k−Clustering with Fair Outliers

Matteo Almanza
almanza@di.uniroma1.it
Sapienza University
Rome, Italy

Alessandro Epasto
aepasto@google.com
Google Research
New York, USA

Alessandro Panconesi
ale@di.uniroma1.it
Sapienza University
Rome, Italy

Giuseppe Re
re@di.uniroma1.it
Sapienza University
Rome, Italy

Abstract

Clustering problems and clustering algorithms are often overly sensitive to the presence of outliers: even a handful of points can greatly affect the structure of the optimal solution and its cost. This is why many algorithms for robust clustering problems have been formulated in recent years. These algorithms discard some points as outliers, excluding them from the clustering. However, outlier selection can be unfair: some categories of input points may be disproportionately affected by the outlier removal algorithm.

We study the problem of k-clustering with fair outlier removal and provide the first approximation algorithm for well-known clustering formulations, such as k-means and k-median. We analyze this algorithm and prove that it has strong theoretical guarantees. We complement this result with an empirical evaluation showing that, while standard methods for outlier removal have a disproportionate impact across categories of input points, our algorithm equalizes the impact while retaining strong experimental performances on multiple real–world datasets. We also show how the fairness of outlier removal can influence the performance of a downstream learning task. Finally, we provide a coreset construction, which makes our algorithm scalable to very large datasets.

CCS Concepts

• **Theory of computation** → **Facility location and clustering**;
• **Computing methodologies** → *Cluster analysis.*

Keywords

Clustering; outliers; fairness; k−means; disparate impact; coreset.

ACM Reference Format:
Matteo Almanza, Alessandro Epasto, Alessandro Panconesi, and Giuseppe Re. 2022. k−Clustering with Fair Outliers. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining (WSDM '22), February 21–25, 2022, Tempe, AZ, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3488560.3498485

WSDM 2022, February 21–25, 2022, Phoenix, AZ, USA

ACM ISBN 978-1-4503-9132-0/22/02…$15.00
https://doi.org/10.1145/3488560.3498485

1 Introduction

Clustering is a fundamental data analysis and machine learning problem, with more than 50 years of history [41] and myriad applications in computer science and beyond. Clustering techniques lie at the core of solutions to many challenges in disciplines as disparate as biology, physics and medicine (see, *e.g.*, [67]). Given this wide applicability, it is not surprising that various clustering formulations have been proposed, including the well-known metric k-clustering problems (such as k-center, k-median, and k-means) [6, 8, 16, 20, 24, 28, 34].

A central issue in clustering is that real-world data can be noisy: a handful of points, called *outliers*, can vastly influence in a negative way the structure and cost of the final clustering solution. In particular, in k-clustering problems, standard approximation algorithms such as k-means++ [6] may be sensitive to outliers and predominantly select them as centers, resulting in a set of centers that are far from most of the points.

This is why the problem of clustering with *outliers*, or *robust* clustering, was introduced in the seminal work of Charikar et al. [21] and extensively studied ever since. In this formulation, the algorithm is requested to remove from the input a fixed number of points, called *outliers*. The quality of the centers computed in output is then evaluated exclusively on the non-removed points, the *inliers*. More precisely, in k-median with outliers, one seeks to find k centers and z outliers out of n points in a metric space in a way that minimizes the sum of distances of the $(n - z)$ remaining inliers to the nearest center (k-center and k-means with outliers are defined analogously). Many approximation algorithms for such problems have been developed and analyzed [16, 20, 24, 28, 34].

Virtually, all prior work on clustering with outliers has ignored the issue of *fairness*, *i.e.*, the fact that outlier removal should not affect different categories disproportionately. The fairness of machine learning systems is an important concern that has recently been brought to the fore [25, 26, 44, 45, 47, 61]. In the context of clustering, fairness is often intended in terms of preventing disparate impact over categories of data points. In the seminal work of Chierichetti et al. [25], the input points are assumed to have a class label, called *color*, representing a sensitive or protected class of the data point (e.g., the demographic group of a user) or a category for which diversity requirements are considered (e.g., a viewpoint in a blog post). The most studied formulation in fair clustering [3, 4, 7, 12, 25, 60] seeks to constraint each cluster in output to faithfully represent the color distribution of the data points in the overall input.

In this paper, we focus on a different and complementary problem to the one addressed in such papers: preventing disproportionate impact in outlier removal. As standard methods for clustering with outliers completely ignore the categories of the data points, it is possible, or even likely, that a given group may be disproportionately impacted by outlier removal. For instance, in a dataset containing images of users or blog posts, outlier removal may disproportionately affect certain groups introducing undesirable biases in the solution. In an extreme case, an entire class of points could be removed. This is problematic when clustering is used for data aggregation and summarization [46], identification of representative data points [56], experiment design [58], and many other applications. The need to ensure fairness in outlier removal is further exacerbated when outliers are used to flag anomalies in data [59, 66].

Given the above motivation, in this paper we formulate and study the problem of *$k-$clustering with fair outliers*, defined as follows. We are given as input a set X of points in a metric space partitioned into c classes called *colors*, an integer parameter $k > 0$, a parameter $p \geq 1$, and, for each color a, an integer z_a representing the number of points of that color to be removed as outliers. The removed points are called *outliers* and the remaining ones *inliers*. We want to find k centers such that, after removal of the outliers, the ℓ_p norm of the vector of the distances of every inlier to the closest center is minimized. Notice that this definition of fairness, given the flexibility of the choice of z_a, allows preventing disparate impact on the set of outliers. In fact, as a special case, this includes the well-known *demographic parity* fairness [11] on the set of outliers: we can ensure that the number of points of color a deemed outliers is proportional to the number of points of color a. We refer to the related work for an in-depth discussion of fairness guarantees.

This problem has been studied only in the case of k-center [5, 10, 42], which corresponds to the ℓ_∞-norm case and is equivalent to minimizing the maximum distance of a point to a center. In this paper, we study it for every finite ℓ_p-norm, including the classical $k-$median problem (ℓ_1-norm) and $k-$means problem (ℓ_2-norm squared). As it is well known, solving k-means type formulations, as opposed to k-center ones, requires radically different algorithmic techniques from prior research on this problem.

As a preliminary step, we prove empirically that the best algorithms for clustering with outliers are *not* fair. Indeed, as our experiments show, these algorithms can compute solutions with high disparate impact in the sets of outliers, providing a strong motivation for studying the problem we formulated.

As a first theoretical result, we show that it is NP-hard to approximate our problem within *any* approximation factor. This forces us to study its bi-criteria version, where we are allowed to relax the bound on the outliers. Then, we design a novel bi-criteria algorithm which we name FairClout (for *Fair*ness in *Cl*ustering with *Out*liers), and we show that it computes a constant approximation of the optimum by removing, for every color a, $O(kz_a)$ points, where k is the number of centers. This worst-case bound is improved to $2z_a$ under the assumption that the optimal clusters have all colors well represented. Our bi-criteria algorithm can always enforce demographic parity on the outliers, as the bi-criteria factor is applied uniformly to all colors, and it has a simple, modular structure. It consists of a filtering step that identifies candidate outliers after which any $k-$clustering approximation algorithm can be deployed: if an α-approximation algorithm is used, the solution is guaranteed to be a fair $O(\alpha)$-approximation. FairClout is built upon the work of [38] on clustering (without fairness), but our analysis is considerably more involved as we need to deal with the fairness constraint. We also provide a coreset construction for this problem, making our algorithm scalable to large datasets.

Then, from the theory side, we initiate a preliminary study of combining outlier fairness with known fairness requirements for clustering. We provide theoretical evidence that imposing other fairness guarantees in addition to fair outlier removal makes the problem inherently harder than addressing either problem independently. A careful exploration of the trade-offs between cost and multiple fairness constraints is called for to properly address the issue, and is left for future work.

Finally, we perform an experimental analysis to assess the behaviour of FairClout, showing that it is efficient and effective with real-world datasets. In particular, we show that our algorithm can ensure fairness in outlier detection with minimal increase in the cost of the solution with respect to non-fair algorithms. We also show how the fairness of outlier removal can influence the performance of a downstream learning task. To enable the replication of our study we provide the source code on `github`.[1]

Road map. The rest of the work proceeds as follows. In Section 2 we review the related work; in Section 3 we introduce the necessary formalism; in Section 4 we prove hardness results for our problem; in Section 5 we detail FairClout; in Section 6 we show a coreset construction for it; in Section 7 we show that combining different notions of fairness incurs complex trade-offs; in Section 8 we lay out an empirical evaluation and we draw some conclusions in Section 9.

2 Related Work

Our work touches upon the well-established area of metric space clustering (in particular, clustering with outliers) as well as the burgeoning area of fairness in unsupervised machine learning. Given the large literature on these topics, we only review them briefly.

k-clustering in metric spaces. One of the most popular classes of problems in clustering is k-clustering in metric spaces, which seeks to find k centers that minimize an objective consisting of the ℓ_p norm of the distances of all points to their closest center. For $p \in \{1, 2, \infty\}$, respectively, this problem corresponds to k-median, k-means, and k-center. All such problems are NP-hard but a long stream of results have provided constant factor approximation algorithms for them (see [19, 64]). Significant work has been devoted to providing scalable algorithms for such problems including studying stream and distributed variants of the problem [6, 8, 49].

Clustering with outliers. The work on clustering with outliers, or robust clustering, was spearheaded by Charikar et al. [21], who provided efficient algorithms for k-center, facility location, and $k-$median with outliers. The results for $k-$median were later improved by Chen [24] and extended to $k-$means by Chawla and Gionis [22]. There have been various approaches to these problems, from local search techniques [32, 43] to variations of the k-means++ algorithm by Arthur and Vassilvitskii [6] [16, 27]. Most relevant to our work is the result of Im et al. [38], which provided a constant

[1] https://github.com/matteojug/kclustering-with-fair-outliers

approximation algorithm for k-means with (non-fair) outliers by applying a noise-removal pre-processing procedure before choosing the clusters' centers. In our paper, we build upon this pre-processing methodology by extending this approach to the *fair* outlier case for arbitrary finite $k-$clustering problems. Related lines of work also include constructing coresets for k-clustering problems with outliers [23, 34] and studying online version of clustering with outliers [15]. Notice that none of the previously mentioned works deal with fairness in outlier removal and, as we show in our experiments, using such algorithms can result in disproportionate impact.

Fairness in machine learning. Fairness in machine learning is a recent area of study (see [11, 44]). Fairness in *supervised* machine learning is well studied (e.g. [36]), however, we notice that standard notions of fairness for this settings (e.g., *equality of opportunity* and *equalized odds* [36]) do not apply to our work which is inherently *unsupervised*, there is no ground-truth on outliers or clusters.

Fairness of clustering. The study of fairness for unsupervised clustering was jump-started by the work of Chierichetti et al. [25] on *fair clustering*, defined as clustering points such that every cluster has at least a certain fraction of input data points from each category. This work has been extended in different directions, including defining more efficient algorithms [7, 60], more general constraints [3, 12, 33], or generalizing beyond the metric space case [4]. Notice that this notion of fairness induces a different problem (balanced color distribution of clusters) to the one we address in this paper (balanced color distribution of outliers). Studying *fair clustering* with *fair outliers* is beyond the scope of the paper and, as we show in Section 7, it is inherently harder than either problem independently. Another notion of fairness for clustering was recently proposed by Abbasi et al. [1], requiring balancedness for the average cost of points belonging to different categories. However, this notion may be ill-defined for a method with outliers. A notion of individual fairness in clustering has also been proposed [52].

Fairness of outlier removal. Our work focuses on the need for fairness and representativity constraints in the removal of outliers in clustering. We aim to avoid the undesired phenomenon of *disparate impact* [31] in the selection of outliers. Our fairness definition (see Section 3) is quite flexible and allows to enforce popular fairness constraints such as *demographic parity* [11], *calibration within groups* [57], *statistical parity* [30], diversity rules (e.g., 80%-rule) [18], and proportional representation rules [54].

A relevant work in this setting is the one of Bandyapadhyay et al. [10], which introduced the problem of k-center with fair outliers (they named it *colorful* k-center). For the general metric case, they provided a bi-criteria 2-approximation that outputs more centers and is based on LP methods. They also provided an $O(1)$-approximation algorithm in the euclidean plane in $n^{\Omega(c)}$ time for a generic number c of colors, where n is the input size. They leave as an open problem whether a proper (as opposed to bi-criteria) $O(1)$-approximation is achievable in euclidean or arbitrary metric space. In this regard, Anegg et al. [5] proved that it is NP-hard to provide any approximation for the colorful k-center problem with a generic number of colors and that the same is true under the Exponential Time Hypothesis [39] if $c = \omega(\log(n))$. We extend this result to k-median and k-means as well. They also came up with a 4-approximation algorithm running in time $n^{\Omega(c)}$, later improved to a 3-approximation by Jia et al. [42]. Notice that none of these works apply to the k-means or k-median clustering formulations, which arguably see more practical use than k-center, nor do they provide a combinatorial algorithm with polynomial running time for any number of colors. Finally, fair outlier removal was also studied for priority k-center [9] and for other problems than k-clustering [13, 37, 40, 51]. In particular, Inamdar and Varadarajan [40] provided an approximation algorithm for the set cover problem and, from that, derived a $O(\log(c))$-approximation algorithm for facility location with fair outliers with c colors. They also proved an $\Omega(\log(c))$ approximability lower bound for this problem. None of these results apply to general k-clustering with fair outliers.

3 Preliminaries

We consider clustering problems with fair outliers in a generic metric space (X, d), where d is a distance function over the set of input points X. In any clustering problem we define, X is colored, i.e. it is partitioned into disjoint subsets that represent the points' categories. Formally, we have a finite set $\mathcal{A}$ of possible colors with $|\mathcal{A}| = c \geq 2$. X is partitioned into c disjoint subsets $\{X_a\}_{a\in\mathcal{A}}$, where X_a is the set of vertices with color a. We also define $n_a := |X_a|$.

We are allowed to discard some outliers from the clustering solution, but we impose limitations on the number of outliers that can belong to each subset. This is intended to prevent disparate impact in the outlier detection process. These constraints are expressed through a vector of integers $\boldsymbol{z} = (z_a)_{a\in\mathcal{A}}$: for each color a, it holds $0 \leq z_a \leq n_a$, and we are required to discard z_a outliers from X_a. The vector $\boldsymbol{z}$ can be an arbitrary integer vector to model any fairness constraint desired. This comprehends many possibilities: as a special case, we can fix z_a/n_a to be equal (up to rounding due to z_a being an integer) across different colors $a \in \mathcal{A}$, achieving *demographic parity* or *calibration within groups*.

Any solution consisting of a set of k centers C, will partition the input set into *outliers* and *inliers*. The former are z_a points of color a from C for each color $a \in \mathcal{A}$. They will be excluded from the clustering solution and they will not affect the value of the cost function. The latter will be grouped into k clusters and they will contribute to the cost of the clustering.

In this paper we study the k-clustering problem, with arbitrary $p \geq 1$ and we assume p to be a fixed constant. The objective function is as follows: given a set C of k centers, the cost of this solution $f^X_{\boldsymbol{z}}(C)$ is defined as the sum, over every inlier x, of $d(x, C)^p$, which is the p-th power of the distance between x and the nearest center in C. Note that the set of outliers (and the the cost $f^X_{\boldsymbol{z}}(C)$) can be determined optimally given a set of centers C. In fact, the set of outliers, which we will denote with $X_{\boldsymbol{z}}(C)$, can be optimally obtained by including the z_a farthest points in X_a from C for each color a (ties broken arbitrarily). The cost of the solution is therefore $f^X_{\boldsymbol{z}}(C) := \sum_{x\in X_{\boldsymbol{z}}(C)} d(x, C)^p$. If C^* is an optimal solution w.r.t $\boldsymbol{z}$, we define $OPT := OPT(X, k, \boldsymbol{z}) = f^X_{\boldsymbol{z}}(C^*)$. We also define $OPT_a := OPT_a(X, k, \boldsymbol{z}) = \sum_{x\in X_{\boldsymbol{z}}(C)\cap X_a} d(x, C^*)^p$ for each color $a \in \mathcal{A}$, where we assume that a fixed optimal solution C^* is chosen.

In this work, we study bi-criteria solutions, which are allowed to discard slightly more than z_a outliers. This is needed, as we show later, to get provable approximation guarantees and it is also reasonable from an experimental perspective, as in real-world

applications one does not usually have strict requirements on the exact number of outliers. We define bi-criteria (α, β)-approximation for k-clustering with fair outliers as obtaining an α-approximation of the optimal solution for z_a outliers per color a, by discarding βz_a outliers instead, $\alpha, \beta \geq 1$. Notice that this preserves demographic parity. Finally, we also use the following notation. Given $x \in X_a, r > 0$, we define the colored ball of radius r around x as $B_a(x, r) := \{y \in X_a \mid d(x, y) \leq r\}$. When discussing general metrics (as opposed to Euclidean metrics), we make the standard assumption that we can compute the distance between a pair of points in constant time.

4 Lower Bounds

First, we show that the problem we are addressing has strong inapproximability bounds forcing us to work with bi-criteria algorithms. This result follows immediately from the recent result of Anegg et al. [5] for $k-$center with fair outliers, and from the fact that distinguishing a non-zero cost solution from a zero-cost solution is equivalent for $k-$center and any $k-$clustering problem.

Theorem 4.1. *For every $p \geq 1$, it is NP-hard to decide whether the $k-$clustering problem with fair outliers on the real line admits a solution of cost 0. Moreover, unless the Exponential Time Hypothesis fails, for any $f : \mathbb{Z}^+ \to \mathbb{Z}^+$ s.t. $f(n) = \omega(\log(n))$, no polynomial algorithm can distinguish whether $k-$clustering with fair outliers on the real line with $c = f(|X|)$ colors admits a solution of cost 0.*

This theorem rules out any proper (as opposed to bi-criteria) polynomial-time approximation algorithm for an arbitrary number of colors for any k-clustering problem with fair outliers. Moreover, under the Exponential Time Hypothesis, such an algorithm does not exist even if we have asymptotically more than $\log(n)$ colors. As a consequence, we can only hope to provide bi-criteria approximation algorithms if we want polynomial algorithms for instances with an arbitrary number of colors. This is what we do next.

5 FairClout: an Algorithm for Clustering with Fair Outliers

In this section, we present our algorithm FairClout for k-clustering with fair outliers. Our work builds upon the recent technique for outliers removal developed by Im et al. [38] for the specific problem of k-means with outliers. Such work does not address fairness considerations in the outlier case, so our algorithm requires special care for handling the color classes and its analysis is considerably more involved. The algorithm is based on a pre-processing step to fairly identify a set of outliers from the data. After the points are removed, any standard (i.e., not fair) algorithm for the classic k-clustering problem without outliers can be applied as a black box on the remaining inliers.

5.1 The algorithm

FairClout, described in Algorithm 1, takes in input the vector of allowed outliers $\boldsymbol{z}$ and the bi-criteria factor $\beta > 1$ and outputs k centers. The algorithm proceeds in three phases: (i) a pre-processing phase for outlier removal, which discards a (tentative) candidate set of outliers; (ii) the application of any $k-$clustering algorithm (*without* outliers) on the non-discarded points. This algorithm identifies the k centers C in output. (iii) The final set of fair outliers is determined optimally (see Section 3) from the centers C in output.

Algorithm 1 FairClout

Input: instance $(X, d), \boldsymbol{z}, p$ of k-clustering with fair outliers, A algorithm for the specific $k-$clustering problem, $\beta > 1$.

for $a \in \mathcal{A}$ **do**
  $d_{a,min} \leftarrow \min_{x_1 \neq x_2 \in X_a} d(x_1, x_2)$
  for $\theta_a = 0, d^p_{a,min}, d^p_{a,min}(1+\epsilon), d^p_{a,min}(1+\epsilon)^2, \ldots$ **do**
    $r_a \leftarrow 2(\theta_a / z_a)^{1/p}$
    for $x \in X_a$ **do**
      if $|B_a(x, r_a)| \geq 2 \cdot z_a$ **then** mark x as *heavy*
    $Y_a \leftarrow \emptyset$
    for $x \in X_a$ **do**
      if $B_a(x, r_a)$ contains no heavy pts **then** $Y_a \leftarrow Y_a \cup \{x\}$
    if $|Y_a| \leq \beta \cdot z_a$ **then break** and use Y_a for color a.
return $C \leftarrow A(X \setminus Y, k)$ (where $Y := \bigcup_{a \in \mathcal{A}} Y_a$)

Surprisingly, we show that the pre-processing phase of our algorithm can be made completely independent across sets with different colors allowing additional parallelization opportunities. Our algorithm applies the procedure independently on each $X_a, a \in \mathcal{A}$, then the selected inliers are merged together and used as input for a generic $k-$clustering algorithm. We observe that this independence is crucial for the efficiency of our algorithm, as we will need to guess approximately the cost of the optimum solution restricted to each color. Since the guess for each color can be verified independently of the other colors' guesses, the pre-processing can be done in polynomial instead of exponential time.

Preliminary outlier removal. We consider the set of points X_a with a fixed color $a \in \mathcal{A}$. Given a point $x \in X_a$, we pick a ball of a certain radius r_a around x. The choice of the radius will be clarified later. If the ball contains $\geq 2z_a$ points with color a, then at least half of those cannot be outliers, so x is not likely to be an outlier in the optimal solution. Points like x are called *heavy*. After identifying heavy points, we choose each point with color a as inlier for the next phase if and only if it has a heavy point with the same color within distance r_a. We will show that, for the right radius, this phase removes less than βz_a points of color a.

Choosing the radius. For each color a, we would like to choose a radius $r_a \sim 2(OPT_a/z_a)^{1/p}$ (recall that OPT_a is the cost incurred by the points of color a in a fixed optimal solution for the entire dataset). Therefore, we need to rely on a good estimate $\theta_a \sim OPT_a$, which we obtain by trying the values in a grid starting at $d^p_{a,min}$ and growing exponentially by $(1+\epsilon)$, where $d_{a,min}$ is the minimum distance between two distinct points in X_a and $\epsilon > 0$ is a fixed constant. To find the right threshold, we choose as θ_a the minimum value that yields the desired number of candidate outliers $\leq \beta z_a$. Before those values, we also try $\theta_a = 0$ to deal with the case $OPT_a = 0$. We will show theoretic guarantees for specific values of β.

Obtaining the centers. We will collect all the non-discarded points with different colors from phase 1 and run a fixed k-clustering algorithm A on this set of points. Note that we are not dealing with outliers anymore at this point as they have already been discarded. The centers returned by the algorithm A will be our output centers.

Final outlier detection. Finally, notice that the output centers define an optimal set of outliers consisting of the βz_a points of color a furthest from the centers. These are the outliers identified by FairClout, which can differ from the previous candidate outliers.

5.2 Theoretical guarantees

Throughout this section, we assume we have access to an α-approximation algorithm A for the k-clustering problem without outliers. We state our main results, with proofs deferred to the appendix.

Theorem 5.1. *Let C be the output of FairClout $(X, k, \boldsymbol{z}, A)$ with $\beta = 3k + 2$. Then C gives an $((1 + (1 + 4(1 + \epsilon)^{1/p})^p) \cdot \alpha,\ 3k + 2)$−approximate solution.*

Theorem 5.2. *Let C be the output of FairClout $(X, k, \boldsymbol{z}, A)$ with $\beta = 2$. If there is an optimal solution where every optimal cluster has $\geq 3z_a$ points with color a for each $a \in \mathcal{A}$, then C gives an $((1 + (1 + 4(1 + \epsilon)^{1/p})^p) \cdot \alpha,\ 2)$−approximate solution.*

Notice that, by using any constant approximation algorithm for the specific k-clustering problem (e.g. [43] for k-means), our algorithm achieves a constant approximation of the optimal solution. As for the outliers, in general, we will discard $O(k)$ times the number of requested outliers for each color. However, if every cluster of an optimal solution has a good amount of points of each color, we will just discard twice the amount of desired outliers.

Running Time. Under the standard assumption in many clustering studies [49] that the ratio between the maximum and the minimum non-zero distance between any two input points is in $poly(n)$, we can prove that a naive implementation of our algorithm would have a total running time $O(c \log(n) \cdot n^2/\epsilon) + T_A(n)$, where T_A is the running time of the chosen k−clustering algorithm A on an instance of size n. More details can be found in the appendix. Notice that it is not possible to perform the ball construction in $o(n^2)$ for general metrics, as one can notice from the metric in which all the points have similar pairwise distances, so there is no structure to take advantage of. However, several methods can be used to perform the ball construction procedure faster for specific metrics (i.e., in $o(n^2)$), e.g., using well-known methods for nearest neighbor search [14, 48, 65]. This is what we do in our experiments by using the method from [17] for euclidean spaces.

6 A Fair Coreset Construction

The pre-processing procedure is the main factor contributing to the complexity of our algorithm: on each set X_a with size n_a, a naive implementation requires n_a^2 pairwise comparisons, making the naive procedure inapplicable to large datasets. This complexity can often be reduced with efficient nearest-neighbor search techniques. Here we introduce a coreset construction for our problem so that we can deal even with large datasets in arbitrary Euclidean spaces. A coreset is intended to be a compact representation of the initial input, on which computing a clustering solution would be much faster. If the computed solution on the coreset is a good approximation of the optimal solution, by definition of coreset, it would also be a good solution for the initial input. Here we formulate the definition and the algorithm only for the problem of k−means with fair outliers ($p = 2$) in a Euclidean space, but the same technique can be extended to any k−clustering problem in a Euclidean space with $p \geq 1$ factor.

We define bi-criteria approximation for coresets for this problem.

Definition 6.1. (coreset) Given $\alpha \geq 1, \beta \geq 1$ and an instance $X \subseteq \mathbb{R}^d, k, \boldsymbol{z}$ of k-means with fair outliers, let $S \subseteq \mathbb{R}^d$ with non-negative weights and $\boldsymbol{z'} = (z'_a)_{a \in \mathcal{A}}$. We say that $(S, k, \boldsymbol{z'})$ is an (α, β)−coreset of $(X, k, \boldsymbol{z})$ if, for any set of k centers $H \subseteq \mathbb{R}^d$, we have that $f^S_{\gamma_1 \boldsymbol{z'}}(H) \leq \gamma_2 \cdot OPT(S, k, \boldsymbol{z'})$ implies $f^X_{\beta \cdot \gamma_1 \boldsymbol{z}}(H) \leq \alpha \cdot \gamma_2 \cdot OPT(X, k, \boldsymbol{z})$ for any $\gamma_1, \gamma_2 > 0$.

Algorithm 2 FairCoreset$(X, \boldsymbol{z}, k)$
Input: instance $X \subseteq \mathbb{R}^d, k, \boldsymbol{z}$ of k-means with fair outliers.

for $a \in \mathcal{A}$ **do**
  for $i = 1 : 33\lceil \log(c \cdot n) \rceil$ **do**
    $p_a \leftarrow \max\left(\frac{36}{z_a} \cdot \log\left(\frac{4n_a k^2}{z_a}\right), 36\frac{k}{z_a} \cdot \log(2k^3)\right)$
    if $p_a > 1$ **then**
      $Y_a^i \leftarrow$ KMeans++$(X_a, 32(k + z_a))$
    else
      Let S_a be a random sample from X_a where each element from X_a is included independently with probability p_a
      $Y_a^i \leftarrow$ KMeans++$(S_a, 32(k + 2.5 p_a z_a))$
  $Y_a \leftarrow \arg\min_{\{Y_i^a \mid 1 \leq i \leq \lceil \log(c) \rceil\}} f^{X_a}_{16 z_a}(Y_a^i)$
return $Y := \bigcup_{a \in \mathcal{A}} Y_a$

In other words, with an (α, β)-coreset construction, we could produce a solution with any algorithm for k−means with fair outliers on the coreset, for which we use the input outlier constraints. If the computed solution is a γ−approximation of the optimal solution on the coreset, then it is an $\alpha \cdot \gamma$−approximation of the optimal solution on the initial input with β times the number of outliers for each color. For the coreset to be practical, it needs to be much smaller than the initial input set. We will show an $(O(1), O(1))$−coreset construction for the problem of k−means with fair outliers. As in FairClout, the construction is independent across the sets of points with different colors. The coreset will have expected size $O(ck \log(n))$, which is also independent of the number of outliers.

6.1 The Coreset Construction

Algorithm 2, which we call FairCoreset, builds upon the SampleCoreset algorithm from Im et al. [38]. It repeats the SampleCoreset algorithm $O(\log(n))$ times on each monochromatic subset of X to increase the probability of each monochromatic set to be a good approximation of the initial set. The resulting coreset is obtained by merging all the monochromatic coresets. For each color, we rely on two classic coreset constructions: KMeans++ from [2] and the sampling procedure by Meyerson et al. [53]. KMeans++ procedure computes a solution for the k−means problem on a given set with the well known $k$means++ algorithm [6]. The resulting coreset contains the centers provided by the algorithm, each with a weight equal to the number of points in the input set for which it is their closest center. We use two different coreset constructions to ensure that the size of the coreset does not grow linearly with the number of outliers. For a fixed color a, when $z_a = O(\log(n))$, we could just use the KMeans++ coreset construction. However, when z_a is much larger than that, Meyerson's sampling procedure still allows us to get a small coreset anyway. Notice also that we are referring to the natural logarithm.

6.2 Theoretical guarantees

We state our main result, with proofs deferred to the appendix. Notice that our coreset construction gives a near-linear time algorithm for k−means with fair outliers.

Theorem 6.2. *With probability $\geq 1-1/n$, Y is a $(124, 16)-$coreset for the $k-$means with fair outliers instance $(X, k, \boldsymbol{z})$ of size $O(c \cdot k\log(n))$ in expectation.*

Theorem 6.3. *Consider an instance $X \subseteq \mathbb{R}^d, k, \boldsymbol{z}$ of k-means with fair outliers and consider an $\alpha-$approximation algorithm A for $k-$means with running time $T_A(n)$. By applying FairCoreset, followed by FairClout with algorithm A and $\beta = 3k+2$, we get an $(O(1), O(k))-$approximation algorithm for $k-$means with fair outliers with expected running time $\tilde{O}(kdn + c^3k^2/\epsilon) + T_A(ck\log(n))$.*

7 Balanced Clusters with Fair Outliers

In this paper, we focus on the important issue of fairness in outlier removal. This notion of fairness is indeed only one of the possible requirements that might be desirable in certain clustering applications. Therefore, a natural question is whether fair outlier removal can be obtained in conjunction with other fairness constraints on the clustering process. As reviewed in Section 2, by and large, prior art has focused on imposing fairness constraints to the *clusters* in output, *e.g.* the balancedness constraint of [25], which is a natural way to ward off disparate impact. This notion requires that, in all clusters, the proportion of points of different categories be roughly the same as their representation in the entire population.

As a first step into the study of combining multiple fairness guarantees, we consider three natural problems: *k-means with fair outliers* (as in our paper); *balanced $k-$means with outliers*, in which we require clusters balancedness after removing a fixed number of outliers (without constraints); and, *balanced $k-$means with fair outliers*, in which we require clusters balancedness after removing a set of outliers *fairly* (as per our definition). In short, the first problem asks for fair outlier removal, the second for balancedness of the clusters, and the third for the two together.

The following result shows that there are inputs such that, while the first two have zero cost, in the third case, the cost is unbounded (the proof can be found in the appendix).

Theorem 7.1. *There exist infinitely many inputs for $2-$means on the euclidean line, where each input point is either blue or red, such that the optimal costs are: (i) zero, for $k-$means with fair outliers; (ii) zero, for balanced $k-$means with outliers; and (iii) unbounded, for balanced $k-$means with fair outliers.*

This suggests that these two natural notions of fairness, when combined, may impose arbitrarily high costs (at least in some instances) and that novel algorithmic techniques may be needed to study such a problem. We leave this research as future work.

8 Empirical Evaluation

In this section, we study our algorithm FairClout experimentally, and compare its performance with several baselines. First, we show empirically that standard clustering algorithms produce unfair outliers. Then, we show that our algorithm can ensure fairness in outlier removal with a negligible increase in the cost of the solution, where the cost is taken by the objective function. Finally, we assess the effectiveness of our coreset construction.

8.1 Experimental setup

Our experiments are all run using publicly available datasets from UCI [29] and DBLP. The algorithms are all implemented in C++ and run on commodity hardware. We provide the source code on `github`.[2] In all our experiments, the input points are in the Euclidean space, and we use the l_2 distance function. Moreover, we focus on the k-means problem ($p = 2$) and we use, as the A solver, k-means++ followed by Lloyd's algorithm [6].

Datasets. We used the following standard datasets commonly adopted in the literature on fairness in clustering. The datasets are described in more detail in the appendix and have up to ~ 5 million points, 33 dimensions, and 5 colors. *4area* is a co-authorship graph of researchers in four areas of computer science, embedded as in [3] in a Euclidean space. The color of each point identifies the researcher's area. *adult* [29] contains census data. For this dataset and all the ones below, from the available features, we consider only the numeric attributes, each normalized independently. As colors, we consider either the race or sex attribute. *bank* [55] contains data points from a bank's marketing campaign, and the colors are marital status and education level. *diabetes* [63] contains admission records of diabetic patients. As colors, we consider either race or gender. *kdd* [62] contains connection logs generated for an intrusion detection task. As colors, we consider the connection protocols. Note that these datasets are fully unsupervised – without ground-truth clusters, which are needed to measure the ability of the algorithms to recover clusters correctly. Thus, we also use synthetic instances.

Synthetic Datasets are generated from a mixture of $k \in [5, 10, 20]$ Gaussian distributions (all having the same variance) to obtain the ground-truth clusters. We generate instances having between 1000 and 10000 points. One of the clusters is chosen as a minority cluster with all points of the same color; the other $k-1$ majority clusters contain points from all the other colors. Finally, some true outliers are added uniformly, spanning all the colors. The instance points are distributed as 96% belonging to majority clusters, 2% to the minority cluster, and the remaining as outliers.

Baselines. We recall that no prior work has addressed the problem of k-means clustering with fair outliers (the algorithms [5, 10] on fair outliers in clustering were on k-center clustering). For this reason, we use as baselines several algorithms for k-means clustering with (non-fair) outliers and standard k-means (without outliers). We use two baseline algorithms that are oblivious to outliers: random (which samples k u.a.r. centers and assigns all points to the nearest center) and k-means++ [6]. We also use the following k-means with outlier algorithms: k-means-- [22], T-kmeans++ [16] and NK-means [38] (using k-means++ as post-pruning algorithm). As some baselines do not take into account the outliers, while other algorithms may discard arbitrarily many points, for a fair comparison all algorithms are forced to discard exactly $\sum_a z_a$ outliers, considering as outliers the farthest points from the chosen centers (this is the same methodology as in [38]). Analogously, although our algorithm is bi-criteria, we force FairClout as well to output exactly z_a outliers for each color a, so that all algorithms are allowed the same number of outliers. For both our algorithm and NK-means we use $\epsilon = 0.1$ for the grid search to guess the value of OPT. To further ensure a fair comparison with the algorithms that do not perform multiple guesses of OPT, we allow all other algorithms to perform the same number of independent runs and take the best solution as the answer. All the algorithms are executed

[2] https://github.com/matteojug/kclustering-with-fair-outliers

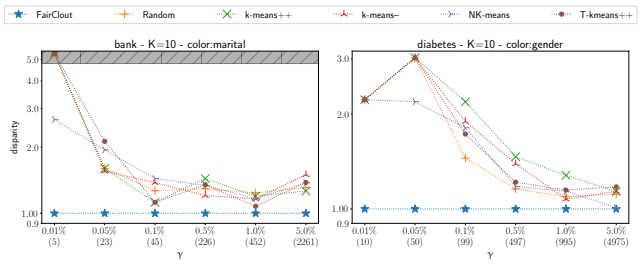

Figure 1: Disparity ratio between the most and least discarded color.

100 times (with runs distributed among the different OPT guesses if the algorithm required such guesses).

Implementation details of FairClout. We made minor changes to FairClout to make it faster while not affecting the theoretical guarantees. The most expensive part of our algorithm is the ball construction needed to find the heavy points. Being able to treat each color separately reduces the running time, but we also used a standard neighbor search technique [35], which allows orders of magnitude speedups over the naive implementation without any accuracy loss. Our algorithm requires a parameter β for the bi-criteria factor. We showed that, depending on the data, β must be between 2 and up to $3k+2$ to obtain worst–case theoretical guarantees. In our implementation, as a heuristic, we always try $\beta = 1, 2$ and $3k+2$ and report the best result found. Notice that this does not affect the theoretical guarantees.

8.2 Experimental results

First, we show that the non-fair baseline algorithms often discard outliers in an unfair way. As a benchmark setting for fair outlier removal, in all our experiments, we use the demographic parity case, i.e., allocating outliers to color classes proportional to the number of input points of each color. Let $0 < \gamma < 1$ be the *fraction* of input points we want to discard. For any color a, we set z_a as the expected number of outliers if they were proportionally discarded, i.e., $z_a := \lceil \gamma |X_a| \rceil$. To evaluate the fairness of all algorithms, we compute $\overline{z}_a$, which is the actual number of outliers of color a discarded by an algorithm. Let $d_a = z_a/\overline{z}_a$ be the ratio of the actual outliers of color a with respect to the desired number of outliers (which in this case guarantees demographic parity). As a measure of unfairness, we define as *disparity* the quantity $\max_a d_a / \min_a d_a$, i.e., the ratio of the most discarded and least discarded color (in relation to their desired number of outliers).

Disparate impact. In Fig. 1, we report the disparity measured for all algorithms on the different datasets for $k = 10$; similar results are obtained with different choices of k. In each plot, the outliers percentage γ varies on the horizontal axis, with the total number of outliers reported in parenthesis. On the vertical axis, we report the disparity as defined above. Note that FairClout has by construction disparity 1 (i.e., ideal proportional allocation). Notice also that a baseline algorithm could output no outliers of a given color, despite outputting many outliers of other colors, in which case the disparity is actually infinite: such values are plotted in the shaded area on the top of each plot. From the plots, it is possible to see that all the baseline algorithms show a high disparity across different datasets and colors. While this is not necessarily surprising, since those algorithms are oblivious to the classes of the points, this empirically establishes the disproportionate impact of outlier detection on different color classes. As expected, our FairClout algorithm instead always achieves perfect demographic parity.

Table 1: Cost increase for a fair solution from FairClout vs the best unfair baselines (k=10).

Dataset	Color	Outlier fraction γ			
		0.05%	0.1%	0.5%	1.0%
4area	area	+0.0%	+0.1%	+0.3%	+0.4%
adult	race	+0.1%	+0.0%	+0.1%	+0.0%
bank	education	+0.6%	+0.7%	+0.8%	+1.0%
diabetes	gender	+0.3%	+0.2%	+0.1%	+0.0%

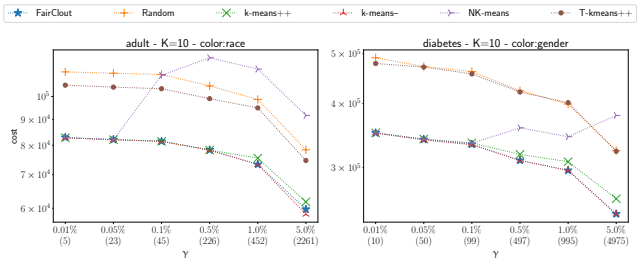

Figure 2: Cost of clustering: unfair baselines vs FairClout.

Cost of fairness. We focus on the cost of fairness in outlier removal, showing that achieving fairness is not very expensive. In Fig. 2, we report the cost of the clustering found by each unfair baseline. We also report the cost of the clustering found by FairClout, which guarantees a fair outlier output. Each of those algorithms is executed on the instance specified in the title of the plot and the specific value of γ reported on the horizontal axis. We report a representative set of the results for space reasons, but a similar picture arises from all datasets. We can see that the extra cost incurred by FairClout to discard the outliers in a fair way is negligible: our results are competitive with the best performing (non-fair) baseline while guaranteeing fairness in the outliers. To further illustrate this, we report in Table 1 the cost increase, in percentage point, between the lowest cost non-fair baseline and FairClout. It is possible to see that the fair solution is always very close to the best unfair solution and the cost increases at most by $\sim 1\%$.

Forcing fairness on unfair algorithms. Notice that, given any algorithm for k-clustering – even without outliers – one can force a fair variant of the algorithm by discarding the furthest z_a points of color a from the computed centers. This heuristic, despite being fair, of course, does not provide any theoretical guarantees for the quality of the solution, contrary to our algorithm. Here, as an additional comparison, we are interested in finding out how our algorithm fares against all non-fair baseline algorithms after transforming them into fair algorithms with this trick. We report some representative results in Figure 3. We observe that our algorithm always has the best cost of all baselines (turned fair), or it is within a fraction of percent of the best cost. Interestingly, despite the other baselines with forced fairness having no theoretical guarantees – contrary to FairClout– we see that some of them can sometimes have reasonably good performances on real datasets, further confirming that in some cases the cost of fairness in outliers is limited.

Synthetic instances with ground-truth clusters. To measure the ability of our algorithm to recover the original clusters, we compare

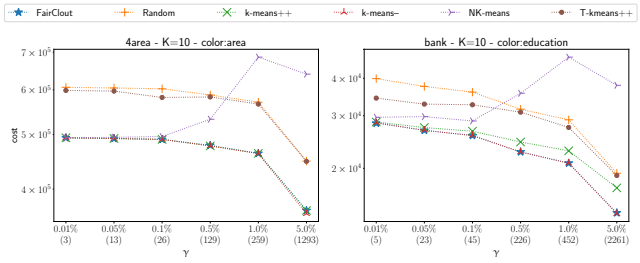

Figure 3: Cost of the clustering: baselines with forced fairness vs FairClout.

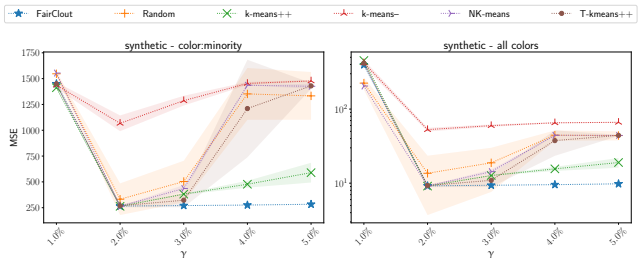

Figure 4: MSE of the regression model trained on the inliers.

the algorithms in terms of the Adjusted Rand Index on the synthetic instances described before. In all the experiments FairClout outperformed the baselines. The results on instances having k=10 clusters and γ=1% true outliers are reported in Table 2.

FairClout on the coreset. So far, we have evaluated our algorithm without using the coreset. In Table 3 we show the impact of using the coreset on the speed and cost of the solution computed by our algorithm. We compared FairClout running on the coreset and directly on the input instance. For each instance, we report the ratio between the coreset and the standard version of the cost ($\Delta Cost$), the total running time ($\Delta Time$)–coreset construction plus algorithm; and the running time of the algorithm only ($\Delta Time$ FairClout). We observe significant speedups: notice how the total running time – including the time spent on building the coreset – shrinks to less than 6% of the original time. As expected, the speedup of the algorithm after the coreset construction is even larger (up to 3 orders of magnitude). This speedup comes with a limited effect on the cost, which, in all datasets, is significantly better than the theoretical worst-case bounds (in the 10%-70% increase range). The coreset allows running our algorithm on the largest instance, kdd, while the version without coreset did not complete in 40 hours. For this reason, we show a lower-bound of the speedup. This experiment shows that the coreset enables scaling up the computation significantly.

Effects of fairness on downstream tasks. We experimented with a simple learning task on top of the synthetic instances described before to show the impact that an unfair selection of the outliers can induce. We associate to each cluster a coefficient vector sampled from a multivariate Gaussian distribution; then, we assign each point a label given by the dot product between the point and the coefficient of the cluster the point belongs to, plus some noise. For the outliers, we use a different set of coefficients. The learning task is a regression task where we want to predict a point label given its coordinates after removing the outliers from the instance. We tested the impact of the outliers selection for this task in the following way. First, an algorithm for clustering with outliers is run on the training instance, and a regression model is trained on the inliers returned by the algorithm. Then, the model is tested on a test instance generated with the same generative model of the training instance without outliers. The model used is the same for all the clustering algorithms. The results are shown in Figure 4. When the outliers fraction γ grows, the unfair algorithms discard an increasing number of points belonging to the minority color, inducing a higher error in the model's performance – both on minority points and in the overall instance. On the other hand, our fair algorithm keeps the same error for growing values of γ, achieving a lower error thanks to fairness. This shows that unfairly removing outliers could be disruptive, even for very simple downstream learning tasks, raising concern for many real-world applications.

Table 2: Adjusted Rand Index on synthetic instances with (k=10) ground-truth clusters and (γ=1%) ground-truth outliers. We included runs with other γ values for comparison.

Algorithm	1.0%	1.5%	2.0%	3.0%
Random	0.68	0.68	0.68	0.69
k-means++	0.22	0.22	0.22	0.22
k-means–	0.22	0.22	0.22	0.22
T-kmeans++	0.74	0.77	0.76	0.77
FairClout	**0.82**	**0.85**	**0.90**	**0.92**

Table 3: Comparison between FairClout with and without coreset (k=10, γ=1%). $\Delta Cost$ is the ratio between costs, $Time$ and $Time$ FairClout the running time (and speedup) of respectively the end to end algorithm (including coreset construction) and FairClout time only.

Dataset	Color	$\Delta Cost$	$Time$	$Time$ FairClout
4area	area	1.089	11.4s (0.011x)	1.0s (0.001x)
adult	race	1.233	74.8s (0.038x)	1.7s (0.0009x)
	sex	1.286	64.3s (0.043x)	1.7s (0.0011x)
bank	education	1.682	24.6s (0.046x)	0.9s (0.0016x)
	marital	1.708	36.5s (0.059x)	0.9s (0.0015x)
diabetes	gender	1.414	159s (0.053x)	3.7s (0.0013x)
kdd	target	N/A	25m (<0.011x)	11.7s (<0.0001x)

9 Conclusions

We studied the problem of clustering with fairness constraints in outlier removal. We proved the need for this type of fairness by showing the disproportionate impact of standard methods for clustering with outliers. To address the issue, we have developed FairClout, a novel algorithm with provable approximation guarantees and strong empirical performance. We believe that our results address an important and neglected issue. We also provided a coreset construction, expanding the applicability of fair outlier removal algorithms to large datasets. For future work, an interesting direction is to study the problem of balanced clustering with fair outliers.

Acknowledgements

This work was supported in part by a Google Focused Research Award, by the PRIN project 2017K7XPAN, by BiCi — Bertinoro international Center for informatics, and by the MIUR under grant "Dipartimenti di eccellenza 2018-2022" of the Department of Computer Science of Sapienza University.

References

[1] Mohsen Abbasi, Aditya Bhaskara, and Suresh Venkatasubramanian. 2020. Fair clustering via equitable group representations. *arXiv:2006.11009* (2020).

[2] Ankit Aggarwal, Amit Deshpande, and Ravi Kannan. 2009. Adaptive sampling for k-means clustering. In *APPROX*. Springer, 15–28.

[3] Sara Ahmadian, Alessandro Epasto, Ravi Kumar, and Mohammad Mahdian. 2019. Clustering without over-representation. In *KDD*. 267–275.

[4] Sara Ahmadian, Alessandro Epasto, Ravi Kumar, and Mohammad Mahdian. 2020. Fair Correlation clustering. In *AISTATS*.

[5] Georg Anegg, Haris Angelidakis, Adam Kurpisz, and Rico Zenklusen. 2020. A Technique for Obtaining True Approximations for k-Center with Covering Constraints. In *IPCO*. Springer, 52–65.

[6] David Arthur and Sergei Vassilvitskii. 2006. *k-means++: The advantages of careful seeding*. Technical Report. Stanford.

[7] Arturs Backurs, Piotr Indyk, Krzysztof Onak, Baruch Schieber, Ali Vakilian, and Tal Wagner. 2019. Scalable fair clustering. *arXiv:1902.03519* (2019).

[8] Bahman Bahmani, Benjamin Moseley, Andrea Vattani, Ravi Kumar, and Sergei Vassilvitskii. 2012. Scalable k-means++. *arXiv:1203.6402* (2012).

[9] Tanvi Bajpai, Deeparnab Chakrabarty, Chandra Chekuri, and Maryam Negahbani. 2021. Revisiting Priority k-Center: Fairness and Outliers. *arXiv preprint arXiv:2103.03337* (2021).

[10] Sayan Bandyapadhyay, Tanmay Inamdar, Shreyas Pai, and Kasturi Varadarajan. 2019. A Constant Approximation for Colorful k-Center. In *ESA*.

[11] Solon Barocas, Moritz Hardt, and Arvind Narayanan. 2019. *Fairness and Machine Learning*. fairmlbook.org. http://www.fairmlbook.org.

[12] Suman Bera, Deeparnab Chakrabarty, Nicolas Flores, and Maryam Negahbani. 2019. Fair algorithms for clustering. In *NeurIPS*. 4954–4965.

[13] Suman K Bera, Shalmoli Gupta, Amit Kumar, and Sambuddha Roy. 2014. Approximation algorithms for the partition vertex cover problem. *Theor. Comput. Sci.* 555 (2014), 2–8.

[14] Alina Beygelzimer, Sham Kakade, and John Langford. 2006. Cover trees for nearest neighbor. In *ICML*. 97–104.

[15] Aditya Bhaskara and Aravinda Kanchana Rwanpathirana. 2020. Robust Algorithms for Online k-means Clustering. In *ALT*. 148–173.

[16] Aditya Bhaskara, Sharvaree Vadgama, and Hong Xu. 2019. Greedy Sampling for Approximate Clustering in the Presence of Outliers. In *NeurIPS*. 11146–11155.

[17] Nitin Bhatia et al. 2010. Survey of nearest neighbor techniques. *arXiv:1007.0085* (2010).

[18] Dan Biddle. 2006. *Adverse impact and test validation: A practitioner's guide to valid and defensible employment testing*. Gower Pub. Ltd.

[19] Johannes Blömer, Christiane Lammersen, Melanie Schmidt, and Christian Sohler. 2016. *Theoretical Analysis of the k-Means Algorithm – A Survey*. Springer, 81–116.

[20] TH Hubert Chan, Arnaud Guerqin, and Mauro Sozio. 2018. Fully dynamic k-center clustering. In *WWW*. 579–587.

[21] Moses Charikar, Samir Khuller, David M Mount, and Giri Narasimhan. 2001. Algorithms for facility location problems with outliers. In *SODA*. SIAM, 642–651.

[22] Sanjay Chawla and Aristides Gionis. 2013. k-means−: A unified approach to clustering and outlier detection. In *SDM*. SIAM, 189–197.

[23] Jiecao Chen, Erfan S Azer, and Qin Zhang. 2018. A practical algorithm for distributed clustering and outlier detection. In *NeurIPS*. 2248–2256.

[24] Ke Chen. 2008. A constant factor approximation algorithm for k-median clustering with outliers. In *SODA*. 826–835.

[25] Flavio Chierichetti, Ravi Kumar, Silvio Lattanzi, and Sergei Vassilvitskii. 2017. Fair clustering through fairlets. In *NIPS*. 5029–5037.

[26] Sam Corbett-Davies and Sharad Goel. 2018. The measure and mismeasure of fairness: A critical review of fair machine learning. *arXiv:1808.00023* (2018).

[27] Amit Deshpande, Praneeth Kacham, and Rameshwar Pratap. 2020. Robust k-means++. In *UAI*. PMLR, 799–808.

[28] Hu Ding, Haikuo Yu, and Zixiu Wang. 2019. Greedy Strategy Works for k-Center Clustering with Outliers and Coreset Construction. In *ESA*.

[29] Dheeru Dua and Casey Graff. 2017. UCI Machine Learning Repository. http://archive.ics.uci.edu/ml

[30] Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. 2012. Fairness through awareness. In *ITCS*. 214–226.

[31] Michael Feldman, Sorelle A Friedler, John Moeller, Carlos Scheidegger, and Suresh Venkatasubramanian. 2015. Certifying and removing disparate impact. In *KDD*.

[32] Zachary Friggstad, Kamyar Khodamoradi, Mohsen Rezapour, and Mohammad R Salavatipour. 2019. Approximation schemes for clustering with outliers. *TALG* (2019).

[33] Mehrdad Ghadiri, Samira Samadi, and Santosh Vempala. 2021. Socially fair k-means clustering. In *FAccT*. 438–448.

[34] Shalmoli Gupta, Ravi Kumar, Kefu Lu, Benjamin Moseley, and Sergei Vassilvitskii. 2017. Local search methods for k-means with outliers. *VLDB* 10, 7 (2017), 757–768.

[35] Sariel Har-Peled. 2011. *Geometric approximation algorithms*. Number 173. AMS.

[36] Moritz Hardt, Eric Price, and Nati Srebro. 2016. Equality of Opportunity in Supervised Learning. *NIPS* 29 (2016), 3315–3323.

[37] David G Harris, Thomas Pensyl, Aravind Srinivasan, and Khoa Trinh. 2019. A lottery model for center-type problems with outliers. *TALG* 15, 3 (2019), 1–25.

[38] Sungjin Im, Mahshid M Qaem, Benjamin Moseley, Xiaorui Sun, and Rudy Zhou. 2020. Fast Noise Removal for k-Means Clustering *(PMLR, Vol. 108)*. 456–466.

[39] Russell Impagliazzo and Ramamohan Paturi. 2001. On the complexity of k-SAT. *J. Comput. System Sci.* 62, 2 (2001), 367–375.

[40] Tanmay Inamdar and Kasturi Varadarajan. 2018. On the Partition Set Cover Problem. *arXiv:1809.06506* (2018).

[41] Anil K Jain. 2010. Data clustering: 50 years beyond K-means. *Pattern recognition letters* 31, 8 (2010), 651–666.

[42] Xinrui Jia, Kshiteej Sheth, and Ola Svensson. 2020. Fair Colorful k-Center Clustering. In *IPCO*. Springer, 209–222.

[43] Tapas Kanungo, David M Mount, Nathan S Netanyahu, Christine D Piatko, Ruth Silverman, and Angela Y Wu. 2002. A local search approximation algorithm for k-means clustering. In *SOCG*. 10–18.

[44] Michael Kearns and Aaron Roth. 2019. *The ethical algorithm: The science of socially aware algorithm design*. Oxford University Press.

[45] Aria Khademi, Sanghack Lee, David Foley, and Vasant Honavar. 2019. Fairness in algorithmic decision making: An excursion through the lens of causality. In *WWW*. 2907–2914.

[46] Matthäus Kleindessner, Pranjal Awasthi, and Jamie Morgenstern. 2019. Fair k-center clustering for data summarization. *arXiv:1901.08628* (2019).

[47] Emmanouil Krasanakis, Eleftherios Spyromitros-Xioufis, Symeon Papadopoulos, and Yiannis Kompatsiaris. 2018. Adaptive sensitive reweighting to mitigate bias in fairness-aware classification. In *WWW*. 853–862.

[48] Robert Krauthgamer and James R Lee. 2004. Navigating nets: Simple algorithms for proximity search. In *SODA*. Citeseer, 798–807.

[49] Silvio Lattanzi and Sergei Vassilvitskii. 2017. Consistent k-clustering. In *ICML*.

[50] Shi Li and Ola Svensson. 2016. Approximating k-median via pseudo-approximation. *SIAM J. Comput.* 45, 2 (2016), 530–547.

[51] Daniel Lokshtanov, Chinmay Sonar, Subhash Suri, and Jie Xue. 2020. Fair Covering of Points by Balls. (2020).

[52] Sepideh Mahabadi and Ali Vakilian. 2020. Individual fairness for k-clustering. In *ICML*. PMLR, 6586–6596.

[53] Adam Meyerson, Liadan O'callaghan, and Serge Plotkin. 2004. A k-median algorithm with running time independent of data size. *Machine Learning* (2004).

[54] Burt L Monroe. 1995. Fully proportional representation. *American Political Science Review* (1995), 925–940.

[55] Sérgio Moro, Paulo Cortez, and Paulo Rita. 2014. A data-driven approach to predict the success of bank telemarketing. *Decis. Support Syst.* 62 (2014), 22 – 31.

[56] Bruno Ordozgoiti and Aristides Gionis. 2019. Reconciliation k-median: Clustering with Non-polarized Representatives. In *WWW*. 1387–1397.

[57] Geoff Pleiss, Manish Raghavan, Felix Wu, Jon Kleinberg, and Kilian Q Weinberger. 2017. On Fairness and Calibration. *NeurIPS* 30 (2017), 5680–5689.

[58] Jean Pouget-Abadie, Vahab Mirrokni, David C Parkes, and Edoardo M Airoldi. 2018. Optimizing cluster-based randomized experiments under monotonicity. In *KDD*. 2090–2099.

[59] Dongmei Ren, Imad Rahal, William Perrizo, and Kirk Scott. 2004. A vertical distance-based outlier detection method with local pruning. In *CIKM*. 279–284.

[60] Melanie Schmidt, Chris Schwiegelshohn, and Christian Sohler. 2018. Fair coresets and streaming algorithms for fair k-means clustering. *arXiv:1812.10854* (2018).

[61] Anurag Shandilya, Kripabandhu Ghosh, and Saptarshi Ghosh. 2018. Fairness of Extractive Text Summarization. In *WWW*. 97–98.

[62] J Stolfo, Wei Fan, Wenke Lee, Andreas Prodromidis, and Philip K Chan. 2000. Cost-based modeling and evaluation for data mining with application to fraud and intrusion detection. *Results from the JAM Project by Salvatore* (2000), 1–15.

[63] Beata Strack, Jonathan Deshazo, Chris Gennings, Juan Luis Olmo Ortiz, Sebastian Ventura, Krzysztof Cios, and John Clore. 2014. Impact of HbA1c Measurement on Hospital Readmission Rates: Analysis of 70,000 Clinical Database Patient Records. *BioMed research international* 2014 (04 2014), 781670.

[64] David P Williamson and David B Shmoys. 2011. *The design of approximation algorithms*. Cambridge University Press.

[65] Peter N Yianilos. 1993. Data structures and algorithms for nearest neighbor search in general metric spaces. In *SODA*, Vol. 93. 311–21.

[66] Hongjing Zhang and Ian Davidson. 2021. Towards Fair Deep Anomaly Detection. In *FAccT*. 138–148.

[67] Ying Zhao and George Karypis. 2005. Data clustering in life sciences. *Mol. biotechnol.* 31, 1 (2005), 55–80.

A FairClout: Analysis

We establish a series of technical lemmas to prove our main result.

Lemma A.1. *For an optimal center $c^* \in C^*$, let C be its cluster and let $x_a(c^*)$ be the closest point to c^* in $C \cap X_a$. If $|C \cap X_a| \geq 3z_a$, then $d(x_a(c^*), c^*) \leq (OPT_a/(3z_a))^{1/p}$. Moreover, if $\theta_a \geq OPT_a$ and $d(x, c^*) \leq (\theta_a/(3z_a))^{1/p}$ for a point $x \in C \cap X_a$, then x is heavy.*

Proof. First, assume by contradiction this is false, then each point with color a in C has distance $> (OPT_a/(3z_a))^{1/p}$ from the center. Since they are $\geq 3z_a$, their total cost in the optimal solution is $> OPT_a$, that is a contradiction. For the second claim, assume by contradiction it is light, so $|B_a(x, r_a)| < 2z_a$. However, at least $3z_a$ points from X_a are in this cluster, so $|(C \cap X_a) \setminus B_a(x, r_a)| \geq z_a + 1$. Let $x' \in (C \cap X_a) \setminus B_a(x, r_a)$. By triangle inequality, we have $d(x, x') \leq d(x', c^*) + d(x, c^*)$, so $d(x', c^*) \geq r_a - d(x, c^*) \geq 2(\theta_a/z_a)^{1/p} - 1/3^{1/p} \cdot (\theta_a/z_a)^{1/p} > (\theta_a/z_a)^{1/p} \geq (OPT_a/z_a)^{1/p}$. Therefore, the total cost of these $\geq z_a + 1$ points from X_a in the optimal solution is $> OPT_a$, that is a contradiction. □

Lemma A.2. *If $\theta_a \geq OPT_a$, then $|Y \cap X_a| \leq 2z_a + (3z_a - 1) \cdot |\{$clusters with $< 3z_a$ points with color $a\}|$.*

Proof. Fix a color a. First, it is easy to observe that we can discard $\leq 3z_a - 1$ points with color a from clusters with $< 3z_a$ points from X_a. In the remaining clusters, we can divide the points in two categories based on their distance from C^*. Assume $d(x, C^*) \leq (\theta_a/z_a)^{1/p}$, and let $c^* \in C^*$ its closest center. From Lemma A.1, we know that $x_a(c^*)$ is heavy. Furthermore, $d(x, x_a(c^*)) \leq d(x, c^*) + d(c^*, x_a(c^*)) \leq (\theta_a/z_a)^{1/p} + 1/3^{1/p} \cdot (OPT_a/z_a)^{1/p} \leq 2(\theta_a/z_a)^{1/p}$. Since $x_a(c^*) \in B_a(x, r_a)$ and $x_a(c^*)$ is heavy, we have that x cannot belong to Y. Assume $d(x, C^*) > (\theta_a/z_a)^{1/p}$ instead. There can be $\leq 2z_a$ such points. Assume by contradiction this is false, so there are $\geq 2z_a + 1$ such points. Then $\geq z_a + 1$ of them are not outliers in the optimal solution, and their total cost in the fixed optimal solution is $> \theta_a \geq OPT_a$, that is a contradiction. □

This lemma proves the outlier bounds for Theorems 5.2 and 5.1. We now need to prove the results for approximation ratio.

Lemma A.3. *Let $x \in X_a, a \in \mathcal{A}$, be a heavy point, $\theta_a \leq (1 + \gamma)OPT_a$ for a constant $\gamma > 0$. Then there exists $c^* \in C^*$ s.t. $d(x, c^*) \leq (1 + 2(1 + \gamma)^{1/p})(OPT_a/z_a)^{1/p}$.*

Proof. Assume by contradiction $d(x, C^*) > (1 + 2(1 + \gamma)^{1/p}) \cdot (OPT_a/z_a)^{1/p}$. Since x is heavy, $|B_a(x, r_a)| \geq 2z_a$, so it contains $\geq z_a$ inliers with color a. Consider any such inlier x', by the triangle inequality we have $(1 + 2(1 + \gamma)^{1/p})(OPT_a/z_a)^{1/p} < d(x, C^*) \leq d(x, x') + d(x', C^*) \leq r_a + d(x', C^*) = 2(\theta_a/z_a)^{1/p} + d(x', C^*)$, so $d(x', C^*) > (1+2(1+\gamma)^{1/p})(OPT_a/z_a)^{1/p} - 2(\theta_a/z_a)^{1/p} \geq (1+2(1+\gamma)^{1/p})(OPT_a/z_a)^{1/p} - 2(1+\gamma)^{1/p} \cdot (OPT_a/z_a)^{1/p} = (OPT_a/z_a)^{1/p}$. This implies that the total cost of these $\geq z_a$ inliers in the optimal solution is $> OPT_a$, which is a contradiction. □

Theorem A.4. *Let C be the output of FairClout $(X, k, \boldsymbol{z}, A)$. If A is an $\alpha-$approximation for the $k-$clustering problem and for each color $a \in \mathcal{A}$: $\theta_a \leq (1 + \gamma)OPT_a$ for a constant $\gamma > 0$; the outlier removal procedure deletes $\boldsymbol{z'} := (z'_a)_{a \in \mathcal{A}}$ outliers, then $f^X_{\boldsymbol{z'}}(C) \leq \alpha \cdot (1 + 4(1 + \gamma)^{1/p})^p \cdot OPT(X, k, \boldsymbol{z})$.*

Proof. By construction, we have $f^X_{\boldsymbol{z'}}(C) \leq f^{X \setminus Y}(C)$. Furthermore, by construction of C, we have that $f^{X \setminus Y}(C) \leq \alpha \cdot OPT(X \setminus Y, k) \leq \alpha \cdot f^{X \setminus Y}(C^*)$. So it suffices to bound $f^{X \setminus Y}(C^*)$. We will now bound separately the points in $(X \setminus Y) \cap X_{\boldsymbol{z}}(C^*)$ from the ones outside this set. First, consider $x \in (X \setminus Y) \cap X_{\boldsymbol{z}}(C^*)$, i.e. inlier in both our solution and the optimal solution. The total cost of all these points with respect to C^* is $\leq OPT$. We can now restrict to $x \notin (X \setminus Y) \cap (X \setminus X_{\boldsymbol{z}}(C^*))$, i.e. inliers in our solution that are outliers in the optimal solution. Fix a color a and consider all such points with this color. They can be at most z_a. Each one of these $\leq z_a$ points has a heavy point at distance $\leq r_a = 2(\theta_a/z_a)^{1/p} \leq 2(1 + \gamma)^{1/p} \cdot (OPT_a/z_a)^{1/p}$. However, by Lemma A.3, each heavy point has distance from $C^* \leq (1 + 2(1 + \gamma)^{1/p}) \cdot (OPT_a/z_a)^{1/p}$, so by the triangle inequality each such inlier has distance from C^* $\leq (1+4(1+\gamma)^{1/p}) \cdot (OPT_a/z_a)^{1/p}$. Therefore, the total cost of these points is is $\leq (1 + 4(1 + \gamma)^{1/p})^p \cdot OPT_a$. Since this is true for each color, the total cost of these points is $\leq (1+4(1+\gamma)^{1/p})^p \cdot OPT$. □

Deriving the Main Theorems. Let $\bar{\theta}_a$ be the smallest value of θ_a in Algorithm FairClout yielding $\leq (3k+2) \cdot z_a$ outliers for a fixed color a, i.e. the value obtained with $\beta = 3k + 2$. There are three cases: (1) $\bar{\theta}_a > d^p_{a,min}$, (2) $\bar{\theta}_a = d^p_{a,min}$ and (3) $\bar{\theta}_a = 0$. We show now that $\bar{\theta}_a \leq (1+\epsilon) \cdot OPT_a$. In the first case, since $\bar{\theta}_a/(1+\epsilon)$ yields too many outliers, by Lemma A.2 it holds $\bar{\theta}_a/(1+\epsilon) < OPT_a$, so $\bar{\theta}_a < (1+\epsilon) \cdot OPT_a$. The inequality holds necessarily in the second case too, unless $OPT_a = 0$, because $d^p_{a,min}$ is the smallest positive value that the cost can take on X_a. However, if $OPT_a = 0$, we would have stopped at $\theta_a = 0$ by Lemma A.2. Finally, if $\bar{\theta}_a = 0$, the inequality holds trivially. Now, by Theorem A.4, $\bar{\theta}_a$ gives an $\alpha \cdot (1 + (1 + 4(1 + \epsilon)^{1/p})^p)-$approximation of the optimal solution. For small ϵ, we get for $k-$ median an $\alpha \cdot (6 + O(\epsilon))-$approximation of the optimal solution, while for $k-$means an $\alpha \cdot (26+O(\epsilon))-$approximation of the optimal solution. For a general $k-$clustering problem with $p \geq 1$, this is an $\alpha \cdot (1 + 5^p + O(\epsilon))-$approximation of the optimal solution. Note that various constant approximation algorithms are known for these problems [43, 50], so we can get $(O(1), O(k))-$approximation algorithms for $k-$clustering problems with fair outliers.From this construction, Theorem 5.1 follows directly. However, if we have an optimal solution such that any cluster contains $\geq 3z_a$ points with color a for each color $a \in \mathcal{A}$, by Lemma A.2 the discarded number of outliers with a good estimate of the optimal cost is at most twice the required amount for each color. So, if this is the case, by choosing $\beta = 2$ in FairClout we can get an $(O(1), O(1))-$approximation because Theorem A.4 and the derived results hold in the same exact way. Therefore, also Theorem 5.2 follows.

Running Time. We recall the assumption that the ratio between the maximum and the minimum non-zero distance between any two input points is in $poly(n)$, so for each color, we will need to make $O(\log(n)/\epsilon)$ guesses, each corresponding to a run of the pre-processing procedure for a color. Thus we will perform $O(c \log(n)/\epsilon)$ total runs of the pre-processing procedure since this phase runs independently over points with different colors. This is because any non-zero optimum cost is between the $p-$th power of the minimum distance of two points and the $p-$th power of the maximum distance between two points times n. Notice that a naive implementation of the algorithm computing the number of elements

in every ball requires time $O(n_a^2)$ for points with color a. So, a naive implementation of our algorithm would have running time $O(c\log(n)\cdot n^2/\epsilon) + T_A(n)$, where T_A is the running time of the chosen $k-$clustering algorithm A on an instance with size n.

B The Coreset: Analysis

The following Theorem follows from [38].

Theorem B.1. *With probability $\geq 1-\frac{1}{cn}$, for a fixed color $a \in \mathcal{A}$, (Y_a, k, z_a) is a $(124, 16)-$coreset for the $k-$means with outliers instance (X_a, k, z_a) of size $O(k\log(n))$ in expectation.*

Proof. Y_a is chosen among $\geq 33\log(cn)$ candidate coresets $\{Y_a^i\}_i$. By the results in [38], each (Y_a^i, k, z_a) is a $(124, 16)-$coreset for the $k-$means with outliers instance (X_a, k, z_a) with probability ≥ 0.03. Therefore, the probability that none of those $\{Y_a^i\}_i$ is such a coreset is $\leq 0.97^{33\log(c\cdot n)} \leq \frac{1}{cn}$. □

Corollary B.2. *With probability $\geq 1-1/n$, simultaneously for every color $a \in \mathcal{A}$, (Y_a, k, z_a) is a $(124, 16)-$coreset for the $k-$means with outliers instance (X_a, k, z_a) of size $O(k\log(n))$ in expectation.*

Proof. By Theorem B.1 and by the independence of the coreset construction across the different colors, this is true with probability $\geq \left(1-\frac{1}{cn}\right)^c \geq 1-1/n$ by Bernoulli's inequality. □

Since $OPT(X_a, k, z_a) \leq OPT_a = OPT(X, k, \boldsymbol{z})|_{X_a}$ by construction, we can finally prove Theorem 6.2.

Proof of Theorem 6.2. By Corollary B.2, with probability $\geq 1-1/n$, simultaneously for every color $a \in \mathcal{A}$, (Y_a, k, z_a) is a $(124, 16)-$coreset for the $k-$means with outliers instance (X_a, k, z_a) of size $O(k\log(n))$ in expectation. We can condition on this and it will suffice to show that, conditioned on this event, the statement of the theorem holds. First, it follows easily that Y has size $O(c\cdot k\log(n))$ in expectation. Now, consider a set $C \subseteq \mathbb{R}^d$ of $k-$centers and assume there are $\gamma_1, \gamma_2 > 0$ such that $f^Y_{\gamma_1 \boldsymbol{z}}(C) \leq \gamma_2 \cdot OPT(Y, k, \boldsymbol{z})$. By hypothesis, from Corollary B.2 we have that $f^{Y_a}_{16\gamma_1 z_a}(C) \leq 124\gamma_2 \cdot OPT(Y_a, k, z_a)$ for each color $a \in \mathcal{A}$. However, by definition $OPT(Y_a, k, z_a) \leq OPT_a$, so $f^{Y_a}_{16\gamma_1 z_a}(C) \leq 124\gamma_2 \cdot OPT_a$. By summing over $a \in \mathcal{A}$, we get that $124\gamma_2 \cdot OPT \geq \sum_{a\in\mathcal{A}} f^{Y_a}_{16\gamma_1 z_a}(C) = f^Y_{16\gamma_1 \boldsymbol{z}}(C)$, as desired. □

By using our algorithm FairClout on the coreset returned by FairCoreset, we can get a bi-criteria approximation algorithm for $k-$means with fair outliers running in near linear time.

Proof of Theorem 6.3. As in Im et al. [38] (see Thm. 5), the running time needed to build a candidate coreset Y_a^i on a set X_a with size n_a is $O(kdn_a\log^2(n))$, so the running time needed to build Y_a is $O(kdn_a\log^3(n))$, and the overall running time for the construction of the coreset Y is $O(kdn\log^3(n))$. The final coreset, by Theorem 6.2, has expected size $O(c\cdot k\log(n))$. To apply FairClout on the coreset with a polynomial-time algorithm A for $k-$means, with a naive implementation of the phase counting the number of points in each ball, we need an extra running time $O\left(c^3k^2\log^2(n)\log(ck\log(n))/\epsilon\right) + T_A(ck\log(n))$. The total running time is therefore $\tilde{O}\left(kdn + c^3k^2/\epsilon\right) + T_A(ck\log(n))$. □

C Balanced Clusters with Fair Outliers

We consider the problem of 2−means on the euclidean line, with points belonging to two categories/colors: *red* and *blue*. We will have n points, $n/2$ red and $n/2$ blue. For simplicity of exposition, all logarithms in this section are in base 2, and we assume that n is a sufficiently large power of 2 and that $\log(n)/5$ is an integer. With respect to this scenario, we define the corresponding problems of fairness. We say that a clustering solution is *balanced* if the number of blue points in each cluster is within 1/4 and 3/4 the total number of points of the cluster itself. We want to exclude $2\log(n)$ outliers. The set of outliers is fair iff exactly $\log(n)$ of them are blue.

Instance: For $t >> 1$, place $\log(n)/5$ blue points and $2\log(n)$ red points in 0, $n/2 - 2\log(n)/5$ blue points and $n/2 - 2\log(n)$ red points in $t\cdot n$, $\log(n)/5$ blue points in $t^2\cdot n^2$. There are one large well-balanced cluster and two far away small unbalanced clusters.

For, *balanced 2−means with non-fair outliers* we show $OPT = 0$. Remove, as outliers, $9\log(n)/5$ red points in 0 and $\log(n)/5$ blue points in $t^2\cdot n^2$. Place the two centers of the clusters in 0 and $t\cdot n$. We have a *balanced* clustering having cost 0.

For *non-balanced 2−means with fair outliers* as well, $OPT = 0$. Remove, as outliers, $\log(n)$ red points in 0, $4\log(n)/5$ blue points in $t\cdot n$ and $\log(n)/5$ blue points in $t^2\cdot n^2$. Place the two centers of the clusters in 0 and $t\cdot n$. We have two clusters having cost 0.

Finally, we show that for *balanced 2−means with fair outliers*, $OPT \geq t^2\cdot n^2/4$ by showing that in each solution, one of the two clusters has diameter $\geq t\cdot n$. After the outlier removal, there will still be $\geq \log(n)$ red points in 0. By the cardinality fairness constraint, they should be in the same cluster with other $\geq \log(n)/4$ blue points. However, among the closest $\log(n)/4$ blue points to them, there need to be some placed in tn or t^2n^2. So each solution has a cluster with diameter $\geq t\cdot n$, having therefore 2−means cost $\geq t^2\cdot n^2/4$.

D Properties of the datasets used

Table 4: Datasets used in the experiments with their respective features, d number of dimensions, c number of colors.

Name	|X|	d	Color	c	$\min_a \lvert X_a\rvert$	$\max_a \lvert X_a\rvert$
4area	25,853	8	area	4	3,905	10,416
adult	45,222	6	race	5	353	38,903
			sex	2	14,695	30,527
bank	45,211	3	marital	3	5,207	27,214
diabetes	99,492	8	race	5	641	76,099
			gender	2	45,917	53,575
kdd	4,898,431	33	protocol	3	194,288	2,833,545

The datasets used are: 4area[3], adult[4], bank[5], diabetes[6], and kdd[7]. In Table 4 we report the properties of those datasets.

[3] https://dblp.uni-trier.de/xml
[4] https://archive.ics.uci.edu/ml/datasets/Adult
[5] https://archive.ics.uci.edu/ml/datasets/Bank+Marketing
[6] https://archive.ics.uci.edu/ml/datasets/diabetes+130-us+hospitals+for+years+1999-2008
[7] http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html