\mu}^{R,u}\|_2^2
\end{aligned}$$

A.3 Proof of Lemma 4

To prove this lemma, we bound the cost at each node u of tree in terms of the mistakes made at this node. For this lemma, we define $\mathcal{X}_u^{\mathsf{cor}}$ to be the set of points in $\mathcal{X}$ that reach node u in T along with their center $c(\mathbf{x})$. We note that $\mathcal{X}_u^{\mathsf{cor}}$ differs from $\mathcal{X}^{\mathsf{cor}}\cap\mathcal{X}_u$ because a point $\mathbf{x}\in\mathcal{X}_u^{\mathsf{cor}}$ may not make it to $\mathcal{X}^{\mathsf{cor}}$ if there is a mistake later on (i.e., $\mathcal{X}^{\mathsf{cor}}$ is the union of $\mathcal{X}_u^{\mathsf{cor}}$ only over leaf nodes).

Lemma 5. *For any node $u\in T$, we have that*

$$\sum_{\mathbf{x}\in\mathcal{X}_u^{\mathsf{cor}}} \|\mathbf{x}-c(\mathbf{x})\|_1 \geq \frac{t_u}{2}\cdot\|\boldsymbol{\mu}^{L,u}-\boldsymbol{\mu}^{R,u}\|_1.$$

and

$$\sum_{\mathbf{x}\in\mathcal{X}_u^{\mathsf{cor}}} \|\mathbf{x}-c(\mathbf{x})\|_2^2 \geq \frac{t_u}{4k}\cdot\|\boldsymbol{\mu}^{L,u}-\boldsymbol{\mu}^{R,u}\|_2^2.$$

Proof. Fix a coordinate $i\in[d]$ and a node $u\in T$. To simplify notation, we let $z_1\leq\cdots\leq z_{k'}$ denote the *sorted* values of ith coordinate of the $k'\leq k$ centers that survive until node u (so that $z_1=\mu_i^{L,u}$ and $z_{k'}=\mu_i^{R,u}$). Observe that for each $\mathbf{x}\in\mathcal{X}_u^{\mathsf{cor}}$, the center $c(\mathbf{x})$ must have survived until node u, and hence, $c(\mathbf{x})_i$ equals one of the values z_j for $j\in[k']$.

We need one crucial definition for the proof, which allows us to relate the cost in coordinate i to the distances between z_1 and $z_{k'}$. For consecutive values $(j,j+1)$, we say that the pair $(j,j+1)$ is *covered* by $\mathbf{x}$ if either

- The segment $[z_j,\frac{z_j+z_{j+1}}{2})$ is contained in the segment $[x_i,c(\mathbf{x})_i]$, or
- The segment $[\frac{z_j+z_{j+1}}{2},z_{j+1})$ is contained in the segment $[x_i,c(\mathbf{x})_i]$.

We prove the following claim, which will allow us to relate the cost in the ith coordinate to value $z_{k'}-z_1$ by decomposing this value into the distance between consecutive centers.

Claim 2. *For each $j=1,2,\ldots,k'-1$, the pair $(j,j+1)$ is covered by at least t_u points $\mathbf{x}\in\mathcal{X}_u^{\mathsf{cor}}$.*

Proof. Suppose for contradiction that this does not hold. We argue that we can find a threshold value for coordinate i that makes fewer than t_u mistakes. To see this, assume that $(j,j+1)$ is covered by fewer than t_u points $\mathbf{x}\in\mathcal{X}_u$. In particular, setting the threshold to be $\frac{z_j+z_{j+1}}{2}$ separates fewer than t_u points $\mathbf{x}$ from their centers $c(\mathbf{x})$. This implies that there are fewer than t_u mistakes at node u, which is a contradiction because the IMM algorithm chooses the coordinate and threshold pair that minimizes the number of mistakes. □

Now this claim suffices to prove Lemma 5. The only challenge is that we must string together the covering points $\mathbf{x}$ to get a bound on $z_{k'} - z_1$.

We start with the k-medians proof. Using the above claim, we can lower bound the contribution of coordinate i to the cost of the given centers. Notice that the values $z_1 \leq \cdots \leq z_{k'}$ partition the interval between $z_1 = \mu_i^{L,u}$ and $z_{k'} = \mu_i^{R,u}$. Thus, each time $\mathbf{x}$ covers a pair $(j, j+1)$, there must be a contribution of $\frac{z_{j+1}-z_j}{2}$ to the cost $|x_i - c(\mathbf{x})_i|$. Because each pair is covered at least t_u times by Claim 2, we conclude that

$$\sum_{\mathbf{x}\in\mathcal{X}_u^{\mathsf{cor}}} |x_i - c(\mathbf{x})_i| \geq t_u \sum_{j=1}^{k'-1} \left(\frac{z_{j+1} - z_j}{2}\right) = \frac{t_u}{2}(z_{k'} - z_1).$$

To relate the bound to $\boldsymbol{\mu}^{L,u}$ and $\boldsymbol{\mu}^{R,u}$, we note that the above argument holds for each coordinate $i \in [d]$, and we have that

$$\sum_{\mathbf{x}\in\mathcal{X}_u^{\mathsf{cor}}} \|\mathbf{x} - c(\mathbf{x})\|_1 = \sum_{i\in[d]} \sum_{\mathbf{x}\in\mathcal{X}_u^{\mathsf{cor}}} |\mathbf{x}_i - c(\mathbf{x})_i| \geq \frac{t_u}{2} \cdot \|\boldsymbol{\mu}^{L,u} - \boldsymbol{\mu}^{R,u}\|_1.$$

For the k-means proof, we apply the same argument as above, this time using Claim 1 to bound the sum of squared values as

$$\sum_{\mathbf{x}\in\mathcal{X}_u^{\mathsf{cor}}} |x_i - c(\mathbf{x})_i|^2 \geq t_u \sum_{j=1}^{k'-1} \left(\frac{z_{j+1} - z_j}{2}\right)^2 \geq \frac{t_u}{k} \left(\sum_{j=1}^{k'-1} \left(\frac{z_{j+1} - z_j}{2}\right)\right)^2 = \frac{t_u}{4k}(z_{k'} - z_1)^2,$$

and therefore, summing over coordinates $i \in [d]$, we have

$$\sum_{\mathbf{x}\in\mathcal{X}_u^{\mathsf{cor}}} \|\mathbf{x} - c(\mathbf{x})\|_2^2 = \sum_{i\in[d]} \sum_{\mathbf{x}\in\mathcal{X}_u^{\mathsf{cor}}} |x_i - c(\mathbf{x})_i|^2 \geq \frac{t_u}{4k} \cdot \|\boldsymbol{\mu}^{L,u} - \boldsymbol{\mu}^{R,u}\|_2^2.$$

□

Proof of Lemma 4. We start with the k-medians proof. The factor of H arises because the same points $\mathbf{x} \in \mathcal{X}$ can appear in at most H sets $\mathcal{X}_u^{\mathsf{cor}}$ because H is the depth of the tree. More precisely, using Lemma 5 for each node u, we have that

$$H \cdot \mathrm{cost}(\boldsymbol{\mu}^1, \ldots, \boldsymbol{\mu}^k) \geq \sum_{u\in T} \sum_{\mathbf{x}\in\mathcal{X}_u^{\mathsf{cor}}} \|\mathbf{x} - c(\mathbf{x})\|_1 \geq \sum_{u\in T} \frac{t_u}{2} \|\boldsymbol{\mu}^{L,u} - \boldsymbol{\mu}^{R,u}\|_1.$$

Applying the same steps for the k-means cost, we have that

$$H \cdot \mathrm{cost}(\boldsymbol{\mu}^1, \ldots, \boldsymbol{\mu}^k) \geq \sum_{u\in T} \sum_{\mathbf{x}\in\mathcal{X}_u^{\mathsf{cor}}} \|\mathbf{x} - c(\mathbf{x})\|_2^2 \geq \sum_{u\in T} \frac{t_u}{4k} \|\boldsymbol{\mu}^{L,u} - \boldsymbol{\mu}^{R,u}\|_2^2.$$

□

B Lower Bound: Threshold Tree with k-leaves

In this section we show that any threshold tree with k leaves must be an $\Omega(\log k)$-approximation, under the k-means and k-medians cost. We will show a data set that will cause many mistakes. This data set consists of k clusters where any two clusters are very far from each other while inside any cluster the points differ by at most two features. Each cluster is created by first taking a codeword and then changing one feature at a time to 0. The consequence of this process is that for every feature there are many points that globally are very different yet locally all equal to 0.

The proof of the lower bound has a few steps:

1. In Section B.1 we show that there is a code such that (i) every two points are far apart, and (ii) when inspecting any $O(\log k)$ features, many codewords are consistent with this local view. From thiscode we construct our data set with dk points and $\text{cost}(opt) = O(dk)$.

2. In Section B.2 we prove that the clusters induced by the threshold tree T are similar to the original clusters, except for at most k points in each cluster. These points will cause $\text{cost}(T)$ to be large.

3. In Section B.3 we uncover a few properties of any threshold tree created by an $O(\log k)$-approximation algorithm: up until level $O(\log k)$ the tree has to be complete and no feature is used more than once.

4. In Section B.4 we put together all the claims and show that each level causes $\Omega(k \log k)$ mistakes, each with a cost of $\Omega(d)$, thus $\text{cost}(T) = \Omega(kd \log k)$ which proves the lower bound of $\Omega(\log k)$-approximation.

Data set construction. We first take k *codewords* $\mathbf{v}^1, \ldots, \mathbf{v}^k \in \{+1, -1\}^d$ that have the properties described in Claim 3. From each codeword $\mathbf{v}$ we create d data points, $\mathcal{X}^{\mathbf{v}}$, each time by changing exactly one feature to 0. In total we have dk points in the data set, $\mathcal{X} = \cup_i \mathcal{X}^{\mathbf{v}^i}$. The cost of the clustering that cluster together all points that belong to the same vector $\mathbf{v}^i$ is $O(dk)$, as the cost of each point is $\Theta(1)$. Thus, $\text{cost}(opt) \leq O(dk)$.

B.1 The data set

Proposition 1 (Hoeffdings inequality)**.** *Let $X_1, ..., X_n$ be independent random variables, where for each i, $X_i \in [0, 1]$. Define the random variable $X = \sum_{i=1}^n X_i$. Then, for any $t \geq 0$,*

$$\Pr(|X - \mathbb{E}[X]| \geq t) \leq 2e^{-\frac{2t^2}{n}}.$$

Claim 3. *For any $k \geq 3$, there are k points $C \subseteq \{\pm 1\}^d$ that have the following properties for any $\epsilon \geq \frac{\ln(k)}{\sqrt{k}}$:*

1. *$d = k^3$*

2. *for every $\mathbf{c} \neq \mathbf{c}' \in C$ their distance is linear, i.e., $|\{i : c_i \neq c'_i\}| \geq d/4$.*

3. *for every $\ell \leq \frac{\ln(k)}{50}$ indexes in $[d]$, and every assignment to these indexes, the number of points in C that has these assignment is at least $k(1/2^\ell - \epsilon)$*

Proof. Take k random points in $\{\pm 1\}^d$. We will show that the probability that all properties hold is bigger than 0 and this will prove our claim using the probabilistic method.

To prove the second property, we use Hoeffding's inequality and union bound. We can bound the probability that any two points in C agree by more than $3d/4$ coordinates by $2k^2 e^{-d/8} < 1 - e^{-1}$ for $k \geq 3$.

To prove the third property we again use Hoeffding's inequality and union bound. This time though we have k random variables, one for each point. There are $\binom{d}{\ell}$ possible ℓ coordinates, and there are 2^ℓ possible assignments to these coordinates. For specific ℓ coordinates and an assignment to these coordinates the expected number of points in C that has the specific assignment is $k/2^\ell$. By Hoeffding's inequality, the probability that we deviate by ϵk is less than $e^{-\epsilon^2 k}$. The probability that the last property does not hold is bounded by

$$\binom{d}{\ell} 2^{\ell+1} e^{-2\epsilon^2 k} \leq e^{\ell \ln d + 2\ell + 1 - 2\epsilon^2 k}.$$

Thus for $\epsilon \geq \frac{\ln(k)}{\sqrt{k}}$, the last term is smaller than e^{-1}.

□

B.2 The cluster created by a threshold tree

Claim 4. *For any threshold tree T with at most k leaves, and for any codeword $\mathbf{v}$, the leaf containing $\mathbf{v}$ also contains at least $d-k$ points of $\mathcal{X}^{\mathbf{v}}$.*

Proof. There are at most $k \geq 3$ leaves in T, thus in the root to leaf path of the codeword v there are at most $k-1$ features. Hence, all data points in $\mathcal{X}^{\mathbf{v}}$ that agree on this features must reach the same leaf and be in the same cluster. There are at least $d-k$ such points in $\mathcal{X}^{\mathbf{v}}$. □

Claim 5. *If there are α points from $\mathcal{X}^{\mathbf{v}^1}$ and β points from $\mathcal{X}^{\mathbf{v}^2}$, $\mathbf{v}^1 \neq \mathbf{v}^2$, that are in the same cluster in T, then their contribution to* $\text{cost}(T)$ *is at least $\frac{1}{4}\min(\alpha,\beta)d$. The claim holds both under the ℓ_1 cost and the ℓ_2 squared cost.*

Proof. Denote the center that contains α points from $\mathcal{X}^{\mathbf{v}^1}$ and β points from $\mathcal{X}^{\mathbf{v}^2}$ by $\boldsymbol{\mu}$. Without loss of generality $\alpha \leq \beta$. We can disjointly match α points from the two different clusters $(\mathbf{x}^1,\mathbf{y}^1),\ldots,(\mathbf{x}^\alpha,\mathbf{y}^\alpha)$, which means that their contribution to $\text{cost}(T)$ is at least

$$\sum_{j=1}^{\alpha} \left\|\mathbf{x}^j-\boldsymbol{\mu}\right\|_2^2 + \left\|\boldsymbol{\mu}-\mathbf{y}^j\right\|_2^2 \geq \sum_{j=1}^{\alpha} \frac{1}{2}\left\|\mathbf{x}^j-\mathbf{y}^j\right\|_2^2 \geq \frac{1}{2}\cdot\alpha\cdot\left(\frac{d}{4}-2\right)\cdot 4 \geq \frac{d\alpha}{4},$$

where the first inequality follows Claim 1, the second inequality follows from Claim 3 and the fact that if two codewords are different by at least $d/4$ features, then the points differ by at least $d/4-2$ features, each contributing a cost of 4, the third inequality follows from the fact that $d=k^3\geq 16$ for $k\geq 3$. Similarly for the ℓ_1 cost

$$\sum_{j=1}^{\alpha} \left\|\mathbf{x}^j-\boldsymbol{\mu}\right\|_1 + \sum_{j=1}^{\alpha} \left\|\boldsymbol{\mu}-\mathbf{y}^j\right\|_1 \geq \left\|\mathbf{x}^j-\mathbf{y}^j\right\|_1 \geq \alpha\cdot\left(\frac{d}{4}-2\right)\cdot 2 \geq \frac{d\alpha}{4}.$$

□

B.3 The threshold tree

The next two claims prove that if a feature is used twice or the tree is not complete until level $\frac{\ln(k)}{50}$, then the clustering tree T cannot be an $O(\log k)$-approximation because it shows that $\text{cost}(T) \gtrapprox d^2 \gg \log k \cdot \text{cost}(opt)$.

Claim 6. *Fix a threshold tree T with $k \geq 3$ leaves. If there is a feature that is used twice on the same root-to-leaf path in T, then*

$$\text{cost}(T) \geq \frac{d(d-k)}{4}.$$

Proof. The proof is composed of two steps, first we show that if a feature is used twice then there is leaf that it is unreachable by a codeword. Then we will show that this implies that two codewords share the same cluster, and thus $\text{cost}(T)$ is high.

Assume that the there are two nodes in T, both of them use the same feature i, one with threshold θ and the other with threshold θ'. If $\theta=\theta'$ then there is a leaf that is not reachable. Otherwise, the two thresholds divide the line into three parts, and since the codewords have only two values, there is a leaf unreachable by any codeword. Summing up these two cases, there is a leaf that is not reached by any codeword.

From the pigeonhole principle there are two codewords that share the same cluster which is a contradiction using Claims 4 and 5. □

Claim 7. *If threshold T contains a leaf at depth less than $\frac{\ln k}{50}$, then*

$$\text{cost}(T) \geq \frac{d(d-k)}{4}.$$

The claim is true both under the k-means and the k-medians cost.

Proof. Assume T is not complete until level $\frac{\log k}{50}$. So there is a leaf at a level smaller than $\frac{\log k}{50}$. By the construction of the data set, there are at least $(\frac{1}{2^\ell} - \epsilon)k > 1$ codewords that reach this leaf. The claim follows from Claims 4 and 5. □

B.4 Proof of Theorem 4

Assume by contradiction that T is an $O(\log k)$-approximation. From Claim 4 we deduce that for each codeword $\mathbf{v}$, at least $d-k$ points from $\mathcal{X}^{\mathbf{v}}$ will be in the same cluster. From Claim 5 and the assumption that T is $O(\log k)$-approximation we get that each $d-k$ such points must be in its own cluster, this cluster will be called the *main cluster* of $\mathcal{X}^{\mathbf{v}}$.

The only values that features can get in the data set are $+1, -1$ or 0. Thus, we can assume, without loss of generality, that each threshold is either 0.5 or -0.5. Focus on some node in T at level ℓ with feature i and threshold θ. If $\theta = 0.5$, then for all codewords $\mathbf{v}$ with $v_i = 1$ the point in $\mathcal{X}^{\mathbf{v}}$ with v_i will be separated for its main cluster. From the construction of the data set, there are at least $(\frac{1}{2^\ell} - \epsilon)k$ such points. Similarly for $\theta = -0.5$, we can show that there are at least $(\frac{1}{2^\ell} - \epsilon)k$ such points. From Claim 6, we deduce that these mistakes are disjoint.

Applying Claim 7, there are $2^{\ell-1}$ nodes at each level up until level $\frac{\log k}{50}$. Hence, total number of mistakes, i.e., points that will not go with their codeword, can be lower bounded by the following using $\epsilon = \frac{\ln(k)}{\sqrt{k}}$ and large enough k:

$$\sum_{\ell=1}^{\frac{\log k}{50}} 2^{\ell-1} \left(\frac{1}{2^\ell} - \epsilon\right) k \geq \frac{k \log k}{200}.$$

Thus, from Claim 4 we can lower bound the cost of T:

$$\text{cost}(T) \geq \frac{kd \log k}{200} = \Omega(\log k)\text{cost}(opt)$$

B.5 IMM Upper Bound for this dataset

We sketch the proof that the IMM algorithm produces a tree of depth $O(\log k)$ for the above dataset construction with high probability. In particular, the upper bound from Theorem 3 is tight for k-medians up to the leading constant for this dataset.

The analysis will follow the standard bound on the maximum clique size in a random graph. Consider fixing any $\ell = 3\log_2 k$ coordinates to ± 1. When the set of k centers C is chosen uniformly at random from $\{\pm 1\}^d$ and $d = k^3$, we show that with high probability there are at most ℓ centers consistent with these values. When IMM builds the tree, it always chooses a threshold that reduces the number of centers in the children of the current node, and hence, it never splits on the same feature twice. Moreover, it stops the recursion when there is a single center in a leaf. Therefore, after $3\log_2 k$ thresholds, the remaining depth of the tree is at most $3\log_2 k$, and hence, the total depth of the tree is at most $6\log_2 k$ as well.

More formally, let $\sigma \in \{\pm 1\}^\ell$ be any sign pattern, and let $C_{I,\sigma}$ be set of centers having pattern σ when projected onto coordinates $I \subseteq [d]$ with $|I| = \ell$. Then, using the standard upper bound on the binomial

coefficient, we have

$$\Pr\Big[|C_{I,\sigma}| \geq \ell\Big] \leq \Pr\Big[|C_{I,\sigma}| = \ell\Big] \leq \mathbb{E}\Big[|\{I : |C_{I,\sigma}| = \ell\}|\Big] = \binom{d}{\ell} 2^{-\ell^2} \leq \left(\frac{de}{\ell}\right)^{\ell} 2^{-\ell^2} = \left(\frac{k^3 e}{2^\ell \ell}\right)^{\ell}.$$

Therefore, plugging in $2^\ell = k^3$, we see that this probability is at most $(e/\ell)^\ell$. Taking a union bound over the 2^ℓ possible settings of $\sigma \in \{\pm 1\}^\ell$ shows that the probability that there are ℓ centers consistent with any σ tends to zero as k increases.

C Upper Bound Proofs for Two Clusters

Theorem 5. *For any dataset $\mathcal{X} \subseteq \mathbb{R}^d$, there is a threshold cut $\widehat{C}$ such that the 2-medians cost satisfies*

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

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

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

Notation. We denote the optimal clusters as C^1 and C^2 with centers $\boldsymbol{\mu}^1$ and $\boldsymbol{\mu}^2$. Notice that we can assume $\boldsymbol{\mu}_i^1 \leq \boldsymbol{\mu}_i^2$ for each coordinate i because negating the ith coordinate for all points in the dataset does not change the 2-medians/means cost. We define t to be the minimum number of necessary changes between C^1, C^2 and any clustering $\widehat{C}^1, \widehat{C}^2$ based on a single threshold cut. In particular, assume that a single threshold partitions $\mathcal{X}$ into $\widehat{C}^1, \widehat{C}^2$ such that $t = \min(|C^1 \Delta \widehat{C}^1|, |C^1 \Delta \widehat{C}^2|)$.

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. In both cases, we simply need that the threshold cut $\widehat{C} = (\widehat{C}^1, \widehat{C}^2)$ minimizes the number of mistakes t over all threshold cuts compared to the optimal clusters.

We begin with a structural claim regarding the mistakes in 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 mistakes. We utilize Hall's theorem on perfect matchings in bipartite graphs.

Proposition 2 (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 6. *Let $\mathcal{X} \subseteq \mathbb{R}^d$ be partitioned into clusters C^1 and C^2. Assume that any threshold cut makes t mistakes compared to C^1 and C^2. For each coordinate $i \in [d]$, 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 if $\mu_i^1 \leq \mu_i^2$ then $q_i^j \leq p_i^j$ for every $j \in [t]$ (and if $\mu_i^1 \geq \mu_i^2$ then $q_i^j \geq p_i^j$ 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 will correspond to a matching the following bipartite graph (P, Q). Let $Q = C^2$ and define $P \subseteq C^1$ to be 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 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 decreasing value of ith coordinate, $p_i^1 \geq \cdots \geq p_i^t$. Now, notice that vertices in A 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 $t - j + 1$ neighbors, because this implies that any subset $P' \subseteq P$ has at least $|P'|$ neighbors, guaranteeing a matching of size $|P| = t$.

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

C.1 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 mistakes, which is the minimum possible. Applying Lemma 3 in the special case of $k=2$ implies that

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

Therefore, we simply need to prove that

$$t \left\|\boldsymbol{\mu}^1 - \boldsymbol{\mu}^2\right\|_1 \le \text{cost}(opt).$$

Applying Lemma 6 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 $\mu_i^1 \le \mu_i^2$ and $q_i^j \le p_i^j$, which implies that

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

C.2 Proof for 2-means

Suppose $\boldsymbol{\mu}^1, \boldsymbol{\mu}^2$ are optimal 2-means centers for clusters C^1 and C^2, and that the threshold cut $\widehat{C}$ makes t mistakes, which is the minimum possible. We can use the same proof idea as in the 2-medians case that first applies Lemma 3 and then use the matching lemma, Lemma 6. This will lead to an analysis of 6 approximation. The reason is that we apply twice Claim 1, which is not tight. A proof that improves that approximation to 4 require us to apply Claim 1 only once by combining its two applications.

We use the same notation as the proof of Lemma 3 and bound $\text{cost}(\widehat{C})$ in terms of the mistakes:

$$\begin{aligned}
\text{cost}(\widehat{C}) \le &\sum_{j=1}^{2} \sum_{\mathbf{x} \in \mathcal{X}^{\mathsf{cor}} \cap \widehat{C}_j} \|\mathbf{x} - \boldsymbol{\mu}^j\|_2^2 + \sum_{j=1}^{2} \sum_{\mathbf{x} \in \mathcal{X}^{\mathsf{mis}} \cap \widehat{C}_j} \|\mathbf{x} - \boldsymbol{\mu}^j\|_2^2 \\
= &\sum_{\mathbf{x} \in \mathcal{X}^{\mathsf{cor}}} \|\mathbf{x} - c(\mathbf{x})\|_2^2 + \sum_{j=1}^{2} \sum_{\mathbf{x} \in \mathcal{X}^{\mathsf{mis}} \cap \widehat{C}_j} \|\mathbf{x} - \boldsymbol{\mu}^j\|_2^2
\end{aligned} \tag{1}$$

The goal now is to bound the second term using $\text{cost}(opt)$. Denote the t points in $\mathcal{X}^{\mathsf{mis}}$ by $\mathcal{X}^{\mathsf{mis}} = \{\mathbf{r}^1, \ldots, \mathbf{r}^t\}$. Assume that the first ℓ points are in the first cluster, $\mathbf{r}^1, \ldots, \mathbf{r}^\ell \in C^1$, and the rest are in the second cluster, $\mathbf{r}^{\ell+1}, \ldots, \mathbf{r}^t \in C^2$.

Applying Lemma 6 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 $q_i^j \leq p_i^j$.

For each point r_j in the first ℓ points in $\mathcal{X}^{\mathsf{mis}}$, if $r_i^j \geq p_i^j$ then we can replace $\mathbf{p}^j$ with $\mathbf{r}^j$, thus we can assume without loss of generality that $p_i^j \geq r_i^j$.

We next want to show that cost(opt) is lower bounded by a function of the mistakes. There will be two cases depending on whether $p_i^j \leq \mu_i^2$ or not. The harder case is the first where the improvement of the approximation from 6 to 4 rises. Instead of first bounding the distance between $\mathbf{r}^j$ and its new center using the distance to its original center and then accounting for $\left\|\boldsymbol{\mu}^1 - \boldsymbol{\mu}^2\right\|_2^2$, we directly account for the distance between $\mathbf{r}^j$ and its new center.

- if $p_i^j \leq \mu_i^2$, then Claim 1 implies that

$$\begin{aligned}(\mu_i^2 - q_i^j)^2 + (p_i^j - \mu_i^1)^2 + (\mu_i^1 - r_i^j)^2 &\geq \frac{1}{3}(\mu_i^2 - q_i^j + p_i^j - \mu_i^1 + \mu_i^1 - r_i^j)^2 \\ &= \frac{1}{3}((\mu_i^2 - q_i^j) + (p_i^j - r_i^j))^2 \geq \frac{1}{3}(\mu_i^2 - r_i^j)^2,\end{aligned}$$

  where the last inequality follows from the fact that $q_i^j \leq p_i^j$ and $r_i^j \leq p_i^j$.

- if $\mu_i^2 \leq p_i^j$, then again Claim 1 implies that

$$\begin{aligned}(p_i^j - \mu_i^1)^2 + (\mu_i^1 - r_i^j)^2 &\geq (\mu_i^2 - \mu_i^1)^2 + (\mu_i^1 - r_i^j)^2 \\ &\geq \frac{1}{2}(\mu_i^2 - \mu_i^1 + \mu_i^1 - r_i^j)^2 = \frac{1}{2}(\mu_i^2 - r_i^j)^2.\end{aligned}$$

The two cases imply that

$$(\mu_i^2 - q_i^j)^2 + (p_i^j - \mu_i^1)^2 + (\mu_i^1 - r_i^j)^2 \geq \frac{1}{3}(\mu_i^2 - r_i^j)^2.$$

Similarly for each point r_j in the last $t - \ell$ points in $\mathcal{X}^{\mathsf{mis}}$, we have

$$(\mu_i^2 - q_i^j)^2 + (p_i^j - \mu_i^1)^2 + (\mu_i^2 - r_i^j)^2 \geq \frac{1}{3}(\mu_i^1 - r_i^j)^2.$$

Therefore,

$$\begin{aligned}\text{cost}(opt) &\geq \sum_{i=1}^{d}\sum_{j=1}^{\ell}(\mu_i^2 - q_i^j)^2 + (p_i^j - \mu_i^1)^2 + (\mu_i^1 - r_i^j)^2 \\ &+ \sum_{i=1}^{d}\sum_{j=\ell+1}^{t}(\mu_i^2 - q_i^j)^2 + (p_i^j - \mu_i^1)^2 + (\mu_i^2 - r_i^j)^2 \\ &\geq \frac{1}{3}\sum_{j=1}^{\ell}\sum_{i=1}^{d}(\mu_i^2 - r_i^j)^2 + \frac{1}{3}\sum_{j=\ell+1}^{t}\sum_{i=1}^{d}(\mu_i^1 - r_i^j)^2 \\ &= \frac{1}{3}\sum_{j=1}^{\ell}\left\|\mathbf{r}^j - \boldsymbol{\mu}^2\right\|_2^2 + \frac{1}{3}\sum_{j=\ell+1}^{t}\left\|\mathbf{r}^j - \boldsymbol{\mu}^1\right\|_2^2 \\ &= \frac{1}{3}\sum_{\mathbf{x}\in\mathcal{X}^{\mathsf{mis}}\cap\widehat{C}_1}\|\mathbf{x} - \boldsymbol{\mu}^1\|_2^2 + \frac{1}{3}\sum_{\mathbf{x}\in\mathcal{X}^{\mathsf{mis}}\cap\widehat{C}_2}\|\mathbf{x} - \boldsymbol{\mu}^2\|_2^2\end{aligned}$$

Together with Inequality (1) we have

$$\text{cost}(\widehat{C}) \leq \text{cost}(opt) + 3 \cdot \text{cost}(opt) = 4 \cdot \text{cost}(opt),$$

and this completes the proof.

D Lower bounds for two clusters

Without loss of generality we can assume that $d \geq 2$. We use the following dataset for both 2-medians and 2-means. It consists of $2d$ points, partitioned into two clusters of size d, which are the points with Hamming distance exactly one from the vector with all 1 entries and the vector with all -1 entries:

Optimal Cluster 1	**Optimal Cluster 2**
$(0, -1, -1, -1 \ldots, -1)$	$(0, 1, 1, 1 \ldots, 1)$
$(-1, 0, -1, -1 \ldots, -1)$	$(1, 0, 1, 1 \ldots, 1)$
$(-1, -1, 0, -1 \ldots, -1)$	$(1, 1, 0, 1 \ldots, 1)$
$\vdots$	$\vdots$
$(-1, -1, -1, -1 \ldots, 0)$	$(1, 1, 1, 1 \ldots, 0)$

Let $\widehat{C} = (\widehat{C}^1, \widehat{C}^2)$ be the best threshold cut.

2-medians lower bound. The cost of the cluster with centers $(1, \ldots, 1)$ and $(-1, \ldots, -1)$ is $2d$, as each point is responsible for a cost of 1. Thus, $\text{cost}(opt) \leq 2d$.

There is a coordinate i and a threshold θ that defines the cut $\widehat{C}$. For any coordinate i, there are only three possible values: $-1, 0, 1$. Thus θ is either in $(-1, 0)$ or in $(0, 1)$. Without loss of generality, assume that $\theta \in (-1, 0)$ and $i = 1$. Thus, the cut is composed of two clusters: one of size $d - 1$ and the other of size $d + 1$, in the following way:

Cluster $\widehat{C}^1$	**Cluster $\widehat{C}^2$**
$(-1, 0, -1, -1 \ldots, -1)$	$(1, 0, 1, 1 \ldots, 1)$
$(-1, -1, 0, -1 \ldots, -1)$	$(1, 1, 0, 1 \ldots, 1)$
$\vdots$	$\vdots$
$(-1, -1, -1, -1 \ldots, 0)$	$(1, 1, 1, 1 \ldots, 0)$
	$(0, 1, 1, 1 \ldots, 1)$
	$(0, -1, -1, -1 \ldots, -1)$

Using Fact 2, an optimal center of the first cluster is all -1, and the optimal center for the second cluster is all 1. The cost of the first cluster is $d - 1$, as each point costs 1. The cost of the second cluster is composed of two terms d for all points that include 1 in at least one coordinate and the cost of point $(0, -1, \ldots, -1)$ is $2(d - 1) + 1$. So the total cost is $4d - 2$. Thus $\text{cost}(\widehat{C}) \geq (2 - 1/d)\text{cost}(opt)$.

2-means lower bound. Focus on the clustering with centers $(^{(d-1)}/_{d}, \ldots, ^{(d-1)}/_{d})$ and $(-^{(d-1)}/_{d}, \ldots, -^{(d-1)}/_{d})$. The cost of each point in the data is composed of (1) one coordinate with value zero, and the cost of this coordinate is $(^{(d-1)}/_{d})^2$ (2) $d - 1$ coordinates each with cost $^{1}/_{d^2}$. Thus, each point has a cost of $^{(d-1)^2}/_{d^2} + ^{d-1}/_{d^2}$. Thus, the total cost is $\frac{2(d-1)^2+2(d-1)}{d} = 2(d-1)$. This implies that $\text{cost}(opt) \leq 2(d-1)$.

Assume without loss of generality that $\widehat{C}$ is defined using coordinate $i = 1$ and threshold -0.5. The resulting clusters $\widehat{C}^1$ and $\widehat{C}^2$ are as in the case of 2-medians. The optimal centers are (see Fact 1):

$\left(-1, -\frac{d-2}{d-1}, \ldots, -\frac{d-2}{d-1}\right)$ and $\left(\frac{d-1}{d+1}, \frac{d-2}{d+1}, \ldots, \frac{d-2}{d+1}\right)$. We want to lower bound cost($\widehat{C}$). We start with the cost of the first cluster, i.e. $\widehat{C}^1$. To do so for each point in $\widehat{C}^1$, we will evaluate the contribution of each coordinate to the cost (1) the first coordinate adds 0 to the cost (2) the coordinate with value 0, adds $\left(\frac{d-2}{d-1}\right)^2$ to the cost (3) the rest of the $d-2$ coordinates adds $1/(d-1)^2$. Thus, each point in $\widehat{C}^1$ adds to the cost $\left(\frac{d-2}{d-1}\right)^2 + \frac{d-2}{(d-1)^2} = \frac{d-2}{d-1}$. Since $\widehat{C}^1$ contains $d-1$ points, its total cost is $d-2$.

Moving on to evaluating the cost of $\widehat{C}^2$, the cost of the point $(0, -1, \ldots, -1)$ is composed of two terms (1) the first coordinate adds $\left(\frac{d-1}{d+1}\right)^2$ to the cost (2) each of the other $d-1$ coordinates adds $\left(1 + \frac{d-2}{d+1}\right)^2$ to the cost. Thus, this point adds

$$\left(\frac{d-1}{d+1}\right)^2 + (d-1)\left(1 + \frac{d-2}{d+1}\right)^2 = \frac{(d-1)d(4d-3)}{(d+1)^2}.$$

Similarly, the point $(0, 1, \ldots, 1)$ adds to the cost

$$\left(\frac{d-1}{d+1}\right)^2 + (d-1)\left(1 - \frac{d-2}{d+1}\right)^2 = \frac{(d-1)(d+8)}{(d+1)^2}.$$

Finally, each of the $d-1$ remaining points in $\widehat{C}^2$ adds to the cost

$$\left(1 - \frac{d-1}{d+1}\right)^2 + \left(\frac{d-2}{d+1}\right)^2 + (d-1)\left(1 - \frac{d-2}{d-1}\right)^2 = \frac{d^2+5d-1}{(d+1)^2}$$

Thus, the cost of $\widehat{C}^2$ is

$$\frac{(d-1)(5d^2+3d+7)}{(d+1)^2}$$

Summing up the costs of $\widehat{C}^1$ and $\widehat{C}^2$, for $d \geq 2$

$$\text{cost}(\widehat{C}) \geq (d-2) + \frac{(d-1)(5d^2+3d+7)}{(d+1)^2} \geq 6(d-1)\left(1 - \frac{1}{d}\right)^2 \geq 3\left(1 - \frac{1}{d}\right)^2 \cdot \text{cost}(opt)$$

E Efficient Implementation using Dynamic Programming for $k = 2$

E.1 The 2-means case

The psudo-code for finding the best threshold for $k = 2$ depicted in Algorithm 2.

In time $O(d)$ we can calculate cost$(p+1)$ and the new centers by using the value cost(p) and the previous centers. Throughout the computation we save in memory

1. Two vectors $\mathbf{s}^p = \sum_{j=1}^{p} \mathbf{x}^j$ and $\mathbf{r}^p = \sum_{j=p+1}^{n} \mathbf{x}^j$.
2. Scalar $u = \sum_{j=1}^{n} \left\|\mathbf{x}^j\right\|_2^2$

We also make use of the identity:

$$\text{cost}(p) \quad = \quad u - \frac{1}{p}\left\|\mathbf{s}^p\right\|_2^2 - \frac{1}{n-p}\left\|\mathbf{r}^p\right\|_2^2.$$

ALGORITHM 2: Optimal Threshold for 2-means

Input : $\mathbf{x}^1, \ldots, \mathbf{x}^n$ – vectors in $\mathbb{R}^d$
Output : i – Coordinate
θ – Threshold

```
best_cost ← ∞
best_coordinate ← NULL
best_threshold ← NULL
u ← Σ_{j=1}^{n} ||x^j||_2^2
foreach i ∈ [1, ..., d] do
    s ← zeros(d)
    r ← Σ_{j=1}^{n} x^j
    X ← sorted(x^1, ..., x^n by coordinate i)
    foreach x^j ∈ X do
        s ← s + x^j
        r ← r − x^j
        cost ← u − (1/j) ||s||_2^2 − (1/(n−j)) ||r||_2^2
        if cost < best_cost and x_i^j ≠ x_i^{j+1} then
            best_cost ← cost
            best_coordinate ← i
            best_threshold ← x_i^j
        end
    end
end
return best_coordinate, best_threshold
```

This identity is correct because

$$
\begin{aligned}
\mathrm{cost}(p) &= \sum_{j=1}^{p} \left\|\mathbf{x}^j - \boldsymbol{\mu}^1(p)\right\|_2^2 + \sum_{j=p+1}^{n} \left\|\mathbf{x}^j - \boldsymbol{\mu}^2(p)\right\|_2^2 \\
&= \sum_{j=1}^{p} \left\|\mathbf{x}^j\right\|_2^2 - 2\sum_{j=1}^{p}\langle \mathbf{x}^j, \boldsymbol{\mu}^1(p)\rangle + \sum_{j=1}^{p} \left\|\boldsymbol{\mu}^1(p)\right\|_2^2 + \\
&\quad \sum_{j=p+1}^{n} \left\|\mathbf{x}^j\right\|_2^2 - 2\sum_{j=p+1}^{n}\langle \mathbf{x}^j, \boldsymbol{\mu}^2(p)\rangle + \sum_{j=p+1}^{n} \left\|\boldsymbol{\mu}^2(p)\right\|_2^2 \\
&= \sum_{j=1}^{n} \left\|\mathbf{x}^j\right\|_2^2 - 2\langle \sum_{j=1}^{p}\mathbf{x}^j, \boldsymbol{\mu}^1(p)\rangle + \frac{1}{p}\left\|\sum_{j=1}^{p}\mathbf{x}^j\right\|_2^2 - \\
&\quad 2\langle \sum_{j=p+1}^{n}\mathbf{x}^j, \boldsymbol{\mu}^2(p)\rangle + \frac{1}{n-p}\left\|\sum_{j=p+1}^{n}\mathbf{x}^j\right\|_2^2 \\
&= \sum_{j=1}^{n} \left\|\mathbf{x}^j\right\|_2^2 - \frac{2}{p}\langle \mathbf{s}^p, \mathbf{s}^p\rangle + \frac{1}{p}\left\|\mathbf{s}^p\right\|_2^2 - \frac{2}{n-p}\langle \mathbf{r}^p, \mathbf{r}^p\rangle + \frac{1}{n-p}\left\|\mathbf{r}^p\right\|_2^2 \\
&= u - \frac{1}{p}\left\|\mathbf{s}^p\right\|_2^2 - \frac{1}{n-p}\left\|\mathbf{r}^p\right\|_2^2
\end{aligned}
$$

By invoking this identity, we can quickly compute the cost of placing the first p points in cluster one and the last $n-p$ points in cluster two. Each such partition can be achieved by using a threshold θ between x_i^p and x_i^{p+1}. Our algorithm computes these costs for each feature $i \in [d]$. Then, we output the feature i and

threshold θ that minimizes the cost. This guarantees that we find the best possible threshold cut.

Overall, Algorithm 2 iterates over the d features, and for each feature it sorts the n vectors according to their values in the current feature. Next, the algorithm iterates over the n vectors and for each potential threshold, it calculates the cost by evaluating the inner product of two d-dimensional vectors. Overall its runtime complexity is $O\left(nd^2 + nd\log n\right)$.

E.2 The 2-medians case

The high level idea of a finding an optimal 2-medians cut is similar to the 2-means algorithm. The algorithm goes over all possible thresholds. For each threshold, it finds the optimal centers and calculates the cost accordingly. Then, it outputs the threshold cut that minimizes the 2-medians cost.

ALGORITHM 3: Optimal Threshold for 2-medians

Input : $\mathbf{x}^1, \ldots, \mathbf{x}^n$ – vectors in $\mathbb{R}^d$
Output: i – Coordinate
θ – Threshold

best_cost $\leftarrow \infty$
best_coordinate $\leftarrow$ NULL
best_threshold $\leftarrow$ NULL
foreach $i \in [1, \ldots, d]$ **do**
  $\boldsymbol{\mu}^2(0) \leftarrow \texttt{median}(\mathbf{x}^1, \ldots \mathbf{x}^n)$
  cost $\leftarrow \sum_{j=1}^{n} \left\|\mathbf{x}^j - \boldsymbol{\mu}^2(0)\right\|_1$
  $\mathcal{X} \leftarrow \texttt{sorted}(\mathbf{x}^1, \ldots, \mathbf{x}^n$ by coordinate $i)$
  foreach $j \in [1, \ldots, n-1]$ **do**
    $\boldsymbol{\mu}^1(j) \leftarrow \texttt{median}(\mathbf{x}^1, \ldots \mathbf{x}^j)$
    $\boldsymbol{\mu}^2(j) \leftarrow \texttt{median}(\mathbf{x}^{j+1}, \ldots \mathbf{x}^n)$
    cost $\leftarrow$ cost $+ \left\|\mathbf{x}^j - \boldsymbol{\mu}^1(j)\right\|_1 - \left\|\mathbf{x}^j - \boldsymbol{\mu}^2(j-1)\right\|_1$
    if *cost* $<$ *best_cost and* $x_i^j \neq x_i^{j+1}$ **then**
      best_cost $\leftarrow$ cost
      best_coordinate $\leftarrow i$
      best_threshold $\leftarrow x_i^j$
    end
  end
end
return best_coordinate, best_threshold

Updating cost. To update the cost we need to show how to express cost$(p+1)$ in terms of cost(p). We know that cost$(p+1)$ is equal to

$$\text{cost}(p+1) = \sum_{\mathbf{x} \in C_1} \left\|x - \boldsymbol{\mu}^1(p+1)\right\|_1 + \sum_{\mathbf{x} \in C_2} \left\|x - \boldsymbol{\mu}^2(p+1)\right\|_1 .$$

For every feature $i \in [d]$, there are $n-1$ thresholds to consider. After sorting by this feature, we can consider all splits into C_1 and C_2, where C_1 contains the p smallest points, and C_2 contains the $n-p$ largest points. We increase p from $p=1$ to $p=n-1$, computing the clusters and cost at each step. If p is odd then the median of C_1 (i.e., the optimal center of C_1) does not change compared to $p-1$. The only contribution to the cost is the point $\mathbf{x}$ that moved from C_2 to C_1. If p is even, then at each coordinate there are two cases, depending on whether the median changes or not. If it changes, then let Δ denote the change in cost of the points in C_1 that are smaller than the median. By symmetry, the change in the cost of the points that are

larger is $-\Delta$. Thus, the change of the cost is balanced by the points that are larger and smaller than the median. Similar reasoning holds for the other cluster C_2. Therefore, we conclude that moving $\mathbf{x}$ from C_2 to C_1 changes the cost by exactly $\left\|\mathbf{x} - \boldsymbol{\mu}^1(p+1)\right\|_1 - \left\|\mathbf{x} - \boldsymbol{\mu}^2(p)\right\|_1$. Thus, we have the following connection between $\text{cost}(p+1)$ and $\text{cost}(p)$:

$$\text{cost}(p+1) = \text{cost}(p) + \left\|\mathbf{x} - \boldsymbol{\mu}^1(p+1)\right\|_1 - \left\|\mathbf{x} - \boldsymbol{\mu}^2(p)\right\|_1 .$$

Updating centers. For each p, the cost update relies on efficient calculations of the centers $\boldsymbol{\mu}^1(p)$ and $\boldsymbol{\mu}^2(p+1)$. The centers $\boldsymbol{\mu}^1(p), \boldsymbol{\mu}^2(p)$ are the medians of the clusters at the pth threshold. Note that moving from the pth thresold to the $(p+1)$th will only change the clusters by moving one vector from one cluster to the other. We can determine the changes efficiently by using d arrays, one for each coordinate. Each array will contain (pointers to) the input vectors $\mathcal{X}$ sorted by their ith feature value. As we move the threshold along a single coordinate, we can read off the partition into two clusters, and we can compute the median of each cluster by considering the midpoint in the sorted list.

Overall, this procedure computes the cost of each threshold, while also determining the partition into two clusters and their centers (medians). The time is $O(nd \log n)$ to sort by each feature, and $O(nd^2)$ to compute $\text{cost}(p)$ for each $p \in [n]$ and each feature. Therefore, the total time for the 2-medians algorithm is $O(nd^2 + nd \log n)$.