Near-optimal Algorithms for Explainable \( k \)-Medians and \( k \)-Means

Konstantin Makarychev \(^*\) \(^1\) Liren Shan \(^*\) \(^1\)


deduction is not necessarily easy to interpret for a human. Every \( k \)-medians or \( k \)-means clustering is defined by a set of \( k \) centers \( c_1, c_2, \ldots, c_k \), where each cluster is the set of points located closer to a fixed center \( c_i \) than to any other center \( c_j \). That is, for points in cluster \( i \), we must have \( \arg \min_j ||x - c_j|| = i \). Thus, in order to determine to which cluster a particular point belongs, we need to compute distances from point \( x \) to all centers \( c_j \). Each distance depends on all coordinates of the points. Hence, for a human, it is not even easy to figure out to which cluster in \( k \)-means or \( k \)-medians clustering a particular point belongs; let alone interpret the entire clustering.

In every day life, we are surrounded by different types of classifications. Consider the following examples from Wikipedia: (1) *Performance cars are capable of going from 0 to 60 mph in under 5 seconds*; (2) *Modern sources currently define skyscrapers as being at least 100 metres or 150 metres in height*; (3) *Very-low-calorie diets are diets of 800 kcal or less energy intake per day, whereas low-calorie diets are between 1000-1200 kcal per day*. Note that all these definitions depend on a single feature which makes them easy to understand.

The above discussion leads us to the idea of Dasgupta et al. (2020), who proposed to use threshold (decision) trees to describe clusters (see also Liu, Xia, and Yu (2005), Fraiman, Ghattas, and Svarc (2013), Bertsimas, Orfanoudaki, and Wiberg (2018), and Saisubramanian, Galhotra, and Zilberstein (2020)).

A threshold tree is a binary classification tree with \( k \) leaves. Every internal node \( u \) of the tree splits the data into two sets by comparing a single feature \( i_u \) of each data point with a threshold \( \theta_u \). The first set is the set of points with \( x_{i_u} \leq \theta_u \); the second set is the set of points with \( x_{i_u} > \theta_u \). These two sets are then recursively partitioned by the left and right children of \( u \). Thus, each point \( x \) in the data set is eventually assigned to one of \( k \) leaves of the threshold tree \( T \). This gives us a partitioning of the data set \( X \) into clusters \( P = (P_1, \ldots, P_k) \). We note that threshold decision trees are special cases of binary space partitioning (BSP) trees and similar to \( k \)-d trees (Bentley, 1975).

Dasgupta et al. (2020) suggested that we measure the quality of a threshold tree using the standard \( k \)-means and \( k \)-medians objectives. Specifically, the \( k \)-medians in \( \ell_1 \) cost

---

\(^*\)Equal contribution \(^1\)Northwestern University, Evanston, IL, USA. Correspondence to: Liren Shan <lirenshan2023@u.northwestern.edu>.

Abstract

We consider the problem of explainable \( k \)-medians and \( k \)-means introduced by Dasgupta, Frost, Moshkovitz, and Rashtchian (ICML 2020). In this problem, our goal is to find a threshold decision tree that partitions data into \( k \) clusters and minimizes the \( k \)-medians or \( k \)-means objective. The obtained clustering is easy to interpret because every decision node of a threshold tree splits data based on a single feature into two groups. We propose a new algorithm for this problem which is \( \tilde{O}(\log k) \) competitive with \( k \)-medians with \( \ell_1 \) norm and \( \tilde{O}(k) \) competitive with \( k \)-means. This is an improvement over the previous guarantees of \( O(k) \) and \( O(k^2) \) by Dasgupta et al (2020). We also provide a new algorithm which is \( \tilde{O}(\log^{3/2} k) \) competitive for \( k \)-medians with \( \ell_2 \) norm. Our first algorithm is near-optimal: Dasgupta et al (2020) showed a lower bound of \( \Omega(\log k) \) for \( k \)-medians; in this work, we prove a lower bound of \( \tilde{\Omega}(k) \) for \( k \)-means. We also provide a lower bound of \( \Omega(\log k) \) for \( k \)-medians with \( \ell_2 \) norm.

1. Introduction

In this paper, we investigate the problem of explainable \( k \)-means and \( k \)-medians clustering which was recently introduced by Dasgupta, Frost, Moshkovitz, and Rashtchian (2020). Suppose, we have a data set which we need to partition into \( k \) clusters. How can we do it? Of course, we could use one of many standard algorithms for \( k \)-means or \( k \)-medians clustering. However, we want to find an explainable clustering – clustering which can be easily understood by a human being. Then, \( k \)-means or \( k \)-medians clustering may not be the best options for us.

Note that though every cluster in a \( k \)-means and \( k \)-medians clustering has a simple mathematical description, this description is not necessarily easy to interpret for a human.
Near-optimal Algorithms for Explainable \( k \)-Medians and \( k \)-Means

<table>
<thead>
<tr>
<th></th>
<th>( k )-medians in ( \ell_1 )</th>
<th>( k )-medians in ( \ell_2 )</th>
<th>( k )-means</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Lower</td>
<td>Upper</td>
<td>Lower</td>
</tr>
<tr>
<td>Our results</td>
<td>( O(\log k \log k \log k) )</td>
<td>( \Omega(k) )</td>
<td>( \Omega(\log^{3/2} k) )</td>
</tr>
<tr>
<td>Dasgupta et al. (2020)</td>
<td>( \Omega(\log k) )</td>
<td>( O(k) )</td>
<td>( \Omega(\log k) )</td>
</tr>
</tbody>
</table>

Figure 1. Summary of our results. The table shows known upper and lower bounds on the price of explainability for \( k \)-medians in \( \ell_1 \) and \( \ell_2 \), and for \( k \)-means.

of the threshold tree \( T \) equals (1), the \( k \)-medians in \( \ell_2 \) cost equals (2) and \( k \)-means cost equals (3):

\[
\text{cost}_{\ell_1}(X, T) = \sum_{i=1}^{k} \sum_{x \in P_i} \|x - c^i\|_1, \tag{1}
\]

\[
\text{cost}_{\ell_2}(X, T) = \sum_{i=1}^{k} \sum_{x \in P_i} \|x - c^i\|_2, \tag{2}
\]

\[
\text{cost}_{\ell_2^2}(X, T) = \sum_{i=1}^{k} \sum_{x \in P_i} \|x - c^i\|^2_2, \tag{3}
\]

where \( c^i \) is the \( \ell_1 \)-median of cluster \( P_i \) in (1), the \( \ell_2 \)-median of cluster \( P_i \) in (2), and the mean of cluster \( P_i \) in (3).

This definition raises obvious questions: Can we actually find a good explainable clustering? Moreover, how good can it be comparing to a regular \( k \)-medians and \( k \)-means clustering? Let \( \text{OPT}_{\ell_1}(X) \), \( \text{OPT}_{\ell_2}(X) \), and \( \text{OPT}_{\ell_2^2}(X) \) be the optimal solutions to (regular) \( k \)-medians with \( \ell_1 \) norm, \( k \)-medians with \( \ell_2 \) norm, and \( k \)-means, respectively. Dasgupta et al. (2020) defined the price of explainability as the ratio \( \text{cost}_{\ell_1}(X, T) / \text{OPT}_{\ell_1}(X) \) for \( k \)-medians in \( \ell_1 \) and \( \text{cost}_{\ell_2}(X, T) / \text{OPT}_{\ell_2}(X) \) for \( k \)-means. The price of explainability shows by how much the optimal unconstrained solution is better than the best explainable solution for the same data set.

In their paper, Dasgupta et al. (2020) gave upper and lower bounds on the price of explainability. They proved that the price of explainability is upper bounded by \( O(k) \) and \( O(k^2) \) for \( k \)-medians in \( \ell_1 \) and \( k \)-means, respectively. Furthermore, they designed two algorithms that given a \( k \)-medians in \( \ell_1 \) or \( k \)-means clustering, produce an explainable clustering with cost at most \( O(k) \) and \( O(k^2) \) times the cost of original clustering (respectively). They also provided examples for which the price of explainability of \( k \)-medians in \( \ell_1 \) and \( k \)-means is at least \( \Theta(\log k) \).

1.1. Our results

In this work, we give almost tight bounds on the price of explainability for both \( k \)-medians in \( \ell_1 \) and \( k \)-means. Specifically, we show how to transform any clustering to an explainable clustering with cost at most \( O(\log k \log k \log k) \) times the original cost for the \( k \)-medians \( \ell_1 \) objective and \( O(k \log k \log k \log k) \) for the \( k \)-means objective. Note that we get an exponential improvement over previous results for the \( k \)-medians \( \ell_1 \) objective. Furthermore, we present an algorithm for \( k \)-medians in \( \ell_2 \) with the price of explainability bounded by \( O(\log^{3/2} k) \). We complement these results with an almost tight lower bound of \( \Omega(k / \log k) \) for the \( k \)-means objective and an \( \Omega(\log k) \) lower bound for \( k \)-medians in \( \ell_2 \) objective. We summarise our results in Table 1.

Below, we formally state our main results. The costs of threshold trees and clusterings are defined by formulas (1), (2), (3), (4), (5), and (6).

**Theorem 1.1.** There exists a polynomial-time randomized algorithm that given a data set \( X \) and a set of centers \( C = \{c^1, \ldots, c^k\} \), finds a threshold tree \( T \) with expected \( k \)-medians in \( \ell_1 \) cost at most

\[
\mathbb{E}[\text{cost}_{\ell_1}(X, T)] \leq O(\log k \log k \log k) \cdot \text{cost}_{\ell_1}(X, C).
\]

**Theorem 1.2.** There exists a polynomial-time randomized algorithm that given a data set \( X \) and a set of centers \( C = \{c^1, \ldots, c^k\} \), finds a threshold tree \( T \) with expected \( k \)-means cost at most

\[
\mathbb{E}[\text{cost}_{\ell_2^2}(X, T)] \leq O(k \log k \log k) \cdot \text{cost}_{\ell_2^2}(X, C).
\]

We note that the algorithms by Dasgupta et al. (2020) also produce trees based on the given set of “reference” centers \( c^1, \ldots, c^k \). However, the approximation guarantees of those algorithms are \( O(k) \) and \( O(k^2) \), respectively. Our upper bound of \( O(k \log k \log k \log k) \) almost matches the lower bound of \( \Omega(\log k) \) given by Dasgupta et al. (2020). The upper bound of \( O(k \log k \log k \log k) \) almost matches the lower bound of \( \Omega(k / \log k) \) we show in Appendix D.

**Theorem 1.3.** There exists a polynomial-time randomized algorithm that given a data set \( X \) and a set of centers \( C = \{c^1, \ldots, c^k\} \), finds a threshold tree \( T \) with expected \( k \)-medians in \( \ell_2 \) cost at most

\[
\mathbb{E}[\text{cost}_{\ell_2}(X, T)] \leq O(\log^{3/2} k) \cdot \text{cost}_{\ell_2}(X, C).
\]

1.2. Related work

Dasgupta et al. (2020) introduced the explainable \( k \)-medians and \( k \)-means clustering problems and developed Iterative...
Mistake Minimization (IMM) algorithms for these problems. Later, Frost, Moshkovitz, and Rashtchian (2020) proposed algorithms that construct threshold trees with more than \( k \) leaves. Decision trees have been used for interpretable classification and clustering since 1980s. Breiman, Friedman, Olshen, and Stone (1984) proposed a popular decision tree algorithm called CART for supervised classification. For unsupervised clustering, threshold decision trees are used in many empirical methods based on different criteria such as information gain (Liu et al., 2005), local 1-means cost (Fraiman et al., 2013), Silhouette Metric (Bertsimas et al., 2018), and interpretability score (Saisubramanian et al., 2020).

The \( k \)-means and \( k \)-medians clustering problems have been extensively studied in the literature. The \( k \)-means++ algorithm proposed by Arthur and Vassilvitskii (2006) is the most widely used algorithm for \( k \)-means clustering. It provides an \( O(\log k) \) approximation. Li and Svensson (2016) provided a \( 1 + \sqrt{2} + \varepsilon \) approximation for \( k \)-medians in general metric spaces, which was improved to \( 2.611 + \varepsilon \) by Byrka, Pensyl, Rybicki, Srinivasan, and Trinh (2014). Ahmadian, Norouzi-Fard, Svensson, and Ward (2019) gave a 6.357 approximation algorithm for \( k \)-means. The \( k \)-medians and \( k \)-means problems are \( \gamma \)-hard (Megiddo & Supowit, 1984; Dasgupta, 2008; Aloise et al., 2009). Recently, Awasthi, Charikar, Krishnaswamy, and Sinop (2015) showed that it is also \( \gamma \)-hard to approximate the \( k \)-means objective within a factor of \( (1 + \varepsilon) \) for some positive constant \( \varepsilon \) (see also Lee et al. (2017)). Bhattacharya, Goyal, and Jaiswal (2020) showed that the Euclidean \( k \)-medians can not be approximated within a factor of \( (1 + \varepsilon) \) for some constant \( \varepsilon \) assuming the unique games conjecture.

Boutsidis et al. (2009), Boutsidis et al. (2014), Cohen et al. (2015), Makarychev et al. (2019) and Becchetti et al. (2019) showed how to reduce the dimensionality of a data set for \( k \)-means clustering. Particularly, Makarychev et al. (2019) proved that we can use the Johnson–Lindenstrauss transform to reduce the dimensionality of \( k \)-medians in \( \ell_2 \) and \( k \)-means to \( d' = O(\log k) \). Note, however, that the Johnson–Lindenstrauss transform cannot be used for the explainable \( k \)-medians and \( k \)-means problems, because this transform does not preserve the set of features. Instead, we can use a feature selection algorithm by Boutsidis et al. (2014) or Cohen et al. (2015) to reduce the dimensionality to \( d' = O(k) \).

Independently of our work, Laber and Murtinho (2021) proposed new algorithms for explainable \( k \)-medians with \( \ell_1 \) and \( k \)-means objectives. Their competitive ratios are \( O(d \log k) \) and \( O(d k \log k) \), respectively. Note that these competitive ratios depend on the dimension \( d \) of the space.

1. \( k \)-medians and \( k \)-means clustering problems have been extensively studied in the literature. The \( k \)-means++ algorithm proposed by Arthur and Vassilvitskii (2006) is the most widely used algorithm for \( k \)-means clustering. It provides an \( O(\log k) \) approximation. Li and Svensson (2016) provided a \( 1 + \sqrt{2} + \varepsilon \) approximation for \( k \)-medians in general metric spaces, which was improved to \( 2.611 + \varepsilon \) by Byrka, Pensyl, Rybicki, Srinivasan, and Trinh (2014). Ahmadian, Norouzi-Fard, Svensson, and Ward (2019) gave a 6.357 approximation algorithm for \( k \)-means. The \( k \)-medians and \( k \)-means problems are \( \gamma \)-hard (Megiddo & Supowit, 1984; Dasgupta, 2008; Aloise et al., 2009). Recently, Awasthi, Charikar, Krishnaswamy, and Sinop (2015) showed that it is also \( \gamma \)-hard to approximate the \( k \)-means objective within a factor of \( (1 + \varepsilon) \) for some positive constant \( \varepsilon \) (see also Lee et al. (2017)). Bhattacharya, Goyal, and Jaiswal (2020) showed that the Euclidean \( k \)-medians can not be approximated within a factor of \( (1 + \varepsilon) \) for some constant \( \varepsilon \) assuming the unique games conjecture.

Boutsidis et al. (2009), Boutsidis et al. (2014), Cohen et al. (2015), Makarychev et al. (2019) and Becchetti et al. (2019) showed how to reduce the dimensionality of a data set for \( k \)-means clustering. Particularly, Makarychev et al. (2019) proved that we can use the Johnson–Lindenstrauss transform to reduce the dimensionality of \( k \)-medians in \( \ell_2 \) and \( k \)-means to \( d' = O(\log k) \). Note, however, that the Johnson–Lindenstrauss transform cannot be used for the explainable \( k \)-medians and \( k \)-means problems, because this transform does not preserve the set of features. Instead, we can use a feature selection algorithm by Boutsidis et al. (2014) or Cohen et al. (2015) to reduce the dimensionality to \( d' = O(k) \).

Independently of our work, Laber and Murtinho (2021) proposed new algorithms for explainable \( k \)-medians with \( \ell_1 \) and \( k \)-means objectives. Their competitive ratios are \( O(d \log k) \) and \( O(d k \log k) \), respectively. Note that these competitive ratios depend on the dimension \( d \) of the space.
We now give an overview of our algorithms.

As the algorithm by Dasgupta et al. (2020), our algorithm builds a binary threshold tree (see Algorithm 1) to split a data set $X \subseteq \mathbb{R}^d$ into subsets that contain a unique optimal center for points contained in that subset. Note that the unique reference center in each leaf may not be the optimal center for points contained in that leaf. Thus, the total cost by assigning each point to the reference center in the same leaf of $T$ is an upper bound of the cost of threshold tree $T$.

The algorithm by Dasgupta et al. (2020) picks splitting cuts in a greedy way. Our algorithm chooses them at random. Specifically, to pick a cut $\omega_t \in \Omega$ at round $t$, our algorithm finds the maximum distance $D_t$ between two distinct centers $c^i, c^j$ that belong to the same set $X_u$ assigned to a leaf node $u$ i.e.,

$$D_t = \max_{u \text{ is a leaf}} \max_{c^i, c^j \in X_u} \|c^i - c^j\|.$$ 

Then, we let $A_t$ be the set of all $\omega \in \Omega$ that separate at least one pair of centers; and $B_t$ be the set of all $\omega \in \Omega$ that separate two centers at distance at most $D_t/k^3$. We pick $\omega_t$ uniformly at random (with respect to measure $\mu$) from the set $R_t = A_t \setminus B_t$.

Every $\omega \in R_t$ is contained in $A_t$, which means $\omega$ separates at least one pair of centers. Thus, our algorithm terminates in at most $k - 1$ iterations. It is easy to see that the running time of this algorithm is polynomial in the number of clusters $k$ and dimension of the space $d$. In Section E, we provide a

---

**Algorithm 1** Threshold tree construction for $k$-medians in $\ell_1$

**Input:** a data set $X \subseteq \mathbb{R}^d$ and set of centers $C = \{c^1, c^2, \ldots, c^k\} \subseteq \mathbb{R}^d$

**Output:** a threshold tree $T$

Set $S_{ij} = \{\omega \in \Omega : \delta_{ij}(\omega) \neq \delta_{ij}(\omega)\}$ for all $i, j \in \{1, \ldots, k\}$. Let $t = 0$.

Create a tree $T_0$ containing a root vertex $r$. Assign set $X_r = X \cup C$ to the root.

while $T_t$ contains a leaf with at least two distinct centers $c^i$ and $c^j$ do

Let $E_t = \bigcup_{\text{leaves } u} \{(i, j) : c^i, c^j \in X_u\}$ be the set of all not yet separated pairs of centers.

Let $D_t = \max_{(i, j) \in E_t} \|c^i - c^j\|_1$ be the maximum distance between two not separated centers.

Define two sets $A_t, B_t \subseteq \Omega$ as follows:

$$A_t = \bigcup_{(i, j) \in E_t} S_{ij} \quad \text{and} \quad B_t = \bigcup_{(i, j) \in E_t \atop s.t. \mu(S_{ij}) \leq D_t/k^3} S_{ij}.$$

Let $R_t = A_t \setminus B_t$. Pick a pair $\omega_t = (i, \theta)$ uniformly at random from $R_t$.

For every leaf node $u$ in $T_t$, split the set $X_u$ into two sets:

$$\text{Left} = \{x \in X_u : x_i \leq \theta\} \quad \text{and} \quad \text{Right} = \{x \in X_u : x_i > \theta\}.$$

If each of these sets contains at least one center from $C$, then create two children of $u$ in tree $T_t$ and assign sets Left and Right to the left and right child, respectively.

Denote the updated tree by $T_{t+1}$.

Update $t = t + 1$.

end while

---

3. Algorithms Overview

We now give an overview of our algorithms.

**k-medians in $\ell_1$.** We begin with the algorithm for $k$-medians in $\ell_1$. We show that its competitive ratio is $O(\log^2 k)$ in Section 4 and then show an improved bound of $O(\log k \log \log k)$ in Section 5.

As the algorithm by Dasgupta et al. (2020), our algorithm (see Algorithm 1) builds a binary threshold tree $T$ top-down. It starts with a tree containing only the root node $r$. This node is assigned the set of points $X_r$ that contains all points in the data set $X$ and all reference centers $c^i$. At every round, the algorithm picks some pair $\omega = (i, \theta) \in \Omega$ (as we discuss below) and then splits data points $x$ assigned to every leaf node $u$ into two groups $\{x \in X_u : x_i \leq \theta\}$ and $\{x \in X_u : x_i > \theta\}$. Here, $X_u$ denotes the set of points assigned to the node $u$. If this partition separates at least two centers $c^i$ and $c^j$, then the algorithm attaches two children to $u$ and assigns the first group to the left child and the second group to the right child. The algorithm terminates when all leaves contain exactly one reference center $c^i$. Then, we assign the points in each leaf of $T$ to its unique reference center. Note that the unique reference center in each leaf may not be the optimal center for points contained in that leaf. Thus, the total cost by assigning each point to the reference center in the same leaf of $T$ is an upper bound of the cost of threshold tree $T$.

The algorithm by Dasgupta et al. (2020) picks splitting cuts in a greedy way. Our algorithm chooses them at random. Specifically, to pick a cut $\omega_t \in \Omega$ at round $t$, our algorithm finds the maximum distance $D_t$ between two distinct centers $c^i, c^j$ that belong to the same set $X_u$ assigned to a leaf node $u$ i.e.,

$$D_t = \max_{u \text{ is a leaf}} \max_{c^i, c^j \in X_u} \|c^i - c^j\|_1.$$ 

Then, we let $A_t$ be the set of all $\omega \in \Omega$ that separate at least one pair of centers; and $B_t$ be the set of all $\omega \in \Omega$ that separate two centers at distance at most $D_t/k^3$. We pick $\omega_t$ uniformly at random (with respect to measure $\mu$) from the set $R_t = A_t \setminus B_t$.

Every $\omega \in R_t$ is contained in $A_t$, which means $\omega$ separates at least one pair of centers. Thus, our algorithm terminates in at most $k - 1$ iterations. It is easy to see that the running time of this algorithm is polynomial in the number of clusters $k$ and dimension of the space $d$. In Section E, we provide a

1As we discuss in Section E, we can also let $R_t = A_t$. However, this will make the analysis more involved.

---
variant of this algorithm with running time $\tilde{O}(kd)$.

**k-medians in $\ell_2$.** Our algorithm for $k$-medians with $\ell_2$ norm recursively partitions the data set $X$ using the following idea. It finds the median point $m$ of all centers in $X$. Then, it repeatedly makes cuts that separate centers from $m$. To make a cut, the algorithm chooses a random coordinate $i \in \{1, \ldots, d\}$, random number $\theta \in [0, R^3]$, and random sign $\sigma \in \{\pm 1\}$, where $R$ is the largest distance from a center in $X$ to the median point $m$. It then makes a threshold cut $(i, m + \sigma \sqrt{\theta})$. After separating more than half centers from $m$, the algorithm recursively calls itself for each of the obtained parts. In Appendix C, we show that the price of explainability for this algorithm is $O(\log^{3/2} k)$.

**k-means.** We now move to the algorithm for $k$-means. This algorithm embeds the space $\ell_2$ into $\ell_1$ using a specially crafted terminal embedding $\varphi$ (the notion of terminal embeddings was formally defined by Elkin et al. (2017)). The embedding satisfies the following property for every center $c$ (terminal) and every point $x \in \ell_2$, we have

$$\|\varphi(x) - \varphi(c)\|_1 \leq \|x - c\|_2^2 \leq 8k \cdot \|\varphi(x) - \varphi(c)\|_1.$$ 

Then, the algorithm partitions the data set $\varphi(X)$ with centers $\varphi(c^1), \ldots, \varphi(c^k)$ using Algorithm 1. The expected cost of partitioning is at most the distortion of the embedding (8k) times the competitive guarantee (O($\log k \log \log k$)) of Algorithm 1. In Section D, we show an almost matching bound for Algorithm 1 before iteration $t$. Consider a point $x$ in $X$. If $x$ is separated from its original center in $C$ by the cut generated at iteration $t$, then $x$ will be eventually assigned to some other center in the same leaf of $T_t$. By the triangle inequality, the new cost of $x$ at the end of the algorithm will be at most $\text{cost}_t(x, C) + D_t$, where $D_t$ is the maximum diameter of any leaf in $T_t$ (see Algorithm 1). Define a penalty function $\phi_t(x)$ as follows: $\phi_t(x) = D_t$ if $x$ is separated from its original center $c$ at time $t$; $\phi_t(x) = 0$, otherwise. Note that $\phi_t(x) \neq 0$ for at most one iteration $t$, and

$$\text{alg}_{\ell_1}(x) \leq \text{cost}_{\ell_1}(x, C) + \sum_t \phi_t(x).$$

We prove the following theorem.

**Theorem 4.1.** Given a set of points $X$ in $\mathbb{R}^d$ and a set of centers $C = \{c^1, \ldots, c^k\} \subseteq \mathbb{R}^d$, Algorithm 1 finds a threshold tree $T$ with expected $k$-medians in $\ell_1$ cost at most

$$\mathbb{E}[\text{cost}_{\ell_1}(X, T)] \leq O(\log^2 k) \cdot \text{cost}_{\ell_1}(X, C).$$

Moreover, the same bound holds for the cost of every point $x \in X$ i.e.,

$$\mathbb{E}[\text{cost}_{\ell_1}(x, T)] \leq O(\log^2 k) \cdot \text{cost}_{\ell_1}(x, C).$$

**Proof.** Let $T_t$ be the threshold tree constructed by Algorithm 1 before iteration $t$. Consider a point $x$ in $X$. If $x$ is separated from its original center in $C$ by the cut generated at iteration $t$, then $x$ will be eventually assigned to some other center in the same leaf of $T_t$. By the triangle inequality, the new cost of $x$ at the end of the algorithm will be at most $\text{cost}_t(x, C) + D_t$, where $D_t$ is the maximum diameter of any leaf in $T_t$ (see Algorithm 1). Define a penalty function $\phi_t(x)$ as follows: $\phi_t(x) = D_t$ if $x$ is separated from its original center $c$ at time $t$; $\phi_t(x) = 0$, otherwise. Note that $\phi_t(x) \neq 0$ for at most one iteration $t$, and

$$\text{alg}_{\ell_1}(x) \leq \text{cost}_{\ell_1}(x, C) + \sum_t \phi_t(x).$$

The sum in the right hand side is over all iterations of the algorithm. We bound the expected penalty $\phi_t(x)$ for each $t$.

**Lemma 4.2.** The expected penalty $\phi_t(x)$ is upper bounded as follows:

$$\mathbb{E}[\phi_t(x)] \leq \mathbb{E}\left[D_t \cdot \int_{\Omega} |\delta_x(\omega) - \delta_c(\omega)| \cdot \frac{1\{\omega \in R_t\}}{\mu(R_t)} d\mu(\omega)\right].$$

where $c$ is the closest center to the point $x$ in $C$; $1\{\omega \in R_t\}$ is the indicator of the event $\omega \in R_t$.

**Proof.** If $x$ is already separated from its original center $c$ at iteration $t$, then $\phi_t(x) = 0$. Otherwise, $x$ and $c$ are separated at iteration $t$ if for the random pair $\omega_t = (i, \theta)$ chosen from $R_t$ in Algorithm 1, we have $\delta_x(\omega_t) \neq \delta_c(\omega_t)$. Write,

$$\mathbb{E}[\phi_t(x)] \leq \mathbb{E}\left[\mathbb{P}_{\omega_t}[\delta_x(\omega_t) \neq \delta_c(\omega_t) \mid T_t] \cdot D_t\right].$$

The probability that $\delta_x(\omega_t) \neq \delta_c(\omega_t)$ given $T_t$ is bounded as

$$\mathbb{P}_{\omega_t}[\delta_x(\omega_t) \neq \delta_c(\omega_t) \mid T_t] = \frac{\mu(\omega \in R_t : \delta_x(\omega) \neq \delta_c(\omega))}{\mu(R_t)}$$

$$= \int_{\Omega} 1\{\delta_x(\omega) \neq \delta_c(\omega)\} \cdot \frac{1\{\omega \in R_t\}}{\mu(R_t)} d\mu(\omega)$$

$$= \int_{\Omega} |\delta_x(\omega) - \delta_c(\omega)| \cdot \frac{1\{\omega \in R_t\}}{\mu(R_t)} d\mu(\omega).$$

\qed
Let \( W_t(\omega) = \frac{D_t \cdot \mathbb{1}\{\omega \in R_t\}}{\mu(R_t)} \).

Then, by Lemma 4.2 and inequality (8), we have
\[
\mathbb{E}[\text{alg}_t(x)] \leq \text{cost}_t(x, C) + \mathbb{E}\left[ \sum_t \int \Omega |\delta_x(\omega) - \delta_c(\omega)| \cdot W_t(\omega) \, d\mu(\omega) \right].
\]

The upper bound on the expected cost of \( x \) in tree \( T \) consists of two terms: The first term is the original cost of \( x \). The second term is a bound on the expected penalty incurred by \( x \). We now bound the second term as \( O(\log^2 k) \cdot \text{cost}_t(x, C) \).

\[
\mathbb{E}\left[ \sum_t \int \Omega |\delta_x(\omega) - \delta_c(\omega)| \cdot W_t(\omega) \, d\mu(\omega) \right] = \int \Omega |\delta_x(\omega) - \delta_c(\omega)| \cdot \mathbb{E}\left[ \sum_t W_t(\omega) \right] \, d\mu(\omega).
\]

By Hölder’s inequality, the right hand side is upper bounded by the following product:

\[
\|\delta_x - \delta_c\| \cdot \max_{\omega \in \Omega} \mathbb{E}\left[ \sum_t W_t(\omega) \right].
\]

The first multiplier in the product exactly equals \( ||x - c||_1 \) (see Equation 7), which, in turn, equals \( \text{cost}_t(x, C) \). Hence, to finish the proof of Theorem 4.1, we need to upper bound the second multiplier by \( O(\log^2 k) \).

**Lemma 4.3.** For all \( \omega \in \Omega \), we have
\[
\mathbb{E}\left[ \sum_t W_t(\omega) \right] \leq O(\log^2 k).
\]

**Proof.** Let \( t' \) be the first iteration and \( t'' \) be the last iteration for which \( W_t(\omega) > 0 \). First, we prove that \( D_{t''} \geq D_t / k^3 \), where \( D_t \) and \( D_{t''} \) are the maximum cluster diameters at iterations \( t \) and \( t'' \), respectively. Since \( W_{t'}(\omega) > 0 \) and \( W_{t''}(\omega) > 0 \), we have \( \{\omega \in R_{t'}\} \neq 0 \) and \( \{\omega \in R_{t''}\} \neq 0 \). Hence, \( \omega \in R_{t'} \) and \( \omega \in R_{t''} \). Since \( \omega \in R_{t''} \), there exists a pair \( (i, j) \in E_{t''} \) for which \( \omega \in S_{ij} \). For this pair, we have \( D_{t''} \geq \mu(S_{ij}) \). Observe that the pair \( (i, j) \) also belongs to \( E_t \), since \( E_{t''} \subseteq E_t \). Moreover, \( \mu(S_{ij}) > D_t / k^3 \), because otherwise, \( S_{ij} \) would be included in \( B_t \) (see Algorithm 1) and, consequently, \( \omega \) would not belong to \( R_{t'} = A_t \setminus B_t \). Thus
\[
D_{t''} \geq \mu(S_{ij}) > D_t / k^3.
\]

By the definition of \( t' \) and \( t'' \), we have
\[
\sum_t W_t(\omega) = \sum_{t=t'}^{t''} W_t(\omega) \leq \sum_{t=t'}^{t''} \frac{D_t}{\mu(R_t)}.
\]

Note that the largest distance \( D_t \) is a non-increasing (random) function of \( t \). Thus, we can split the iterations of the algorithm \( \{t', \ldots, t''\} \) into \( \lceil 3 \log k \rceil \) phases. At phase \( s \), the maximum diameter \( D_t \) is in the range \( (D_t/2^{s+1}, D_t/2^s) \). Denote the set of all iterations in phase \( s \) by \( \text{Phase}(s) \).

Consider phase \( s \). Let \( D = D_t / 2^s \). Phase \( s \) ends when all sets \( S_{ij} \) with \( \mu(S_{ij}) \geq D/2 \) are removed from the set \( E_t \). Let us estimate the probability that one such set \( S_{ij} \) is removed from \( E_t \) at iteration \( t \). Set \( S_{ij} \) is removed from \( E_t \) if the random threshold cut \( \omega_t \) chosen at iteration \( t \) separates centers \( c_i \) and \( c_j \), or, in other words, if \( w_t \in S_{ij} \). The probability of this event equals:
\[
\mathbb{P}[\omega_t \in S_{ij} | T_t] = \frac{\mu(S_{ij} \cap R_t)}{\mu(R_t)} = \frac{\mu(S_{ij}) - \mu(S_{ij} \cap B_t)}{\mu(R_t)} \geq \frac{\mu(S_{ij}) - \mu(B_t)}{\mu(R_t)}.
\]

Note that \( \mu(S_{ij}) > D/2 \geq D_t/2 \) and \( \mu(B_t) < \left( \frac{k}{D_t} \right) \cdot \frac{4}{D_t} < \frac{4}{2^s} \) (because \( B_t \) is the union of at most \( \binom{k}{2} \) sets of measure at most \( D_t / k^3 \) each). Hence,
\[
\mathbb{P}[\omega_t \in S_{ij} | T_t] \geq \frac{D_t}{4\mu(R_t)} \geq \frac{1}{4} W_t(\omega).
\]

If \( W_t(\omega) \) did not depend on \( t \), then we would argue that each set \( S_{ij} \) (with \( \mu(S_{ij}) \geq D/2 \)) is removed from \( E_t \) at most \( 4/W_t(\omega) \) iterations, in expectation, and, consequently, all sets \( S_{ij} \) are removed in at most \( O(\log k) \cdot 4/W_t(\omega) \) iterations, in expectation (note that the number of sets \( S_{ij} \) is upper bounded by \( \binom{k}{2} \)). Therefore,
\[
\mathbb{E}\left[ \sum_{t \in \text{Phase}(s)} W_t(\omega) \right] \leq O(\log k) \cdot \frac{4}{W_t(\omega)} \cdot W_t(\omega) = O(\log k).
\]

However, we cannot assume that \( W_t(\omega) \) is a constant. Instead, we use the following claim with \( E = \{0, \ldots, k-1\} \times \{0, \ldots, k-1\}, E_t' = \{(i, j) \in E_t : \mu(S_{ij}) \geq D/2\}, \) and \( p_t = W_t(\omega)/4 \).

**Claim 4.4.** Consider two stochastic processes \( E_t \) and \( p_t \) adapted to filtration \( \mathcal{F}_t \). The values of \( E_t \) are subsets of some finite non-empty set \( E \). The values of \( p_t \) are numbers in \([0, 1]\). Suppose that for every step \( t, E_{t+1} \subseteq E_t \) and for every \( e \in E_t, \mathbb{P}[e \notin E_{t+1} | \mathcal{F}_t] \geq p_t \). Let \( \tau \) be the (stopping) time \( t \) when \( E_t = \emptyset \). Then,
\[
\mathbb{E}\left[ \sum_{t=0}^{\tau-1} p_t \right] \leq \ln |E| + O(1).
\]

We prove this claim in Appendix A.
By Claim 4.4,
\[
E\left[ \sum_{t \in \text{Phase}(s)} W_t(\omega) \right] \leq O(\log k).
\]

The expected sum of \(W_t\) over all phases is upper bounded by \(O(\log^2 k)\), since the number of phases is upper bounded by \(O(\log k)\). We note that if the number of phases is upper bounded by \(L\), then the expected sum of \(W_t\) over all phases is upper bounded by \(O(L \log k)\). This concludes the proofs of Lemma 4.3 and Theorem 4.1. \(\square\)

5. Improved Analysis for \(k\)-medians in \(\ell_1\)

In this section, we provide an improved analysis of our algorithm for \(k\)-medians in \(\ell_1\).

**Theorem 5.1.** Given a set of points \(X\) in \(\mathbb{R}^d\) and set of centers \(C = \{c^1, \ldots, c^k\} \subset \mathbb{R}^d\), Algorithm 1 finds a threshold tree \(T\) with expected \(k\)-medians \(\ell_1\) cost at most
\[
E[\text{cost}_{\ell_1}(X, T)] \leq O(\log k \log \log k) \cdot \text{cost}_{\ell_1}(X, C).
\]

**Proof.** In the proof of Theorem 4.1, we used a pessimistic estimate on the penalty a point \(x \in X\) incurs when it is separated from its original center \(c\). Specifically, we bounded the penalty by the maximum diameter of any leaf in the tree \(T_t\). In the current proof, we will use an additional bound: The distance from \(x\) to the closest center after separation. Suppose, that \(x\) is separated from its original center \(c\). Let \(c'\) be the closest center to \(x\) after we make cut \(\omega_t\) at step \(t\). That is, \(c'\) is the closest center to \(x\) in the same leaf of the threshold tree \(T_{t+1}\). Note that after we make additional cuts, \(x\) may be separated from its new center \(c'\) as well, and the cost of \(x\) may increase. However, as we already know, the expected cost of \(x\) may increase in at most \(O(\log^2 k)\) times in expectation (by Theorem 4.1). Here, we formally apply Theorem 4.1 to the leaf where \(x\) is located and treat \(c'\) as the original center of \(x\). Therefore, if \(x\) is separated from \(c\) by a cut \(\omega_t\) at step \(t\), then the expected cost of \(x\) in the end of the algorithm is upper bounded by
\[
E[\text{alg}_{\ell_1}(x) \mid T_t, \omega_t] \leq O(\log^2 k) \cdot \|c' - x\|_1 \quad (10)
\]
\[
= O(\log^2 k) \cdot D_t^{\min}(x, \omega_t).
\]

In the formula above, we used the following definition: \(D_t^{\min}(x, \omega)\) is the distance from \(x\) to the closest center \(c'\) in the same leaf of \(T_t\) as \(x\) which is not separated from \(x\) by the cut \(\omega\). Note that in this case, our algorithm will never make cut \(\omega\), since it always makes sure that the both parts of the cut contain at least one center from \(C\). Similarly to \(D_t^{\min}(x, \omega)\), we define \(D_t^{\max}(x, \omega)\): \(D_t^{\max}(x, \omega)\) is the distance from \(x\) to the farthest center \(c''\) in the same leaf of \(T_t\) as \(x\) which is not separated from \(x\) by the cut \(\omega\). We also let \(D_t^{\max}(x, \omega) = 0\) if there is no such \(c''\). Note that \(D_t^{\max}(x, \omega)\) is an upper bound on the cost of \(x\) in the eventual threshold tree \(T\) if cut \(\omega\) separated \(x\) from \(c\) at step \(t\).

We now have three bounds on the expected cost of \(x\) in the final tree \(T\) given that the algorithm separates \(x\) from its original center \(c\) at step \(t\) with cut \(\omega\). The first bound is \(D_t^{\max}(x, \omega)\); the second bound is \(O(\log^2 k) \cdot D_t^{\min}(x, \omega)\), and the third bound is \(\|x - c\|_1 + D_t\). We use the first bound if \(D_t^{\max}(x, \omega) \leq 2\|x - c\|_1\). We call such cuts \(\omega\) light cuts. We use the second bound if \(D_t^{\max}(x, \omega) > 2\|x - c\|_1\) but \(D_t^{\max}(x, \omega) \leq D_t/\log^4 k\). We call such cuts \(\omega\) medium cuts. We use the third bound if \(D_t^{\max}(x, \omega) > 2\|x - c\|_1\) and \(D_t^{\max}(x, \omega) > D_t/\log^4 k\). We call such cuts \(\omega\) heavy cuts.

Note that in the threshold tree returned by the algorithm, one and only one of the following may occur: (1) \(x\) is separated from the original center \(c\) by a light, medium, or heavy cut; (2) \(x\) is not separated from \(c\). We now estimate expected penalties due to light, medium, or heavy cuts.

If the algorithm makes a light cut, then the maximum cost of point \(x\) in \(T\) is at most \(2\|x - c\|_1 = 2\text{cost}_{\ell_1}(x, C)\). So we should not worry about such cuts. If the algorithm makes a medium cut, then the expected additional penalty for \(x\) is upper bounded by
\[
D_t^{\min}(x, \omega_t) \cdot O(\log^2 k) \leq O(\phi_t(x)/\log^2 k),
\]
where \(\phi_t(x)\) is the function from the proof of Theorem 4.1. Thus, the total expected penalty due to a medium cut (added up over all steps of the algorithm) is \(\Omega(\log^2 k)\) times smaller than the penalty we computed in the proof of Theorem 4.1. Therefore, the expected penalty due to a medium cut is at most \(O(\|x - c\|_1)\).

We now move to heavy cuts. Denote the set of possible heavy cuts for \(x\) in \(R_t\) by \(H_t\). That is, if \(x\) is not separated from its original center \(c\) by step \(t\), then
\[
H_t = \left\{ \omega \in R_t : D_t^{\max}(x, \omega) > D_t/\log^4 k \text{ and } D_t^{\max}(x, \omega) > 2\|x - c\|_1 \right\}.
\]

Otherwise, let \(H_t = \emptyset\). Define a density function \(\widetilde{W}_t(\omega)\) similarly to \(W_t(\omega)\):
\[
\widetilde{W}_t(\omega) = \frac{D_t \cdot 1_{\{\omega \in H_t\}}}{\mu(R_t)}.
\]

Then, the expected penalty due to a heavy cut is bounded, similarly to Lemma 4.2, by
\[
\sum_{t} E\left[ \int_{0}^{1} |\delta_x(\omega) - \delta_{c'}(\omega)| \cdot \widetilde{W}_t d\mu(\omega) \right].
\]
Therefore, to finish the proof of Theorem 1.1, we need to prove the following analog of Lemma 4.3.

**Lemma 5.2.** For all \( \omega \in \Omega \), we have
\[
E \left[ \sum_i \tilde{W}_i(\omega) \right] \leq O(\log k \log \log k).
\]

**Proof.** As in the proof of Lemma 4.3, consider the first and last steps when \( \tilde{W}_i(\omega) > 0 \). Denote these steps by \( t^\ast \) and \( t^{**} \), respectively. In the proof of Lemma 4.3, we had a bound \( D_{t^\ast} \geq D_{t^\ast}/k^3 \) (see inequality (9)). We now show a stronger bound on \( t^\ast \) and \( t^{**} \).

**Claim 5.3.** We have \( D_{t^{**}} \geq D_{t^\ast}/2 \log^4 k \).

This claim implies that the number of phases defined in Lemma 4.3 is bounded by \( O(\log log k) \), which immediately implies Lemma 5.2. So, to complete the proof, it remains to show Claim 5.3.

**Proof of Claim 5.3** First, note that \( \mathbb{1}\{\omega \in H_{t^\ast}\} > 0 \) and, consequently, \( \tilde{c} \) is a heavy at step \( t^{**} \). Thus, \( D_{t^{**}}(x, \omega) \) is positive. Hence, this cut separates \( c \) from at least one other center \( c' \) in the same leaf of the current threshold tree \( T_{t^\ast} \). Let \( c'' \) be the farthest such center from point \( x \). Then, \( \|c'' - x\|_1 = D_{t^{**}}(x, \omega) \). Since centers \( c \) and \( c'' \) are not separated prior to step \( t^{**} \), we have
\[
D_{t^\ast} \geq \|c - c''\|_1 \geq \|c'' - x\|_1 - \|x - c\|_1.
\]

Since \( \omega \) is a heavy cut and not a light cut, \( \|x - c''\|_1 > 2\|x - c\|_1 \). Thus,
\[
D_{t^\ast} \geq \|x - c''\|_1 \geq D_{t^{**}}(x, \omega) / 2.
\]

Since \( D_{t^{**}}(x, \omega) \) is non-decreasing (for fixed \( x \) and \( \omega \)) since the distance from \( x \) to the closest center \( c'' \) cannot decrease over time. Therefore,
\[
D_{t^{**}} \geq D_{t^{**}}(x, \omega) / 2 \geq D_{t^\ast} / 2 \log^4 k.
\]

In the last inequality, we used that \( \omega \) is a heavy cut at time \( t^\ast \). This finishes the proof of Claim 5.3.

### 6. Terminal Embedding of \( \ell^2_2 \) into \( \ell_1 \)

In this section, we show how to construct a coordinate cut preserving terminal embedding of \( \ell^2_2 \) into \( \ell_1 \) with distortion \( O(k) \) for every set of terminals \( K \subset \mathbb{R}^d \) of size \( k \).

Let \( K \) be a finite subset of points in \( \mathbb{R}^d \). We say that \( \varphi : x \mapsto \varphi(x) \) is a terminal embedding of \( \ell^2_2 \) into \( \ell_1 \) with a set of terminals \( K \) and distortion \( \alpha \) if for every terminal \( y \in K \) and every point \( x \in \mathbb{R}^d \), we have
\[
\| \varphi(x) - \varphi(y) \|_1 \leq \| x - y \|_2^2 \leq \alpha \cdot \| \varphi(x) - \varphi(y) \|_1.
\]

**Lemma 6.1.** For every finite set of terminals \( K \) in \( \mathbb{R}^d \), there exists a coordinate cut preserving terminal embedding of \( \ell^2_2 \) into \( \ell_1 \) with distortion \( 8|K| \).

**Proof.** We first prove a one dimensional analog of this theorem (which corresponds to the case when all points and centers are in one dimensional space).

**Lemma 6.2.** For every finite set of real numbers \( K \), there exists a cut preserving embedding \( \psi_K : \mathbb{R} \to \mathbb{R} \) such that for every \( x \in \mathbb{R} \) and \( y \in K \), we have
\[
|\psi_K(x) - \psi_K(y)| \leq |x - y|^2 \leq 8|K| \cdot |\psi_K(x) - \psi_K(y)|.
\]

**Proof.** Let \( k \) be the size of \( K \) and \( y_1, \ldots, y_k \) be the elements of \( K \) sorted in increasing order. We first define \( \psi_K \) on points in \( K \) and then extend this map to the entire real line \( \mathbb{R} \). We map each \( y_i \) to \( z_i \) defined as follows: \( z_1 = 0 \) and for \( i = 2, \ldots, k \),
\[
z_i = 1/2 \sum_{j=1}^{i-1} (y_{j+1} - y_j)^2.
\]

Now consider an arbitrary number \( x \in \mathbb{R} \). Let \( y_i \) be the closest point to \( x \) in \( K \). Let \( \varepsilon_x = \text{sign}(x - y_i) \). Then,
\[
x = y_i + \varepsilon_x |x - y_i|.
\]

Note that \( \varepsilon_x = 1 \) if \( x \) is on the right to \( y_i \), and \( \varepsilon_x = -1 \), otherwise. Let the function \( \psi_K \) be
\[
\psi_K(x) = z_i + \varepsilon_x |x - y_i|^2.
\]

For \( x = (y_i + y_{i+1})/2 \), both \( y_i \) and \( y_{i+1} \) are the closest points to \( x \) in \( K \). In this case, we have
\[
z_i + \varepsilon_x |x - y_i|^2 = z_{i+1} + \varepsilon_x |x - y_{i+1}|^2,
\]

which means \( \psi_K(x) \) is well-defined. An example of the terminal embedding function \( \psi_K(x) \) is shown in Figure 2. We show that this function \( \psi_K \) is a cut preserving embedding satisfying inequality (11) in Lemma B.1.

Using the above lemma, we can construct a terminal embedding \( \psi \) from \( d \)-dimensional \( \ell^2_2 \) into \( d \)-dimensional \( \ell_1 \) as follows. For each coordinate \( i \in \{1, 2, \ldots, d\} \), let \( K_i \) be the set of the \( i \)-th coordinates for all terminals in \( K \). Define one dimensional terminal embeddings \( \psi_i \) for all coordinates \( i \). Then, \( \psi \) maps every point \( x \in \ell^2_2 \) to \( \psi(x) = (\psi_1(x), \ldots, \psi_d(x)) \). We show that this terminal embedding \( \psi \) is coordinate cut preserving in Lemma B.2.

**For explainable k-means clustering, we first use the terminal embedding of \( \ell^2_2 \) into \( \ell_1 \). Then, we apply Algorithm 1 to the instance after the embedding. By using this terminal embedding, we can get the following result.**
Theorem 6.3. Given a set of points $X$ in $\mathbb{R}^d$ and a set of centers $C$ in $\mathbb{R}^d$, Algorithm 1 with terminal embedding finds a threshold tree $T$ with expected $k$-means cost at most
\[
E[\text{cost}_{\ell_2}(X,T)] \leq O(k \log k \log \log k) \cdot \text{cost}_{\ell_2}(X,C).
\]

Proof. Let $\varphi$ be the terminal embedding of $\ell_2^d$ into $\ell_1$ with terminals $C$. Let $T'$ be the threshold tree returned by our algorithm on the instance after embedding. Since the terminal embedding $\varphi$ is coordinate cut preserving, the threshold tree $T'$ also provides a threshold tree $T$ on the original $k$-means instance. Let $\varphi(C)$ be the set of centers after embedding. For any point $x \in X$, the expected cost of $x$ is at most
\[
E[\text{cost}_{\ell_2}(x,T)] \leq 8k \cdot E[\text{cost}_{\ell_1}(\varphi(x),T')]
\leq O(k \log k \log \log k) \cdot \text{cost}_{\ell_1}(\varphi(x),\varphi(C))
\leq O(k \log k \log \log k) \cdot \text{cost}_{\ell_2}(x,C),
\]
where the first and third inequality is from the terminal embedding in Lemma 6.1 and the second inequality is due to Theorem 5.1.

Acknowledgements

Konstantin Makarychev and Liren Shan were supported by NSF Awards CCF-1955351 and CCF-1934931.

References


Dasgupta, S., Frost, N., Moskhovitz, M., and Rashtchian, C. Explainable $k$-means and $k$-medians clustering. In


