Fair Wasserstein Coresets

Zikai Xiong[*‡] **Niccolò Dalmasso**[†⋆] **Shubham Sharma**[†] **Freddy LeCue**[†]

Daniele Magazzeni[†] **Vamsi K. Potluru**[†] **Tucker Balch**[†] **Manuela Veloso**[†]

Abstract

Data distillation and coresets have emerged as popular approaches to generate a smaller representative set of samples for downstream learning tasks to handle large-scale datasets. At the same time, machine learning is being increasingly applied to decision-making processes at a societal level, making it imperative for modelers to address inherent biases towards subgroups present in the data. While current approaches focus on creating fair synthetic representative samples by optimizing local properties relative to the original samples, their impact on downstream learning processes has yet to be explored. In this work, we present fair Wasserstein coresets (`FWC`), a novel coreset approach which generates fair synthetic representative samples along with sample-level weights to be used in downstream learning tasks. `FWC` uses an efficient majority minimization algorithm to minimize the Wasserstein distance between the original dataset and the weighted synthetic samples while enforcing demographic parity. We show that an unconstrained version of `FWC` is equivalent to Lloyd's algorithm for k-medians and k-means clustering. Experiments conducted on both synthetic and real datasets show that `FWC`: (i) achieves a competitive fairness-utility tradeoff in downstream models compared to existing approaches, (ii) improves downstream fairness when added to the existing training data and (iii) can be used to reduce biases in predictions from large language models (GPT-3.5 and GPT-4).

1 Introduction

In the last decade, the rapid pace of technological advancement has provided the ability of collecting, storing and processing massive amounts of data from multiple sources [65]. As the volume of data continues to surge, it often surpasses both the available computational resources as well as the capacity of machine learning algorithms. In response to this limitation, dataset distillation approaches aim to reduce the amount of data by creating a smaller, yet representative, set of samples; see [80, 40] for comprehensive reviews on the topic. Among those approaches, coresets provide a weighted subset of the original data that achieve similar performance to the original dataset in (usually) a specific machine learning task, such as clustering [26, 21], Bayesian inference [11], online learning [9] and classification [16], among others.

In tandem with these developments, the adoption of machine learning techniques has seen a surge in multiple decision-making processes that affect society at large [69, 81]. This proliferation of machine learning applications has highlighted the need to mitigate inherent biases in the data, as these biases can significantly impact the equity of machine learning models and their decisions [14]. Among many definitions of algorithmic fairness, demographic parity is one of the most prominently used metric

[*]Operations Research Center, Massachusetts Institute of Technology, `zikai@mit.edu`

[†]J.P.Morgan AI Research, {`niccolo.dalmasso, shubham.x2.sharma, freddy.lecue, daniele.magazzeni, vamsi.k.potluru, tucker.balch, manuela.veloso`}`@jpmchase.com`

[‡] Work done while at J.P.Morgan AI Research.

[⋆] Corresponding Author.

[29], enforcing the distribution of an outcome of a machine learning model to not differ dramatically across different subgroups in the data.

Current methodologies for generating a smaller set of fair representative samples focus on the local characteristics of these samples with respect to the original dataset. For instance, [13, 30, 4, 24] obtain representative points by clustering while enforcing each cluster to include the same proportion of points from each subgroup in the original dataset. In another line of work, [33, 44, 53, 72, 12] create representative points by ensuring that points in the original dataset each have at least one representative point within a given distance in the feature space. While these methods can successfully reduce clustering cost and ensure a more evenly spread-out distribution of representative points in the feature space, it is unclear whether such representative samples can positively affect performance or discrimination reduction in downstream learning processes. As the induced distribution of the representative points might be far away from the original dataset distribution, downstream machine learning algorithm might lose significant performance due to this distribution shift, without necessarily reducing biases in the original data (as we also demonstrate empirically in our experiments).

Contributions In this work, we introduce **F**air **W**asserstein **C**oresets (`FWC`), a novel coreset approach that not only generates synthetic representative samples but also assigns sample-level weights to be used in downstream learning tasks. `FWC` generates synthetic samples by minimizing the Wasserstein distance between the distribution of the original datasets and that of the weighted synthetic samples, while simultaneously enforcing an empirical version of demographic parity. The Wasserstein distance is particularly suitable to this task due to its various connections with downstream learning processes and coresets generation (Section 2). Our contributions are as follows:

1. we show how the `FWC` optimization problem can be reduced to a nested minimization problem in which fairness constraints are equivalent to linear constraints (Section 3);
2. we develop an efficient majority minimization algorithm [56, 39] to solve the reformulated problem (Section 4). We analyze theoretical properties of our proposed algorithm and `FWC` (Section 5) and show that, in the absence of fairness constraints, our algorithm reduces to an equivalent version of Lloyd's algorithm for k-means and k-medians clustering, extending its applicability beyond fairness applications (Section 6);
3. we empirically validate the scalability and effectiveness of `FWC` by providing experiments on both synthetic and real datasets (Section 7). In downstream learning tasks, `FWC` result in competitive fairness-utility tradeoffs against current approaches, even when we enhance the fairness of existing approaches using fair pre-processing techniques, with an average disparity reduction of 53% and 18%, respectively. In addition, we show `FWC` can correct biases in large language models when passing coresets as examples to the LLM, reducing downstream disparities by 75% with GPT-3.5 [55] by 35% with GPT-4 [1]. Finally, we show `FWC` can improve downstream fairness-utility tradeoffs in downstream models when added to the training data (via data augmentation, see Appendix C.2).

Finally, we refer the reader to the Appendix for more details on the optimization problem (Section A), theoretical proofs (Section B) and further experiments and details (Section C).

2 Background and Related Work

Notation We indicate the original dataset samples $\{Z_i\}_{i=1}^n$, with $Z_i = (D_i, X_i, Y_i) \in \mathcal{Z} = (\mathcal{D} \times \mathcal{X} \times \mathcal{Y}) \subseteq \mathbb{R}^d$, where X indicates the non-sensitive features, D indicates one or more protected attributes such as ethnicity or gender, and Y is a decision outcome. In this work, we assume D and Y to be discrete features, i.e., to have a finite number of levels so that $|\mathcal{D}| \ll n$ and $|\mathcal{Y}| \ll n$. For example, Y might indicate a credit card approval decision based on credit history X, with D denoting sensitive demographic information. Given a set of weights $\{\theta\}_{i=1}^n$, define $p_{Z;\theta}$ the weighted distribution of a dataset $\{Z_i\}_{i=1}^n$ as $p_{Z;\theta} = \frac{1}{n}\sum_{i=1}^n \theta_i \delta_{Z_i}$, where δ_x indicates the Dirac delta distribution, i.e., the unit mass distribution at point $x \in \mathcal{X}$. Using this notation, we can express the empirical distribution of the original dataset by setting $\theta_i = e_i = 1$ for any i, i.e., $p_{Z;e} = \frac{1}{n}\sum_{i=1}^n e_i \delta_{Z_i}$. For a matrix $A, A^\top$ denotes its transpose. For two vectors (or matrices) $\langle u, v\rangle \overset{\text{def.}}{=} \sum_i u_i v_i$ is the canonical inner product (the Frobenius dot-product for matrices). We define $\mathbf{1}_m \overset{\text{def.}}{=} (1, \dots, 1) \in \mathbb{R}_+^m$.

Wasserstein distance and coresets Given two probability distributions p_1 and p_2 over a metric space $\mathcal{X}$, the Wasserstein distance, or optimal transport metric, quantifies the distance between the two distributions as solution of the following linear program (LP):

$$\mathcal{W}_c(p_1, p_2) \stackrel{\text{def.}}{=} \min_{\pi \in \Pi(p_1, p_2)} \int_{\mathcal{X} \times \mathcal{X}} c(x_1, x_2) \mathrm{d}\pi(x_1, x_2), \tag{1}$$

with $\Pi(p_1, p_2)$ indicating the set of all joint probability distributions over the product space $\mathcal{X} \times \mathcal{X}$ with marginals equal to (p_1, p_2) [35]. The operator $c(x, y)$ represents the "cost" of moving probability mass from x and y, and reduces to a matrix C if the underlying metric space $\mathcal{X}$ is discrete.

The Wasserstein distance has several connections with downstream learning processes and coresets. Firstly, the higher the Wasserstein distance between the weighted representative samples $p_{\hat{Z};\theta}$ and the original dataset $p_{Z;e}$, the higher the distribution shift, the more degradation we might expect in terms of downstream learning performance [62]. Secondly, the Wasserstein distance between two probability distributions can also be used to bound the deviation of functions applied to samples from such distributions. Define the following deviation:

$$d(p_{\hat{Z};\theta}, p_{Z;e}) \stackrel{\text{def.}}{=} \sup_{f \in \mathcal{F}} \left| \mathbb{E}_{z \sim p_{\hat{Z};\theta}} f(z) - \mathbb{E}_{z \sim p_{Z;e}} f(z) \right| .$$

When $\mathcal{F}$ is the class of Lipschitz-continuous functions with Lipschitz constant equal or less than 1, the deviation $d(p_{\hat{Z};\theta}, p_{Z;e})$ is equal to 1-Wasserstein distance $\mathcal{W}_1(p_{\hat{Z};\theta}, p_{Z;e})$ [66, 74]. The connection with learning processes and the downstream deviation $d(p_{(\hat{X},\hat{D});\theta}, p_{(X,D);e})$ is immediate when considering Lipschitz continuous classifiers with Lipschitz constant less than 1 (such as logistic regression or Lipschitz-constrained neural networks [2])[3]. For other classifiers, we note that the 1-Wasserstein distance still bounds the downstream discrepancy: in Proposition 2.1 we show that the 1-Wasserstein distance bounds the downstream discrepancy for ReLu-activated multilayer perceptrons (MLPs), which we use in our experiments on real datasets (Section 7).

Proposition 2.1. *Let $g_\psi \in \mathcal{G}^K$ be the class of K-layer multilayer perceptrons with ReLu activations. Then, the downstream discrepancy in downstream performance of g_ψ applied to samples from $p_{(\hat{X},\hat{D});\theta}$ and $p_{(X,D);e}$ is bounded by the 1-Wasserstein distance :*

$$d(p_{(\hat{X},\hat{D});\theta}, p_{(X,D);e}) = \sup_{g_\psi \in \mathcal{G}^{\mathcal{K}}} \left| \mathbb{E}_{(x,d) \sim p_{(\hat{X},\hat{D});\theta}} g_\psi(x, d) - \mathbb{E}_{(x,d) \sim p_{(X,D);e}} g_\psi(x, d) \right| \leq L_k \mathcal{W}_1(p_{\hat{Z};\theta}, p_{Z;e}), \tag{2}$$

where L_k is the MLP Lipschitz constant upper bound defined in [75, Section 6.1, Equation (8)].

We point out that minimizing the Wasserstein distance, hence bounding the downstream performance as in Equation (2), is equivalent to the definition of a measure coresets proposed by [15]. The Wasserstein distance is also connected to coresets in the sense of the best discrete approximation of a continuous distributions. Considering a feature space endowed with a continuous distribution, minimizing the p-Wasserstein distance across all the distributions of size m is topologically equivalent to identify the samples of size m that provide the best Voronoi tessellation of the space in L_p sense [57, 41]. Although other definitions of coresets using Wasserstein distance have been proposed in the literature, they require either to solve the underlying optimal transport problems or the knowledge of the downstream classifier and loss function [15, 46, 82, 43]. `FWC` is agnostic to any downstream model or loss function, and uses an efficient implementation that does not actually incur in the usual high cost connected to optimal transport. By adapting the approach proposed by [77], we solve an equivalent re-formulated linear optimization problem, which is more computationally tractable than classic approaches such as the simplex or the interior point method (see Section 4.1).

Demographic parity Also known as statistical parity, demographic parity (DP) imposes the decision outcome and protected attributes to be independent [18]. Using a credit card approval

[3] In such cases, $d(p_{(\hat{X},\hat{D});\theta}, p_{(X,D);e}) = \mathcal{W}_1(p_{(\hat{X},\hat{D});\theta}, p_{(X,D);e}) \leq \mathcal{W}_1(p_{\hat{Z};\theta}, p_{Z;e})$, see Lemma B.1.

decision example, demographic parity enforces an automatic decision process to approve similar proportion of applicants across different race demographics. DP is one of the most extensively analyzed fairness criterion; we refer the reader to [29] for a review. In this work, we use the demographic parity definition by [10], which enforces the ratio between the conditional distribution of the decision outcome across each subgroups $p(y|D = d)$ and the marginal distribution of the decision outcome $p(y)$ to be close to 1 in a given dataset. We refer to the "empirical version" of demographic parity to indicate that the conditional and marginal distributions are quantities estimated from the data. Note that the demographic parity definition we adopt enforces a condition on the weighted average of the conditional distributions across groups, which is different from a more recent approach of using Wasserstein distance, and specifically Wasserstein barycenters, to enforce demographic parity [25, 32, 23, 78].

3 FWC: Fair Wasserstein Coresets

Given a dataset $\{Z_i\}_{i=1}^n$, our goal is to find a set of samples $\{\hat{Z}_j\}_{j=1}^m$ and weights $\{\theta_j\}_{j=1}^m$ such that $m \ll n$ and that the Wasserstein distance between $p_{Z;e}$ and $p_{\hat{Z};\theta}$ is as small as possible. In addition, we use the fairness constraints proposed by [10] to control the demographic parity violation for the $p_{\hat{Z};\theta}$ distribution. Let $p_{\hat{Z};\theta}(y|d)$ indicate the conditional distribution $p_{\hat{Z};\theta}(\hat{Y} = y|\hat{D} = d)$. Imposing a constraint on the demographic disparity violation then reduces to requiring the conditional distribution under the weights $\{\theta_i\}_{i\in[n]}$ to be close to a target distribution p_{Y_T} across all possible values of the protected attributes D,

$$J\left(p_{\hat{Z};\theta}(y|d), p_{Y_T}(y)\right) = \left|\frac{p_{\hat{Z};\theta}(y|d)}{p_{Y_T}(y)} - 1\right| \leq \epsilon,\ \forall\, d \in \mathcal{D}, y \in \mathcal{Y}, \tag{3}$$

where ϵ is a parameter that determines the maximum fairness violation, and $J(\cdot,\cdot)$ is the probability ratio between distributions as defined in [10].

Using the notation above, our goal can then be formulated as the following optimization problem:

$$\begin{aligned} \min_{\theta\in\Delta_m, \hat{Z}\in\mathcal{Z}^m} \quad & \mathcal{W}_c(p_{\hat{Z};\theta}, p_{Z;e}) \\ \text{s.t.} \quad & J\left(p_{\hat{Z};\theta}(y|d), p_{Y_T}(y)\right) \leq \epsilon,\ \forall\, d \in \mathcal{D}, y \in \mathcal{Y}, \end{aligned} \tag{4}$$

where Δ_m indicates the set of valid weights $\{\theta \in \mathbb{R}_+^m : \sum_{i=1}^m \theta_i = m\}$. Note that the optimization problem in (4) shares some similarities with the optimization problem in [77] in using the Wasserstein distance as a distance metric between distributions and using (3) to enforce demographic parity. However, [77] only provide sample-level integer weights for the original dataset and do not generate any new samples, while our approach provides a separate set of samples $\{\hat{Z}_j\}_{j=1}^m$ with associated real-valued weights $\{\theta_j\}_{j=1}^m$, with $m \ll n$.

We now take the following steps to solve the optimization problem in (4): (i) we reduce the dimensionality of the feasible set by fixing $\hat{Y}$ and $\hat{D}$ a priori, (ii) we formulate the fairness constraints as linear constraints, (iii) we add artificial variables to express the objective function and (iv) we simplify the optimization problem to minimizing a continuous non-convex function of the $\{\hat{X}_j\}_{j=1}^m$.

Step 1. Reduce the feasible set of the optimization problem As in practice all possible Y_i and D_i are known a priori, and there are only a limited number of them, we can avoid optimizing over them and instead manually set the proportion of each combination of $\hat{Y}$ and $\hat{D}$. This reduces the optimization problem feasible set only over Δ_m and $\mathcal{X}^m$. The following lemma shows that this in fact does not affect the optimization problem:

Lemma 3.1. *For any $m > 0$, the best fair Wasserstein coreset formed by m data points $\{\hat{Z}_i : i \in [m]\}$ is no better (i.e., the optimal Wasserstein distance value is no lower) than the best fair Wasserstein coreset formed by $m|\mathcal{D}||\mathcal{Y}|$ data points $\{(d, X_i, y)_i : i \in [m], d \in \mathcal{D}, y \in \mathcal{Y}\}$.*

Hence, we simply set the proportions of $\{(\hat{D}_i, \hat{Y})_i\}_{i\in[m]}$ in the coresets to be similar to their respective proportions in the original dataset. The optimization problem then reduces to

$$\begin{aligned} \min_{\theta\in\Delta_m, \hat{X}\in\mathcal{X}^m} \quad & \mathcal{W}_c(p_{\hat{Z};\theta}, p_{Z;e}) \\ \text{s.t.} \quad & J\left(p_{\hat{Z};\theta}(y|d), p_{Y_T}(y)\right) \leq \epsilon,\ \forall\, d\in\mathcal{D}, y\in\mathcal{Y}\ , \end{aligned} \tag{5}$$

whose solutions are the features in the coreset $\{\hat{X}_j\}_{j=1}^m$ and the corresponding weights $\{\theta_j\}_{j=1}^m$.

Step 2. Equivalent linear constraints Following [77], the fairness constraint in Equation (3) can be expressed as $2|\mathcal{Y}||\mathcal{D}|$ linear constraints on the weights θ, as the disparity reduces to the following for all $d\in\mathcal{D}, y\in\mathcal{Y}$:

$$\sum_{i\in[m]:\hat{D}_i=d,\hat{Y}_i=y} \theta_i \leq (1+\epsilon)\cdot p_{Y_T}(y)\cdot \sum_{i\in[m]:\hat{D}_i=d} \theta_i\ , \quad \sum_{i\in[m]:\hat{D}_i=d,\hat{Y}_i=y} \theta_i \geq (1-\epsilon)\cdot p_{Y_T}(y)\cdot \sum_{i\in[m]:\hat{D}_i=d} \theta_i\ .$$

We can express these by using a $2|\mathcal{Y}||\mathcal{D}|$-row matrix A as $A\theta\geq\mathbf{0}$.

Step 3. Reformulate the objective function by introducing artificial variables When keeping the samples $\hat{X}$ fixed, we can follow [60] to derive an equivalent formulation of the Wasserstein distance in the objective as a linear program with mn variables. By indicating the transportation cost matrix $C(\hat{X})$, we define its components as follows,

$$C(\hat{X})_{ij} \overset{\text{def.}}{=} c(Z_i, \hat{Z}_j),\ \text{for } i\in[n], j\in[m]\ .$$

Note that $C(\hat{X})$ is a convex function of $\hat{X}$ when, e.g., using any L^p norm to define the transportation cost. Therefore, now the problem (5) is equivalent to

$$\begin{aligned} \min_{\hat{X}\in\mathcal{X}^m, \theta\in\Delta_m, P\in\mathbb{R}^{n\times m}} \quad & \langle C(\hat{X}), P\rangle \\ \text{s.t.} \quad & P\mathbf{1}_m = \frac{1}{n}\cdot\mathbf{1}_n,\ P^\top\mathbf{1}_n = \frac{1}{m}\cdot\theta,\ P\geq\mathbf{0},\ A\theta\geq\mathbf{0}\ . \end{aligned} \tag{6}$$

Step 4. Reduce to an optimization problem of $\hat{X}$ As from one of the constraints we get $\theta = m\cdot P^\top\mathbf{1}_n$, we further simplify problem (6) as:

$$\begin{aligned} \min_{\hat{X}\in\mathcal{X}^m, P\in\mathbb{R}^{n\times m}} \quad & \langle C(\hat{X}), P\rangle \\ \text{s.t.} \quad & P\mathbf{1}_m = \frac{1}{n}\cdot\mathbf{1}_n,\ P\geq\mathbf{0},\ AP^\top\mathbf{1}_n\geq\mathbf{0}\ . \end{aligned} \tag{7}$$

Let $F(C)$, as a function F of C, denote the optimal objective value of the following optimization problem

$$\begin{aligned} \min_{P\in\mathbb{R}^{n\times m}} \quad & \langle C, P\rangle \\ \text{s.t.} \quad & P\mathbf{1}_m = \frac{1}{n}\cdot\mathbf{1}_n,\ P\geq\mathbf{0},\ AP^\top\mathbf{1}_n\geq\mathbf{0} \end{aligned} \tag{8}$$

and then problem (7) is equivalent to

$$\min_{\hat{X}\in\mathcal{X}^m} \quad F(C(\hat{X}))\ . \tag{9}$$

In (9) the objective is continuous but nonconvex with respect to $\hat{X}$. Once the optimal $\hat{X}^\star$ is solved, then the optimal $P^\star$ of the problem (7) is obtained by solving problem (8) with C replaced with $C(\hat{X}^\star)$. Finally, the optimal $\theta^\star$ follows by the equation $\theta^\star = m\cdot(P^\star)^\top\mathbf{1}_n$. We now provide a majority minimization algorithm for solving problem (9).

4 Majority Minimization for Solving the Reformulated Problem

Majority minimization aims at solving nonconvex optimization problems, and refers to the process of defining a convex surrogate function that upper bounds the nonconvex objective function, so that optimizing the surrogate function improves the objective function [56, 39]. As the algorithm proceeds, the surrogate function also updates accordingly, which ensures the value of the original objective function keeps decreasing. Following this framework, we define the surrogate function $g(\cdot; \hat{X}^k)$ as follows for the k-th iterate $\hat{X}^k \in \mathcal{X}^m$:

$$g(\hat{X}; \hat{X}^k) \stackrel{\text{def.}}{=} \langle C(\hat{X}), P_k^\star \rangle \,, \tag{10}$$

in which $P_k^\star$ is the minimizer of problem (8) with the cost $C = C(\hat{X}^k)$[4].

With this surrogate function, Algorithm 1 summarizes the overall algorithm to minimize problem (9). In each iteration of Algorithm 1, line 3 is straightforward since it only involves computing the new cost matrix using the new feature vectors $\hat{X}^k$. We separately discuss how to solve the optimization problems in lines 4 and 5 below.

Algorithm 1 Majority Minimization for Solving (9)

1: Initial feature vectors $\hat{X}^k$ and $k = 0$
2: **while** True **do**
3: $C \leftarrow C(\hat{X}^k)$; ▷ *update the cost matrix C*
4: $P_k^\star \leftarrow$ optimal solution of problem (8); ▷ *updating the surrogate function (Section 4.1)*
5: $\hat{X}^{k+1} \leftarrow \arg\min_{\hat{X} \in \mathcal{X}^m} g(\hat{X}; \hat{X}^k)$; ▷ *updating feature vectors (Section 4.2)*
6: **if** $g(\hat{X}^{k+1}; \hat{X}^k) = g(\hat{X}^k; \hat{X}^k)$ **then**
7: $\theta_k^\star \leftarrow m \cdot (P_k^\star)^\top \mathbf{1}_n$; ▷ *if algorithm has converged, compute optimal weights*
8: **return** $\hat{X}^k$, $\theta_k^\star$ ▷ *return coresets and sample-level weights*
9: **end if**
10: $k \leftarrow k + 1$;
11: **end while**

4.1 Updating the Surrogate Function (Line 4)

To update the surrogate function, we need to solve problem (8), which is a large-scale linear program. Rather than solving the computationally prohibitive dual problem we solve a lower-dimensional dual problem by using a variant of the `FairWASP` algorithm proposed by [77]. We adapt `FairWASP` for cases where $m \neq n$ to find the solution of (8) via applying the cutting plane methods on the Lagrangian dual problems with reduced dimension[5]. We choose `FairWASP` over established commercial solvers due to its computational complexity being lower than other state of the art approaches such as interior-point or simplex method; see Lemma A.2 in Appendix A for more details.

4.2 Updating Feature Vectors (Line 5)

To update the feature vectors, we need to obtain the minimizer of the surrogate function $g(\hat{X}; \hat{X}^k)$, i.e.,

$$\min_{\hat{X} \in \mathcal{X}^m} g(\hat{X}; \hat{X}^k) \,. \tag{11}$$

The above can be written as the following problem:

$$\min_{\hat{X}_j \in \mathcal{X}: j \in [m]} \sum_{i \in [n]} \sum_{j \in [m]} c(Z_i, \hat{Z}_j) P_{ij} \tag{12}$$

for $P = P_k^\star$, in which each component of P is nonnegative and $\hat{Z}_j = (\hat{d}_j, \hat{X}_j, \hat{y}_j)$, for the known fixed $\hat{d}_j$ and $\hat{y}_j$. Furthermore, the matrix P is sparse, containing at most n non-zeros (as when

[4] We show this surrogate function is adequate, i.e., is convex and an upper bound of the original objective function, in Section 5.

[5] As opposed to the scenario where $m = n$, which was tackled by [77].

updating $P_k^\star$ for problem (8), see Appendix A). Moreover, problem (12) can be separated into the following m subproblems,

$$\min_{\hat{X}_j \in \mathcal{X}} \sum_{i \in [n]} c(Z_i, \hat{Z}_j) P_{ij} \ , \ \text{for } j \in [m] \ . \tag{13}$$

Each subproblem computes the weighted centroid of $\{Z_i : i \in [n], P_{ij} > 0\}$ under the distance function c. Therefore, (11) is suitable for parallel and distributed computing. Additionally, since the cost matrix $C(\hat{X})$ is a convex function of $\hat{X}$, each subproblem is a convex problem so gradient-based methods could converge to global minimizers. Furthermore, under some particular conditions, solving these small subproblems can be computationally cheap:

1. If $\mathcal{X}$ is convex and $c(Z, \hat{Z}) \overset{\text{def.}}{=} \|Z - \hat{Z}\|_2^2$, then the minimizer of (13) is the weighted average $\sum_{i \in [n]} P_{ij} X_i / \sum_{i \in [n]} P_{ij}$.
2. If $\mathcal{X}$ is convex and $c(Z, \hat{Z}) \overset{\text{def.}}{=} \|Z - \hat{Z}\|_1$, then the minimizer of (13) requires sorting the costs coordinate-wisely and finding the median.
3. If creating new feature vectors is not permitted and $\mathcal{X} = \{X_i : i \in [n]\}$, solving (13) requires finding the smallest $\sum_{i \in [n]: P_{ij} \neq 0} c(Z_i, (\hat{d}_j, X, \hat{y}_j)) P_{ij}$ for X within the finite set $\mathcal{X}$. The matrix P is highly sparse so this operation is not computationally expensive.

5 Theoretical Guarantees

In this section we provide theoretical insights on `FWC` complexity, convergence behavior of Algorithm 1 as well as generalizability of `FWC` performance on unseen test sets.

5.1 Computational Complexity

First, we consider the `FairWASP` variant used in Algorithm 1, line 4. The initialization requires $O(mn)$ flops and uses $O(n|Y||D|)$ space for storing the cost matrix. After that, the per-iteration time and space complexities are both only $O(n|Y||D|)$. Lemma 5.1 analyzes the computational complexity of our adaptation of the `FairWASP` algorithm when solving problem (8).

Lemma 5.1. *With efficient computation and space management, the cutting plane method has a computational complexity of*

$$\tilde{O}\left(nm + |\mathcal{D}|^2 |\mathcal{Y}|^2 n \cdot \log(R/\epsilon)\right) \tag{14}$$

flops and $O(n|\mathcal{D}||\mathcal{Y}|)$ space. Here R denotes the size of an optimal dual solution of (8), and $\tilde{O}(\cdot)$ absorbs m, n, $|\mathcal{D}|$, $|\mathcal{Y}|$ in the logarithm function.

Hence, the overall complexity of `FWC` is $\tilde{O}(mn + |\mathcal{D}|^2 |\mathcal{Y}|^2 n \cdot \log(R/\epsilon))$. Note that in practice, both $|\mathcal{D}|$ and $|\mathcal{Y}|$ are very small compared with the coreset size m and dataset size n, so the overall complexity is almost as low as $O(mn)$.

5.2 Convergence Guarantees

First, we establish that our proposed surrogate function is indeed convex and a valid upper bound. We then show Algorithm 1 converges to a first-order stationary point, within finite iterations if the minimizer of problem (11) is unique. Note that because $g(\hat{X}; \hat{X}^k) = \langle C(\hat{X}), P_k^\star \rangle$, the minimizer is unique whenever the cost matrix $C(\cdot)$ is strongly convex.

Lemma 5.2. *The function $g(\hat{X}; \hat{X}^k)$ is convex function of $\hat{X}$ and a valid upper bound, i.e., $g(\hat{X}; \hat{X}^k) \geq F(C(\hat{X}))$. This inequality holds at equality when $\hat{X} = \hat{X}^k$.*

Theorem 5.3. *The objective value is monotonically decreasing, i.e., $F(C(\hat{X}^{k+1})) \leq F(C(\hat{X}^k))$ for any $k \geq 0$. And once the algorithm stops and $C(\hat{X}^k)$ is smooth at $\hat{X}^k$, then $\hat{X}^k$ is a first-order stationary point of (9).*

Theorem 5.4. *When the minimizer of (11) is unique, Algorithm 1 terminates within finite iterations.*

5.3 Generalization Guarantees

Proposition 5.5 below bounds the distance and demographic parity between the FWC samples and the true underlying distribution of the data, from which the original dataset of size n was observed. This generalizes the performance of FWC to unseen test sets sampled from the data generating distribution.

Proposition 5.5. *Let λ indicate the distance between $p_{\hat{Z};\theta}$ and $p_{Z;e}$ after convergence of FWC i.e., $\mathcal{W}(p_{\hat{Z};\theta}, p_{Z;e}) = \lambda$. Let q_Z be the true underlying distribution of the data supported over $\mathbb{R}^d$, with marginal distribution over y bounded away from zero, so that $\rho = \min_{y\in\mathcal{Y}} q_Z(y) > 0$. Then with probability $1-\alpha$:*

$$\mathcal{W}_c(p_{\hat{Z}}, q_Z) \leq \lambda + \mathcal{O}(\log(1/\alpha)^{1/d} n^{-1/d}) \tag{15}$$

$$\sup_{y\in\mathcal{Y}, d\in\mathcal{D}} J(p_{\hat{Z}}(y|d), q_Y(y)) \leq \frac{\epsilon}{\rho} + \mathcal{O}\left(\sqrt{\frac{\log 2/\alpha}{n\rho^2}}\right) \tag{16}$$

In addition, in Appendix B.1 we consider the task of learning using FWC samples. We show that the error in downstream learning tasks can be seen as the sum of (i) the approximation error FWC samples make with respect to the original dataset and (ii) how well $\hat{Y}$ can be learnt from $\hat{X}$ and $\hat{D}$ from FWC samples. However, as FWC samples $\{\hat{Z}_j\}_{j=1}^m$ are not i.i.d., standard sample complexity results in e.g., empirical risk minimization, do not apply, highlighting the hardness in developing finite-sample learning bounds in this setting.

6 An Alternative View: Generalized clustering algorithm

When the fairness constraints are absent, problem (8) reduces to:

$$\begin{array}{ll} \min_{P\in\mathbb{R}^{n\times m}} & \langle C, P\rangle \\ \text{s.t.} & P\mathbf{1}_m = \frac{1}{n}\cdot\mathbf{1}_n,\ P \geq \mathbf{0}_{n\times m}\ . \end{array} \tag{17}$$

The minimizer $P^\star$ of (17) can be written in closed form. For each $i \in [n]$, let $C_{ij_i^\star}$ denote a smallest component on the i-th row of C. Then the components of a minimizer $P^\star$ can be written as $P^\star_{ij} = \frac{1}{n}\cdot\mathbf{I}(j = j_i^\star)$ (where $\mathbf{I}$ is the indicator function). Hence, without fairness constraints, FWC corresponds to Lloyd's algorithm for clustering. Specifically, Lloyd's algorithm iteratively computes the centroid for each subset in the partition and subsequently re-partitions the input based on the closeness to these centroids [42]; these are the same operations FWC does in optimizing the surrogate function and solving problem (17). Thus, when $c(x, y)$ is correspondingly defined as $\|x-y\|_1$ or $\|x-y\|_2^2$, FWC corresponds to Lloyd's algorithm applied to k-medians or k-means problems, except the centroids have fixed values for $\hat{D}$ and $\hat{Y}$ (see Section 3).

Comparison with k-means and k-medoids FWC and Lloyds' algorithm for k-means or k-median share similar per-iteration complexity, with the main difference in complexity due to solving problem (8). We solve this problem efficiently by utilizing a variant of the FairWASP approach by [77] (Section 4.1), hence avoiding the usual complexity in solving optimal transport problems. As shown in Section 5, the leading term in the runtime complexity is $\mathcal{O}(nm)$, which comes from calculating and storing the cost matrix C. This level of complexity is the same as those in k-means and k-medoids. In addition, from our experiments we also see that the per-iteration complexity of FWC is roughly linear with the original dataset size n (see the runtime experiment in Section 7 and Appendix C).

7 Experiments

Runtime analysis We evaluate the runtime performance of FWC by creating a synthetic dataset of dimension n and features of dimension p, with the goal of creating a coreset of size m (see Appendix C.1 for details). We fix two out of the three parameters to default values $(n, m, p) = (5000, 250, 25)$ and vary the other across suitable ranges, to analyse the runtime and total number of iterations. Figure 1, top left, and Table 2, in Appendix C.1, show the runtime and number of iterations when increasing the dataset size n from 1,000 to 1,000,000, with averages and standard deviations over 10 separate runs; both the runtime and number of iterations grow proportionally to the sample size n. Figure 2 in Appendix C.1 also shows that requiring a larger coreset size m implies the need of fewer iterations but longer iteration runtime, as more representatives need to be computed.

Real datasets results We evaluate the performance of FWC on 4 datasets widely used in the fairness literature [19]: (i) Adult [7], (ii) German Credit [28], (iii) Communities and Crime [64] and (iv) Drug [20]. For each dataset, we consider 3 different coreset sizes $m = 5\%, 10\%, 20\%$ (apart from the Adult dataset, in which we select m equal to 0.5%, 1% and 2% due to the large dataset size). We compare our approach with: (a) `Fairlets` and `IndFair`, two fair clustering approaches by [4] and [12], (b) `K-Median Coresets`, a coreset approach by [3], (c) k-means [42] and k-medoids [45, 58], two classic clustering approaches and (d) `Uniform Subsampling` of the original dataset. For FWC, we consider three different values of the fairness violation hyper-parameters ϵ for the optimization problem in (5). We compute the fairness-utility tradeoff by first training a 2-layer multilayer perceptron (MLP) classifier with ReLu activations on the coresets created by each approach and then evaluating the classifier demographic disparity (fairness) and AUC (utility). Figure 1 shows the model with the best fairness-utility tradeoff across the three coreset sizes m, for each approach. FWC obtains equal or better fairness-utility tradeoffs (smaller disparity at the same level of utility, higher utility with the same disparity, or both) across all datasets, and performance remains competitive even when using a fairness pre-processing approach [34]. Appendix C.2 includes more experiments and details, which highlight that: (a) FWC consistently achieves coresets that are closer in distribution to the original dataset with respect to the other methods and, although not natively minimizing clustering cost, also provide competitive performance for smaller datasets (Tables 3 and 4); (b) when added to the training data using the data augmentation schema proposed by [68, Section 2.1] FWC generally either increase the performance or reduce the demographic disparity in the downstream learning process.

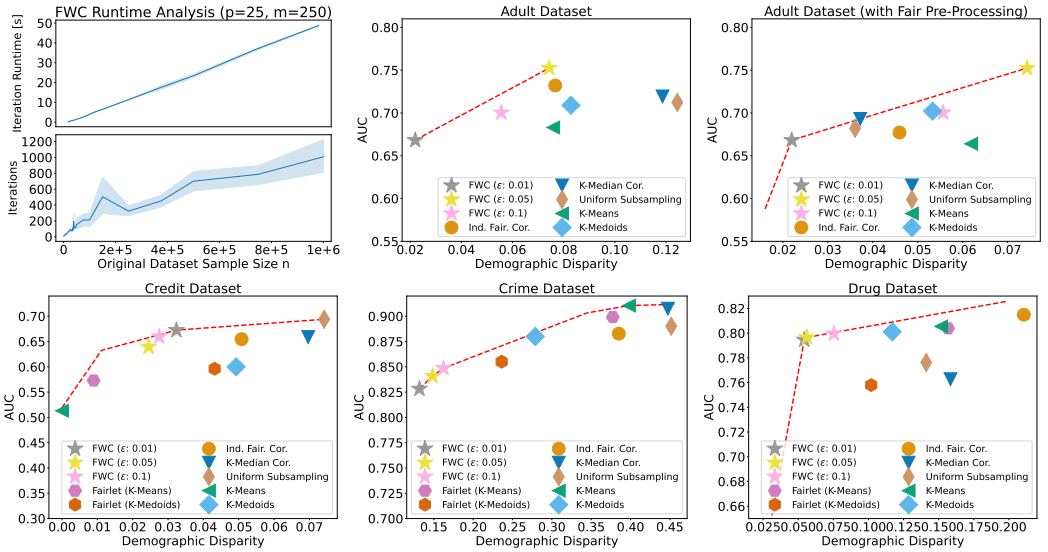

Figure 1: *Top left:* FWC runtime when changing the original dataset size n. *Others*: Fairness-utility tradeoff on real datasets for a downstream MLP classifier, selecting the model with the best fairness-utility tradeoff across three different coreset sizes m, with averages taken over 10 runs. FWC consistently achieves a comparable/better tradeoff as shown by the Pareto frontier (dashed red line, computed over all models and coreset sizes), even when adjusting the other coresets with a fairness pre-processing technique [33]. See text and Appendix C.2 for more details.

Using FWC to improve fairness for LLM [76] evaluate GPT-3.5 and GPT-4 for fairness on predictive tasks for the UCI Adult dataset in a zero and few shot setting. We use a similar evaluation setup and use FWC in the few shot setting as examples and evaluate the results for the gender protected attribute. Specifically, we transform the tabular data into language descriptions, and ask GPT-3.5 Turbo and GPT-4 to perform classification tasks on it. We select 200 samples to construct the test set and use a set of 16 samples found using FWC as examples. Further details on this experiment are provided in the appendix. The results are shown in Table 1. Examples provided by FWC help reduce demographic disparity more than providing balanced few shot examples, while losing on predictive accuracy (note that the drop in accuracy is similar to the drop observed in [76] and is representative of the fairness-utility trade-off). Owing to the token limitation of LLM's, these representative coresets

can evidently be valuable to provide a small set of samples that can help mitigate bias. When accounting for the standard deviations for demographic parity in Table 1, FWC reduces the LLM bias when compared to zero shot prompting for GPT-4 across all runs. For GPT-3.5 Turbo, while the average disparity is reduced across runs, such consistency is indeed not observed, owing to a diverse set of outputs from the large language model. Due to limited availability of computational resources (associated with querying these models), we leave a more thorough evaluation across different datasets and models to future work.

Table 1: Using the same setup as in [76], we use GPT-3.5 Turbo and GPT-4 LLM's for fairness evaluations, with a test set of 200 samples with 0.5 base parity ($b_p = 0.5$). Few Shot - FWC is used to provide sixteen examples with weights to the model as examples. Accuracy and demographic disparity (DP) are based on the resulting predictions from GPT-3.5 and GPT-4 models.

Adult Dataset	**Zero Shot**		**Few Shot** ($b_p = 0$)		**Few Shot** (FWC)	
	Accuracy	DP	Accuracy	DP	Accuracy	DP
GPT-3.5 Turbo	53.55 ± 0.87	0.040 ± 0.017	57.99 ± 1.88	0.019 ± 0.015	55.02 ± 1.26	$\mathbf{0.010 \pm 0.03}$
GPT-4	76.54 ± 1.63	0.42 ± 0.016	74.39 ± 2.86	0.33 ± 0.09	65.20 ± 0.85	$\mathbf{0.27 \pm 0.04}$

8 Discussion and Conclusions

We introduce FWC, a novel coreset approach that generates synthetic representative samples along with sample-level weights for downstream learning tasks. FWC minimizes the Wasserstein distance between the distribution of the original datasets and that of the weighted synthetic samples while enforcing demographic parity. We demonstrate the effectiveness and scalability of FWC through experiments conducted on both synthetic and real datasets, as well as reducing biases in LLM predictions (GPT 3.5 and GPT 4). Future extensions include: (i) targeting different fairness metrics such as equalized odds [27, 49] as well as robustness of fairness-performance tradeoff over distribution shifts [48, 67], (ii) exploring privacy and explainability properties of FWC [50, 51], (iii) utilizing coresets for accelerating gradient descents algorithms and test their convergence [47, 70] (iv) reformulating the optimization framework to target deep neural network pruning [52, 54] and (v) investigate applications of fair synthetic data in the financial sector [61].

Disclaimer

This paper was prepared for informational purposes by the Artificial Intelligence Research group of JPMorgan Chase & Co. and its affiliates ("JP Morgan") and is not a product of the Research Department of JP Morgan. JP Morgan makes no representation and warranty whatsoever and disclaims all liability, for the completeness, accuracy or reliability of the information contained herein. This document is not intended as investment research or investment advice, or a recommendation, offer or solicitation for the purchase or sale of any security, financial instrument, financial product or service, or to be used in any way for evaluating the merits of participating in any transaction, and shall not constitute a solicitation under any jurisdiction or to any person, if such solicitation under such jurisdiction or to such person would be unlawful.

References

[1] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[2] C. Anil, J. Lucas, and R. Grosse. Sorting out lipschitz function approximation. In *International Conference on Machine Learning*, pages 291–301. PMLR, 2019.

[3] O. Bachem, M. Lucic, and S. Lattanzi. One-shot coresets: The case of k-clustering. In *International Conference on Artificial Intelligence and Statistics*, pages 784–792. PMLR, 2018.

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

[5] S. Barocas, M. Hardt, and A. Narayanan. *Fairness and Machine Learning: Limitations and Opportunities*. MIT Press, 2023.

[6] G. Basso. A hitchhiker's guide to wasserstein distances. *Online manuscript available at https://api. semanticscholar. org/CorpusID*, 51801464, 2015.

[7] B. Becker and R. Kohavi. Adult. UCI Machine Learning Repository, 1996. DOI: https://doi.org/10.24432/C5XW20.

[8] D. Bertsimas and J. N. Tsitsiklis. *Introduction to linear optimization*, volume 6. Athena scientific Belmont, MA, 1997.

[9] Z. Borsos, M. Mutny, and A. Krause. Coresets via bilevel optimization for continual learning and streaming. *Advances in Neural Information Processing Systems*, 33:14879–14890, 2020.

[10] F. Calmon, D. Wei, B. Vinzamuri, K. Natesan Ramamurthy, and K. R. Varshney. Optimized pre-processing for discrimination prevention. *Advances in Neural Information Processing Systems*, 30, 2017.

[11] T. Campbell and T. Broderick. Bayesian coreset construction via greedy iterative geodesic ascent. In *International Conference on Machine Learning*, pages 698–706. PMLR, 2018.

[12] R. Chhaya, A. Dasgupta, J. Choudhari, and S. Shit. On coresets for fair regression and individually fair clustering. In *International Conference on Artificial Intelligence and Statistics*, pages 9603–9625. PMLR, 2022.

[13] F. Chierichetti, R. Kumar, S. Lattanzi, and S. Vassilvitskii. Fair clustering through fairlets. *Advances in Neural Information Processing Systems*, 30, 2017.

[14] A. Chouldechova. Fair prediction with disparate impact: A study of bias in recidivism prediction instruments. *Big Data*, 5(2):153–163, 2017.

[15] S. Claici, A. Genevay, and J. Solomon. Wasserstein measure coresets. *arXiv preprint arXiv:1805.07412*, 2018.

[16] C. Coleman, C. Yeh, S. Mussmann, B. Mirzasoleiman, P. Bailis, P. Liang, J. Leskovec, and M. Zaharia. Selection via proxy: Efficient data selection for deep learning. In *International Conference on Learning Representations*, 2020.

[17] A. Dvoretzky, J. Kiefer, and J. Wolfowitz. Asymptotic minimax character of the sample distribution function and of the classical multinomial estimator. *The Annals of Mathematical Statistics*, pages 642–669, 1956.

[18] C. Dwork, M. Hardt, T. Pitassi, O. Reingold, and R. Zemel. Fairness through awareness. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*, pages 214–226, 2012.

[19] A. Fabris, S. Messina, G. Silvello, and G. A. Susto. Algorithmic fairness datasets: The story so far. *Data Mining and Knowledge Discovery*, 36(6):2074–2152, 2022.

[20] E. Fehrman, A. K. Muhammad, E. M. Mirkes, V. Egan, and A. N. Gorban. The five factor model of personality and evaluation of drug consumption risk. In *Data Science: Innovative Developments in Data Analysis and Clustering*, pages 231–242. Springer, 2017.

[21] D. Feldman. Core-sets: Updated survey. *Sampling Techniques for Supervised or Unsupervised Tasks*, pages 23–44, 2020.

[22] N. Fournier and A. Guillin. On the rate of convergence in wasserstein distance of the empirical measure. *Probability theory and related fields*, 162(3):707–738, 2015.

[23] S. Gaucher, N. Schreuder, and E. Chzhen. Fair learning with wasserstein barycenters for non-decomposable performance measures. In *International Conference on Artificial Intelligence and Statistics*, pages 2436–2459. PMLR, 2023.

[24] M. Ghadiri, S. Samadi, and S. Vempala. Socially fair k-means clustering. In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pages 438–448, 2021.

[25] P. Gordaliza, E. Del Barrio, G. Fabrice, and J.-M. Loubes. Obtaining fairness using optimal transport theory. In *International conference on machine learning*, pages 2357–2365. PMLR, 2019.

[26] S. Har-Peled and S. Mazumdar. On coresets for k-means and k-median clustering. In *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, pages 291–300, 2004.

[27] M. Hardt, E. Price, and N. Srebro. Equality of opportunity in supervised learning. *Advances in neural information processing systems*, 29, 2016.

[28] H. Hofmann. Statlog (German Credit Data). UCI Machine Learning Repository, 1994. DOI: https://doi.org/10.24432/C5NC77.

[29] M. Hort, Z. Chen, J. M. Zhang, M. Harman, and F. Sarro. Bias mitigation for machine learning classifiers: A comprehensive survey. *ACM J. Responsib. Comput.*, nov 2023.

[30] L. Huang, S. Jiang, and N. Vishnoi. Coresets for clustering with fairness constraints. *Advances in Neural Information Processing Systems*, 32, 2019.

[31] H. Jiang, Y. T. Lee, Z. Song, and S. C.-w. Wong. An improved cutting plane method for convex optimization, convex-concave games, and its applications. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, pages 944–953, 2020.

[32] R. Jiang, A. Pacchiano, T. Stepleton, H. Jiang, and S. Chiappa. Wasserstein fair classification. In *Uncertainty in artificial intelligence*, pages 862–872. PMLR, 2020.

[33] C. Jung, S. Kannan, and N. Lutz. Service in your neighborhood: Fairness in center location. *Foundations of Responsible Computing (FORC)*, 2020.

[34] F. Kamiran and T. Calders. Data preprocessing techniques for classification without discrimination. *Knowledge and Information Systems*, 33(1):1–33, 2012.

[35] L. Kantorovitch. On the translocation of masses. *Management Science*, 5(1):1–4, 1958.

[36] L. G. Khachiyan. Polynomial algorithms in linear programming. *USSR Computational Mathematics and Mathematical Physics*, 20(1):53–72, 1980.

[37] Y.-g. Kim, K. Lee, and M. C. Paik. Conditional wasserstein generator. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[38] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[39] K. Lange. *MM optimization algorithms*. SIAM, 2016.

[40] S. Lei and D. Tao. A comprehensive survey of dataset distillation. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 46(01):17–32, jan 2024.

[41] Y. Liu and G. Pagès. Characterization of probability distribution convergence in Wasserstein distance by L^p-quantization error function. *Bernoulli*, 26(2):1171 – 1204, 2020.

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

[43] N. Loo, R. Hasani, A. Amini, and D. Rus. Efficient dataset distillation using random feature approximation. *Advances in Neural Information Processing Systems*, 35:13877–13891, 2022.

[44] S. Mahabadi and A. Vakilian. Individual fairness for k-clustering. In *International Conference on Machine Learning*, pages 6586–6596. PMLR, 2020.

[45] F. E. Maranzana. On the location of supply points to minimize transportation costs. *IBM Systems Journal*, 2(2):129–135, 1963.

[46] B. Mirzasoleiman, J. Bilmes, and J. Leskovec. Coresets for data-efficient training of machine learning models. In *International Conference on Machine Learning*, pages 6950–6960. PMLR, 2020.

[47] B. Mirzasoleiman, J. Bilmes, and J. Leskovec. Coresets for data-efficient training of machine learning models. In H. D. III and A. Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 6950–6960. PMLR, 13–18 Jul 2020.

[48] A. Mishler and N. Dalmasso. Fair when trained, unfair when deployed: Observable fairness measures are unstable in performative prediction settings. *arXiv preprint arXiv:2202.05049*, 2022.

[49] A. Mishler, E. H. Kennedy, and A. Chouldechova. Fairness in risk assessment instruments: Post-processing to achieve counterfactual equalized odds. In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pages 386–400, 2021.

[50] P. Mohassel, M. Rosulek, and N. Trieu. Practical privacy-preserving k-means clustering. *Proceedings on privacy enhancing technologies*, 2020.

[51] M. Moshkovitz, S. Dasgupta, C. Rashtchian, and N. Frost. Explainable k-means and k-medians clustering. In H. D. III and A. Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 7055–7065. PMLR, 13–18 Jul 2020.

[52] B. Mussay, M. Osadchy, V. Braverman, S. Zhou, and D. Feldman. Data-independent neural pruning via coresets. In *International Conference on Learning Representations*, 2020.

[53] M. Negahbani and D. Chakrabarty. Better algorithms for individually fair k-clustering. *Advances in Neural Information Processing Systems*, 34:13340–13351, 2021.

[54] R. Ohib, N. Gillis, N. Dalmasso, S. Shah, V. K. Potluru, and S. Plis. Explicit group sparse projection with applications to deep learning and NMF. *Transactions on Machine Learning Research, arXiv:1912.03896*, 2022.

[55] OpenAI. Chatgpt3.5. *https://chat.openai.com*, 2022.

[56] J. M. Ortega and W. C. Rheinboldt. *Iterative solution of nonlinear equations in several variables*. SIAM, 2000.

[57] G. Pagès. Introduction to vector quantization and its applications for numerics. *ESAIM: proceedings and surveys*, 48:29–79, 2015.

[58] H.-S. Park and C.-H. Jun. A simple and fast algorithm for k-medoids clustering. *Expert Systems with Applications*, 36(2):3336–3341, 2009.

[59] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12(Oct):2825–2830, 2011.

[60] G. Peyré, M. Cuturi, et al. Computational optimal transport: With applications to data science. *Foundations and Trends® in Machine Learning*, 11(5-6):355–607, 2019.

[61] V. K. Potluru, D. Borrajo, A. Coletta, N. Dalmasso, Y. El-Laham, E. Fons, M. Ghassemi, S. Gopalakrishnan, V. Gosai, E. Kreačić, et al. Synthetic data applications in finance. *arXiv preprint arXiv:2401.00081*, 2023.

[62] J. Quiñonero-Candela, M. Sugiyama, A. Schwaighofer, and N. D. Lawrence. *Dataset shift in machine learning*. Mit Press, 2022.

[63] Y. P. Raykov, A. Boukouvalas, F. Baig, and M. A. Little. What to do when k-means clustering fails: a simple yet principled alternative algorithm. *PloS one*, 11(9):e0162259, 2016.

[64] M. Redmond. Communities and Crime. UCI Machine Learning Repository, 2009. DOI: https://doi.org/10.24432/C53W3X.

[65] S. Sagiroglu and D. Sinanc. Big data: A review. In *2013 International Conference on Collaboration Technologies and Systems (CTS)*, pages 42–47. IEEE, 2013.

[66] F. Santambrogio. Optimal transport for applied mathematicians. *Birkäuser, NY*, 55(58-63):94, 2015.

[67] S. Sharma, J. Henderson, and J. Ghosh. Feamoe: fair, explainable and adaptive mixture of experts. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence*, IJCAI '23, 2023.

[68] S. Sharma, Y. Zhang, J. M. Rios Aliaga, D. Bouneffouf, V. Muthusamy, and K. R. Varshney. Data augmentation for discrimination prevention and bias disambiguation. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, pages 358–364, 2020.

[69] M. Sloane, E. Moss, and R. Chowdhury. A silicon valley love triangle: Hiring algorithms, pseudo-science, and the quest for auditability. *Patterns*, 3(2), 2022.

[70] M. Sordello, N. Dalmasso, H. He, and W. Su. Robust learning rate selection for stochastic optimization via splitting diagnostic. *Transactions on Machine Learning Research arXiv:1910.08597*, 2024.

[71] J. Thickstun. Kantorovich-rubinstein duality. *Online manuscript available at https://courses. cs. washington. edu/courses/cse599i/20au/resources/L12_ duality. pdf*, 2019.

[72] A. Vakilian and M. Yalciner. Improved approximation algorithms for individually fair clustering. In *International Conference on Artificial Intelligence and Statistics*, pages 8758–8779. PMLR, 2022.

[73] A. Vattani. K-means requires exponentially many iterations even in the plane. In *Proceedings of the twenty-fifth annual symposium on Computational geometry*, pages 324–332, 2009.

[74] C. Villani et al. *Optimal transport: Old and new*, volume 338. Springer, 2009.

[75] A. Virmaux and K. Scaman. Lipschitz regularity of deep neural networks: analysis and efficient estimation. *Advances in Neural Information Processing Systems*, 31, 2018.

[76] B. Wang, W. Chen, H. Pei, C. Xie, M. Kang, C. Zhang, C. Xu, Z. Xiong, R. Dutta, R. Schaeffer, et al. Decodingtrust: A comprehensive assessment of trustworthiness in gpt models. *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023.

[77] Z. Xiong, N. Dalmasso, A. Mishler, V. K. Potluru, T. Balch, and M. Veloso. FairWASP: Fast and optimal fair wasserstein pre-processing. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(14):16120–16128, Mar. 2024.

[78] S. Xu and T. Strohmer. Fair data representation for machine learning at the pareto frontier. *Journal of Machine Learning Research*, 24(331):1–63, 2023.

[79] R. Yin, Y. Liu, W. Wang, and D. Meng. Randomized sketches for clustering: Fast and optimal kernel k-means. *Advances in Neural Information Processing Systems*, 35:6424–6436, 2022.

[80] R. Yu, S. Liu, and X. Wang. Dataset distillation: A comprehensive review. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 46(01):150–170, jan 2024.

[81] A. Zhang, L. Xing, J. Zou, and J. C. Wu. Shifting machine learning for healthcare from development to deployment and from models to data. *Nature Biomedical Engineering*, 6(12):1330–1345, July 2022.

[82] B. Zhao, K. R. Mopuri, and H. Bilen. Dataset condensation with gradient matching. In *International Conference on Learning Representations*, 2021.

Appendix

A Details on Updating the Surrogate Function (line 4 of Algorithm 1)

To update the surrogate function, we need to solve problem (8), which is a huge-scale linear program with $O(n)$ constraints and $O(mn)$ nonnegative variables. In this work, we adapt `FairWASP` [77] for cases where $m \neq n$, as opposed to the scenario where $m = n$, which was tackled by [77]. Before showing the main idea of the algorithm, we rephrase a useful lemma for doing linear minimization on $S_{n,m} \overset{\text{def.}}{=} \{P \in \mathbb{R}^{n\times m} : P\mathbf{1}_m = \frac{1}{n}\cdot\mathbf{1}_n, P \geq \mathbf{0}\}$.

Lemma A.1. *For the function $G(C) \overset{\text{def.}}{=} \max_{P\in S_{n,m}}\langle C, P\rangle$, it is a convex function of C in $\mathbb{R}^{n\times m}$. For each $i \in [n]$, let $C_{ij_i^\star}$ denote a largest component on the i-th row of C, then $G(C) = \frac{1}{n}\sum_{i=1}^n C_{ij_i^\star}$. Define the components of $P^\star$ as follows:*

$$P^\star_{ij} = \begin{cases} 0 & \text{if } j \neq j_i^\star \\ \frac{1}{n} & \text{if } j = j_i^\star \end{cases} \tag{18}$$

and then $P^\star \in \arg\max_{P\in S_{n,m}}\langle C, P\rangle$ and $P^\star \in \partial G(C)$.

Proof. The proof of the above lemma is equivalent with that of Lemma 1 of [77] in the case when $m \neq n$, which can be extended directly. □

With this lemma, now we show how to efficiently solve problem (8) via its dual problem. Although (8) is of large scale and computationally prohibitive, it is equivalent to the following saddle point problem on the Lagrangian:

$$\min_{P\in S_{n,m}}\max_{\lambda\in\mathbb{R}^h_+} L(P,\lambda) \overset{\text{def.}}{=} \langle C, P\rangle - \lambda^\top A P^\top \mathbf{1}_n \tag{19}$$

where h is the number of rows of A, which is at most $2|\mathcal{Y}||\mathcal{D}|$. It should be mentioned that h is significantly smaller than mn; for example, for classification tasks with only two protected variables, h is no larger than 8, independent of the number of samples or features. Since $L(\cdot,\cdot)$ is bilinear, the minimax theorem guarantees that (19) is equivalent to $\max_{\lambda\in\mathbb{R}^h_+}\min_{P\in S_{n,m}} L(P,\lambda)$. This is further equal to the dual problem:

$$\max_{\lambda\in\mathbb{R}^h_+} -\Big[G(\lambda) \overset{\text{def.}}{=} \max_{P\in S_{n,m}}\Big\langle\sum_{j=1}^h \lambda_j \mathbf{1}_n a_j^\top - C, P\Big\rangle\Big], \tag{20}$$

in which $a_j^\top$ denotes the j-th row of A. Note that the problem (20) has much fewer decision variables than that of (19) and Lemma A.1 ensures the function $G(\cdot)$ is convex and has easily accessible function values and subgradients. Therefore, directly applying a cutting plane method has low per-iteration complexity and solves the problem (20) in linear time. We include the details on the cutting plane method in Section A.1 below. Finally, the primal optimal solution of (8) can be easily recovered from the dual optimal solution $\lambda^\star$ via solving $\max_{P\in S_{n,m}}\big\langle\sum_{j=1}^h \lambda_j^\star \mathbf{1}_n a_j^\top - C, P\big\rangle$, under the assumption that this problem has a unique minimizer, which almost always holds in practice for the computed $\lambda^\star$ and is also assumed by [77]. In this way, we have shown how the problem (8) can be efficiently solved by applying a cutting plane method on its dual problem.

A.1 Details of the Cutting Plane Method for Solving (20)

The cutting plane method is designed for convex problems where a *separation oracle* can be employed [36]. For any $\lambda \in \mathbb{R}^m$, a separation oracle operates by generating a vector g which satisfies $g^\top\lambda \geq g^\top\lambda^\star$ for all $\lambda^\star$ in the set of optimal solutions. By repeatedly applying the separation oracle to cut down the potential possible feasible set, the cutting plane method progressively narrows down the feasible solution space until it reaches convergence. The specific steps of the cutting plane algorithm are detailed in Algorithm 2, with the key distinctions among different versions of this method lying in how lines 3 and 4 are implemented.

Algorithm 2 General Cutting Plane Method for (20)

1: Choose a bounded set E_0 that contains an optimal solution
2: **for** k from 0 to n **do**
3: Choose an interior point λ^k of E_k;
4: Compute $g \in \mathbb{R}^m$ such that

$$g^\top \lambda^k \geq g^\top \lambda^\star \text{ for any optimal solution } \lambda^\star;$$

5: Choose the next bounded set $E_{k+1} \supseteq \{\lambda \in E_k : g^\top \lambda \leq g^\top \lambda^k\}$;
6: **end for**

For the problem (20), a separation oracle (line 4 in Algorithm 2) can directly use the vector of subgradients, which are efficiently accessible, as we mentioned in section 4. Given that we have shown (20) is a low-dimensional convex program with subgradient oracles, there exist many well-established algorithms that can be used. Suppose that the norm of an optimal $\lambda^\star$ is bounded by R, to the best of our knowledge, the cutting plane method with the best theoretical complexity is given by [31], who proposed an improved cutting plane method that only needs $O((h \cdot \mathrm{SO} + h^2) \cdot \log(hR/\epsilon))$ flops. Here SO denotes the complexity of the separation oracle. Note that here h is at most $2|\mathcal{D}||\mathcal{Y}|$, which is far smaller than n or m. Here we restate the Corollary 5 of [77] below for the overall time and space complexity of applying the cutting plane method in the case $m \neq n$.

Lemma A.2 (Essentially Corollary 5 of [77])**.** *With efficient computation and space management, the cutting plane method solves the problem (20) within*

$$\tilde{O}\left(mn + |\mathcal{D}|^2|\mathcal{Y}|^2 n \cdot \log(\tfrac{R}{\epsilon})\right) \tag{21}$$

flops and $O(n|\mathcal{D}||\mathcal{Y}|)$ space. Here we use $\tilde{O}(\cdot)$ to hide m, n, $|\mathcal{D}|$, and $|\mathcal{Y}|$ in the logarithm function.

The above result is essentially Corollary 5 of [77] by slightly extending the proof to the general case $m \neq n$. Finally, in terms of the implementation, we follow [77] and use the analytic center cutting plane method.

B Theoretical Proofs

This section includes the theoretical proofs for Section 2 and Section 5. We first show the Wasserstein distance upper bounds downstream disparity for MLP networks (Proposition 2.1). We then show (i) the optimization problem can be reduced to optimizing over the $\hat{X}$ rather than $\hat{Z}$ (Lemma 3.1), (ii) the surrogate function is convex and a valid upper bound of the optimization objective (Lemma 5.2), (iii) our proposed algorithm converges to a first-order stationary point in $\hat{X}$ (Theorem 5.3), and (iv) our proposed algorithm terminates in a finite amount of iterations (Theorem 5.4). We also prove the generalization bound for `FWC` performance in terms of Wasserstein distance and demographic parity to unseen datasets coming from the same (unknown) distribution the original dataset was sampled from. Finally, we include Section B.1 to note how the downstream learning using `FWC` can be broken down into two terms, which highlights the challenges in analyzing its theoretical properties.

Lemma B.1. *Let $Z = (X, Y), \hat{Z} = (\hat{X}, \hat{Y}) \in \mathcal{Z} = (\mathcal{X} \times \mathcal{Y})$ be two pairs of random variables with joint distributions p_Z and $p_{\hat{Z}}$ and marginal distributions p_X, p_Y and $p_{\hat{X}}$ and $p_{\hat{Y}}$ respectively. Let $\Pi(Z, \hat{Z})$ indicate the set of all joint probability distributions over the product space $\mathcal{Z} \times \mathcal{Z}$ that admit marginal and conditional distributions over $\mathcal{X}$ and $\mathcal{Y}$. For any non-negative cost operator c:*

$$\mathcal{W}_c(p_X, p_{\hat{X}}) \leq \mathcal{W}_c(p_Z, p_{\hat{Z}}).$$

Proof. Proof of Lemma B.1 Let $\Pi_X(X, \hat{X})$ indicate the set of all marginal (joint) probability distributions over the product space $\mathcal{X} \times \mathcal{X}$. For any $\pi \in \Pi(Z, \hat{Z})$ and the corresponding $\pi_x \in \Pi_X(X, \hat{X})$ we have that:

$$\int_{\mathcal{Z}\times\mathcal{Z}} c(z,\hat{z})d\pi(z,\hat{z}) \geq \int_{\mathcal{Z}\times\mathcal{Z}} c(x,\hat{x})d\pi(z,\hat{z}) \tag{22}$$
$$= \int_{\mathcal{X}\times\mathcal{X}} c(x,\hat{x})d\pi_x(x,\hat{x})$$
$$\geq \min_{\pi_x\in\Pi_x(x,\hat{x})} c(x,\hat{x})d\pi_x(x,\hat{x}) = \mathcal{W}_c(p_X, p_{\hat{X}})$$

As this is valid for any $\pi \in \Pi(Z,\hat{Z})$, select:

$$\pi^\star = \arg\min_{\pi\in\Pi(Z,\hat{Z})} \int_{\mathcal{Z}\times\mathcal{Z}} c(z,\hat{z})d\pi(z,\hat{z}).$$

Then the left-hand-side of (22) is equal to $\mathcal{W}_c(p_Z, p_{\hat{Z}})$, hence proving that $\mathcal{W}_c(p_Z, p_{\hat{Z}}) \geq \mathcal{W}_c(p_X, p_{\hat{X}})$.

□

Proof. Proof of Proposition 2.1 The proof of the upper bound follows from the first part of the proof of the Kantorovich-Rubinstein duality [66]. In this work we follow the proof by [6, 71], which consider the Lagrangian form of the 1-Wasserstein distance and express it in the following form:

$$\mathcal{W}(p_{(X,D)}, p_{(\hat{X},\hat{D})}) = \sup_{f,g:f(x)+g(y)\leq\|x-y\|_2} \left|\mathbb{E}_{(x,d)\sim p_{(X,D)}} f_\theta(x,d) - \mathbb{E}_{(x,d)\sim p_{(\hat{X},\hat{D})}} f_\theta(x,d)\right|,$$

where $f, g : \mathcal{X} \to \mathbb{R}$ are bounded, measurable functional Lagrangian multipliers. Let L_{f_θ} be the Lipschitz constant of the MLP f_θ, and define the following function:

$$h(x,d) = \frac{f_\theta(x,d)}{L_{f_\theta}}.$$

By definition, $h(x,d)$ is 1-Lipschitz. Again following [6, 71], we know that for 1-Lipschitz functions the following holds:

$$\left|\mathbb{E}_{(x,d)\sim p_{(X,D)}} f_\theta(x,d) - \mathbb{E}_{(x,d)\sim p_{(\hat{X},\hat{D})}} f_\theta(x,d)\right| = L_{f_\theta}\left|\mathbb{E}_{(x,d)\sim p_{(X,D)}} h(x,d) - \mathbb{E}_{(x,d)\sim p_{(\hat{X},\hat{D})}} h(x,d)\right|$$
$$= L_{f_\theta}\left|\int_{(\mathcal{X}\times\mathcal{D})\times(\mathcal{X}\times\mathcal{D})} h(x_1,d_1,x_2,d_2)d\pi\left(p_{(X,D)}, p_{(\hat{X},\hat{D})}\right)\right|$$
$$\leq L_{f_\theta}\int_{(\mathcal{X}\times\mathcal{D})\times(\mathcal{X}\times\mathcal{D})} |h(x_1,d_1,x_2,d_2)|\, d\pi\left(p_{(X,D)}, p_{(\hat{X},\hat{D})}\right)$$
$$\leq L_{f_\theta}\int_{(\mathcal{X}\times\mathcal{D})\times(\mathcal{X}\times\mathcal{D})} \|(x_1,d_1)-(x_2,d_2)\|_2\, d\pi\left(p_{(X,D)}, p_{(\hat{X},\hat{D})}\right)$$
$$\leq L_{f_\theta}\mathcal{W}_1(p_{(X,D)}, p_{(\hat{X},\hat{D})}) \leq L_{f_\theta}\mathcal{W}_1(p_Z, p_{\hat{Z}}),$$

where the last inequality is due to Lemma B.1. The result can be obtained by using the upper bound in [75, Section 6.1], which shows that $L_{f_\theta} \leq L_k$ for K-layer MLPs with ReLu activations.

□

Proof of Lemma 3.1. Once we generate $m|\mathcal{D}||\mathcal{Y}|$ data points, the feasible set of the latter Wasserstein coreset contains the feasible set of the former Wasserstein coreset. □

Proof of Lemma 5.2. The convexity follows directly from the convexity of $C(\hat{X})$, as the $P_k^\star \geq 0$ in (10).

Before proving it is an upper bound, we show some important properties of $F(C)$ as a function of C. Firstly, $F(C)$ is concave on C because of the concavity of the minimum LP's optimal objective on the objective vector. Secondly, since the feasible set of problem (8) is bounded, the optimal solution $F(C)$ is continuous with respect to C. Thirdly, due to the sensitivity analysis of LP [8], a supergradient of $F(C)$ at point C is the corresponding optimal solution $P^\star$. Here the definition of supergradients for concave functions is analogous to the definition of subgradients for convex functions.

Now we prove $g(\hat{X}; \hat{X}^k)$ is an upper bound of $F(C(\hat{X}))$. Because P_k^k is a supergradient of $F(C)$ when $C = C(\hat{X}^k)$,

$$F(C) + \langle P_k^k, C(\hat{X}) - C \rangle \geq F(C(\hat{X}))\ ,$$

in which the left-hand side is equal to $g(\hat{X}; \hat{X}^k)$ because $F(C) = \langle P_k^k, C \rangle$ and $g(\hat{X}; \hat{X}^k) = \langle C(\hat{X}), P_k^\star \rangle$. Therefore, the surrogate function is an upper bound of the objective function $F(C(\hat{X}))$, i.e., $g(\hat{X}; \hat{X}^k) \geq F(C(\hat{X}))$. Moreover, due to the definition in (10), $g(\hat{X}; \hat{X}^k) = F(C(\hat{X}))$ when $\hat{X} = \hat{X}^k$. □

Proof of Theorem 5.3. The monotonically decreasing part of the claim follows by:

$$F(C(\hat{X}^{k+1})) \leq g(\hat{X}^{k+1}; \hat{X}^k) = \arg\min_{\hat{X} \in \mathcal{X}^m} g(\hat{X}; \hat{X}^k) \leq g(\hat{X}^k; \hat{X}^k) = F(C(\hat{X}^k))\ . \tag{23}$$

Here the first inequality is due to the fact that $g(\hat{X}; \hat{X}^k) \geq F(C(\hat{X}))$ for any $\hat{X}$. The final equality is because $g(\hat{X}; \hat{X}^k) = F(C(\hat{X}))$ when $\hat{X} = \hat{X}^k$. Once $g(\hat{X}^k; \hat{X}^k) = g(\hat{X}^{k+1}; \hat{X}^k)$ and thus $\hat{X}^k \in \arg\min_{\hat{X} \in \mathcal{X}^m} g(\hat{X}; \hat{X}^k)$, then $\hat{X}^k$ is a global minimizer of the convex upper bound $g(\cdot; \hat{X}^k)$ for $F(C(\cdot))$ and the upper bound $g(\hat{X}^k; \hat{X}^k)$ attains the same function value with $F(C(\hat{X}^k))$. Therefore, if the surrogate function is smooth at $\hat{X}^k$, which could be achieved if $C(\hat{X}^k)$ is smooth at $\hat{X}^k$, then X^k is a first-order stationary point of (9). □

Proof of Theorem 5.4. Because (11) has a unique minimizer, the second inequality in (23) holds strictly when $\hat{X}^{k+1} \neq \hat{X}^k$, or equivalently $g(\hat{X}^{k+1}; \hat{X}^k) \neq g(\hat{X}^k; \hat{X}^k)$. Once $\hat{X}^{k+1} = \hat{X}^k$, then the algorithm terminates. Note that there are only finite possible optimal basic feasible solution $P^\star$ that could be generated by `FairWASP`, as shown in Lemma A.1. However, before the majority minimization converges, (23) holds strictly and the corresponding $P_k^\star$ keeps changing. Therefore, after finite iterations, there must be a $P_t^\star$ equal to a previous $P_j^\star$ for $j < t$. When that happens, because the surrogate functions are the same and thus have the same minimizer, $\hat{X}^{t+1} = \hat{X}^{j+1}$, and the inequalities (23) then hold at equality when $k = j, j+1, \ldots, t$. This implies that $g(\hat{X}^{j+1}; \hat{X}^j) = g(\hat{X}^j; \hat{X}^j)$, so the algorithm terminates within finite iterations. □

Proof of Proposition 5.5. For determining the convergence in Wasserstein distance between $p_{\hat{Z}}$ and q_Z, we first use the triangle inequality:

$$\mathcal{W}_c(p_{\hat{Z}}, q_Z) \leq \mathcal{W}_c(p_{\hat{Z}}, p_Z) + \mathcal{W}_c(p_Z, q_Z) = \lambda + \mathcal{W}_c(p_Z, q_Z)$$

Note that the first term is deterministic, as it is the result of the optimization problem (4). For the second term, we use the result from [22], which implies that with probability $1 - \alpha$ and for the 1-Wasserstein distance:

$$\mathbb{P}(\mathcal{W}_c(p_Z, q_Z) > \xi) \leq \exp\left(-cn\xi^{1/d}\right),$$

and so by setting the right-hand side equal to α, or equivalently, setting

$$\xi = \sqrt[d]{\frac{c \log(1/\alpha)}{n}},$$

we obtain the first result.

For determining the convergence of the disparity between $p_{\hat{Z}}$ and $p_{\hat{Z}}(y, d)$ and $q_Y(y)$, we again first use the triangle inequality from the definition of the disparity J:

$$\begin{aligned}\sup_{y\in\mathcal{Y},d\in\mathcal{D}} J(p_{\hat{Z}}(y|d),q_Y(y)) &= \sup_{y\in\mathcal{Y},d\in\mathcal{D}} \frac{|p_{\hat{Z}}(y|d)-q_Y(y)|}{q_Y(y)} \\ &\leq \sup_{y\in\mathcal{Y},d\in\mathcal{D}} \left(\frac{|p_{\hat{Z}}(y|d)-p_Y(y)|}{q_Y(y)} + \frac{|p_Y(y)-q_Y(y)|}{q_Y(y)}\right) \\ &\leq \sup_{y\in\mathcal{Y},d\in\mathcal{D}} \frac{|p_{\hat{Z}}(y|d)-p_Y(y)|}{q_Y(y)} + \sup_{y\in\mathcal{Y}} \frac{|p_Y(y)-q_Y(y)|}{q_Y(y)}\end{aligned}$$

As the minimum of the marginal distribution of $q_Y(y)$ is bounded away from zero $\min_{y\in\mathcal{Y}} q_Z(y) = \rho > 0$, and since by the optimization problem (4) we have $J(p_{\hat{Z}}(y|d),p_Y(y)) \leq \epsilon$ for all $y \in \mathcal{Y}, d \in \mathcal{D}$:

$$\begin{aligned}\sup_{y\in\mathcal{Y},d\in\mathcal{D}} J(p_{\hat{Z}}(y|d),q_Y(y)) &\leq \sup_{y\in\mathcal{Y},d\in\mathcal{D}} \frac{|p_{\hat{Z}}(y|d)-p_Y(y)|}{p_Y(y)}\frac{p_Y(y)}{q_Y(y)} + \sup_{y\in\mathcal{Y}} \frac{|p_Y(y)-q_Y(y)|}{q_Y(y)} \\ &\leq \sup_{y\in\mathcal{Y},d\in\mathcal{D}} J(p_{\hat{Z}}(y|d),p_Y(y))\frac{1}{\rho} + \sup_{y\in\mathcal{Y}} \frac{|p_Y(y)-q_Y(y)|}{q_Y(y)} \\ &\leq \frac{\epsilon}{\rho} + \sup_{y\in\mathcal{Y}} \frac{|p_Y(y)-q_Y(y)|}{q_Y(y)}.\end{aligned}$$

Note that the first term is deterministic, while the second one is not, as we need to account for the uncertainty of observing n i.i.d. samples $\{Z_i\}_{i=1}^n$. For the second part, we use the Dvoretzky–Kiefer–Wolfowitz (DKW, [17]) inequality:

$$\begin{aligned}\mathbb{P}\left(\sup_{y\in\mathcal{Y}} \frac{|p_Y(y)-q_Y(y)|}{q_Y(y)} > \xi\right) &\leq \mathbb{P}\left(\sup_{y\in\mathcal{Y}} |p_Y(y)-q_Y(y)| > \xi\rho\right) \\ &\leq 2\exp(-2n\rho^2\xi^2)\end{aligned}$$

By setting the right-hand side equal to α, or equivalently, setting

$$\xi = \sqrt{\frac{\log(\frac{2}{\alpha})}{2n\rho^2}},$$

we obtain the second result.

Finally, we note that the assumption on the marginal distribution of q_Y is bounded away from zero, i.e., $\rho > 0$, is reasonable as the outcome is a discrete (usually binary) random variable. This assumption would be much more restricting in case y was a continuous random variable (e.g., in regression settings). □

B.1 Downstream Learning using `FWC`

Overall, the hardness of deriving bounds for the synthetic representatives provided by `FWC` can be analyzed using the following breakdown, which is adapted from [78]. Consider the given dataset $Z = \{(X_i, Y_i, D_i)\}_{i=1}^n$, the synthetic representatives obtained using `FWC` $\hat{Z} = \{(\hat{X}_j, \hat{Y}_j, \hat{D}_j)\}_{j=1}^m$ and $h \in \mathcal{H} = L^2(\mathcal{X}\times\mathcal{Y}\times\mathcal{D})$, the set of measurable square-integrable function in L^2. If we consider the downstream learning process using `FWC` samples over the L^2 space:

$$\inf_{h\in\mathcal{H}} \mathbb{E}_{Y|X,D}\left[\|Y - h(\hat{X},\hat{D})\|_2^2\right],$$

then the above can be expanded in two terms, due to the property of the conditional expectation being an orthogonal operator in $\mathcal{H}$:

$$\inf_{h\in\mathcal{H}} \mathbb{E}_{Y|X,D}\left[\|Y - h(\hat{X},\hat{D})\|_2^2\right] =$$
$$= \underbrace{\mathbb{E}_{Y|X,D}\left[\|Y - \mathbb{E}_{X,D}[\hat{Y}|\hat{X},\hat{D}]\|_2^2\right]}_{\texttt{FWC Approximation Error}} + \underbrace{\inf_{h\in\mathcal{H}} \mathbb{E}_{Y|X,D}\left[\|\mathbb{E}_{X,D}[\hat{Y}|\hat{X},\hat{D}] - h(\hat{X},\hat{D})\|_2^2\right]}_{\text{Learning with FWC Samples}} \quad (24)$$

The first term corresponds to the loss of information in approximating Y with $\hat{Y}$ via the FWC approach, and is actually independent of any downstream learning. This condition requires for the first moment (which for the binary Y case is equivalent to the joint distribution) of Y and $\hat{Y}$ to be as close as possible. Using FWC this is enforced by minimizing the Wasserstein distance. Indeed, if in definition (1) one restricts to couplings that admit marginal and conditional distributions, then the conditional distributions of $Y|X,D$ and $\hat{Y}|\hat{X},\hat{D}$ are upper bounded in Wasserstein sense by the Wasserstein distance between the joint distribution of p_Z and $p_{\hat{Z}}$ [37].

The second terms refers to the training process using FWC samples Firstly, by using the equivalence in [78], the second term is equivalent to $\inf_{h\in\mathcal{H}} \mathbb{E}_{Y|X,D}\left[\|\hat{Y} - f(\hat{X},\hat{D}\|_2^2\right]$, which correspond the finding the best L^2 function to approximate the distribution of $\hat{Y}$. This fact implies that using FWC samples is indeed mathematically equivalent to the learning task for the original Y. However, this second term also highlights the hardness of developing learning bounds, as the FWC synthetic representatives $\hat{Z} = \{(\hat{X}_j, \hat{Y}_j, \hat{D}_j)\}_{j=1}^m$ are not i.i.d., and hence standard bounds are not applicable.

C Experiment Details

C.1 Runtime Analysis on Synthetic Dataset

As mentioned in Section 7, we generate a synthetic dataset in which one feature is strongly correlated with the protected attribute D to induce a backdoor dependency on the outcome. We consider a binary protected attribute, $D \in \{0,1\}$, which could indicate e.g., gender or race. The synthetic dataset contains two features, a feature X_1 correlated with the protected attribute and a feature X_2 uncorrelated with the protected attribute. For $D = 0$, X_1 is uniformly distributed in $[0, 10]$, while for $D = 1$, $X_1 = 0$. Instead, X_2 is 5 times a random variable from a normal distribution $\mathcal{N}(0,1)$. Finally, the outcome Y is binary, so $Y = \{0,1\}$: $Y_i = 1$ when $Y_i > m_x + \epsilon_i$ and $Y_i = 0$ when $Y_i \leq m_x + \epsilon_i$, where m_x is the mean of $\{(X_1)_i + (X_2)_i\}_i$ and the noise ϵ_i comes from a normal distribution $\mathcal{N}(0,1)$.

This experiment visualizes the speed of our method with respect to different numbers of overall samples n, number of samples in the compressed dataset m, and the dimensionality of features p. We evaluated the performance of the algorithm under the synthetic data with different configurations of n, p, and m. In this experiment, we set compute the fair Wasserstein coreset under the l_1-norm distance and we use k-means [42] to initialize the starting coreset $\hat{X}^0$. We terminate the algorithm when $\hat{X}^k = \hat{X}^{k-1}$. The time per iteration and total iterations for varying n, p, and m are shown in the Figures 1 (top left) and 2. We see that increasing the sample size of the original dataset n increases the runtime and number of iterations, while increasing the number of coresets m or dimensionality of the features p reduces the overall numbers of iterations but increases each iteration's runtime. Additionally, for the setting of Figure 1 (top left), in which we vary the dataset size n, Table 2 provides FWC average runtimes from $n = 500$ to $n = 1,000,000$. We compare FWC runtimes with the runtime at $n = 500$ (our lowest dataset size in the experiment) extrapolated (i) linearly, with a factor of 1, (ii) linearly, with a factor of 10 and (iii) quadratically. We can see that the complexity is near linear and less than quadratic with respect to the dataset size n, although the rate indeed seem to increase for n at $500,000$ and above, which can be attributed to the increasing number of iterations required to achieve convergence. This is akin to the phenomenon well-known for k-means, for which in larger datasets k-means might take an exponentially large number of iterations to terminate [73].

In practice, a fixed number of overall iterations is set to avoid this case: *sklearn* sets it to 300[6], *feiss* to 25[7] and *Matlab* to 100[8].

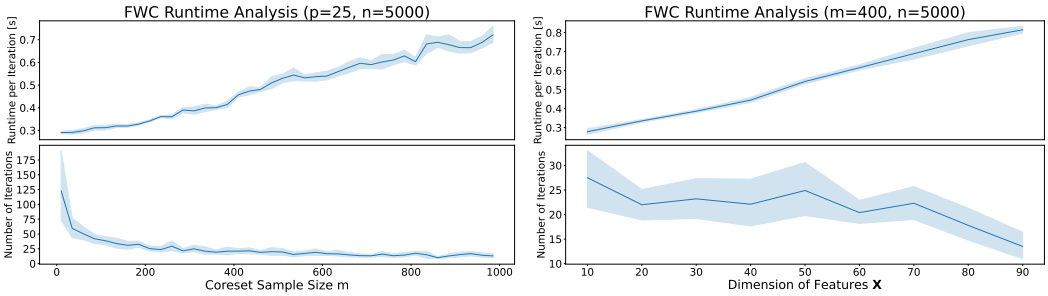

Figure 2: Runtime analysis of `FWC` when varying the size of the coreset m (left) and the dimensionality of the features p (right). We report averages and one standard deviation computed over 10 runs.

Table 2: Average runtimes for `FWC` in the same settings as Figures 1 (top left), varying the dataset size n while fixing $m = 250$ and $p = 25$, compared with the runtimes for the smallest dataset extrapolated (i) linearly, with a factor of 1, (ii) linearly, with a factor of 10 and (iii) quadratically. `FWC` enjoys a near linear time complexity, increasing with the largest dataset sizes; this phenomenon is shared with other clustering algorithm such as k-means (see text).

Dataset size n	**Runtime [seconds]**	**Linear (Factor** 1**)**	**Actual / Linear (Factor** 1**)**	**Linear (Factor** 10**)**	**Actual / Linear (Factor** 10**)**	**Quadratic**	**Actual / Quadratic**
500	8.9e-01	-	-	-	-	-	-
1,000	9.4e-01	1.8e+00	0.52	1.8e+01	0.05	3.6e+00	0.26
2,500	2.2e+00	4.5e+00	0.49	4.5e+01	0.05	2.2e+01	0.10
5,000	1.0e+01	8.9e+00	1.17	8.9e+01	0.12	8.9e+01	0.12
10,500	1.8e+01	1.9e+01	0.95	1.9e+02	0.09	3.9e+02	0.05
24,500	8.7e+01	4.4e+01	1.99	4.4e+02	0.20	2.1e+03	0.04
48,500	3.0e+02	8.6e+01	3.46	8.6e+02	0.35	8.4e+03	0.04
75,000	7.0e+02	1.3e+02	5.25	1.3e+03	0.52	2.0e+04	0.03
100,000	1.0e+03	1.8e+02	5.79	1.8e+03	0.58	3.6e+04	0.03
250,000	3.8e+03	4.5e+02	8.51	4.5e+03	0.85	2.2e+05	0.02
500,000	1.6e+04	8.9e+02	18.34	8.9e+03	1.83	8.9e+05	0.02
750,000	2.9e+04	1.3e+03	21.64	1.3e+04	2.16	2.0e+06	0.01
1,000,000	4.9e+04	1.8e+03	27.58	1.8e+04	2.76	3.6e+06	0.01

C.2 Real Datasets

We consider the following four real datasets widely used in the fairness literature [19]:

- the *Adult dataset* [7], which reports demographic data from the 1994 US demographic survey about $\sim 49,000$ individuals. We use all the available features for classification apart from the "`fnlwgt`" feature, including gender as the protected attribute D and whether the individual salary is more than USD$50,000$;
- the *Drug dataset* [20], which contains drug usage history for $1,885$ individuals. Features X include the individual age, country of origin, education and scores on various psychological test. We use the individual gender as the protected variable D. The response Y is based on whether the individual has reported to have ever used the drug "cannabis" or not;
- the *Communities and Crime dataset* [64] was put together towards the creation of a software tool for the US police department. The dataset contains socio-economic factors for $\sim 2,000$ communities in the US, along with the proportion of violent crimes in each community. As protected attribute D, we include whether the percentage of the black population in the community is above the overall median. For the response Y, we use a binary indicator of whether the violent crimes percentage level is above the mean across all communities in the dataset;

[6] `https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html`
[7] `https://faiss.ai/cpp_api/struct/structfaiss_1_1Clustering.html`
[8] `https://www.mathworks.com/help/stats/kmeans.html`

- the *German Credit dataset* [28] reports a set of $1,000$ algorithmic credit decisions from a regional bank in Germany. We use all the available features, including gender as protected attribute D and whether the credit was approved as response Y.

As perfect demographic parity achieves a value of 0 for the discrepancy J, so we have included demographic "dis"-parity to indicate any deviation from demographic parity. Across all experiments we compute demographic parity as the following absolute difference:

$$DD \stackrel{\text{def.}}{=} \left| p\left(h(X,D)=1|D=1\right) - p(h(X,D)=1|D=0)\right|, \tag{25}$$

for a given classifier h and a protected attribute D with two levels; the larger this difference, the larger the disparity. We also include the implementation and hyper-parameters of the methods used in the fairness-utility tradeoffs throughout the experiments in this paper:

- For `FWC` we set the fairness violation hyper-parameter ϵ of problem in Equation (5) to be $\epsilon = [0.01, 0.05, 0.1]$, hence obtaining three separate `FWC` models, `FWC` (0.01), `FWC` (0.05) and `FWC` (0.1);
- For `Fairlet` [4], we use the implementation available at the following GitHub repository: `https://github.com/talwagner/fair_clustering/tree/master`
- For `IndFair` [12] and `K-Median Coresets` [3], we use the implementation available at the following GitHub repository: `https://github.com/jayeshchoudhari/CoresetIndividualFairness/tree/master`
- For k-means [42] and k-medoids [45, 58] we use the implementations available in the Python package `Scikit-Learn` [59]

All computations are run on an Ubuntu machine with 32GB of RAM and 2.50GHz Intel(R) Xeon(R) Platinum 8259CL CPU. For all datasets, we randomly split 75% of the data into training/test set, and change the split during each separate run; the training data are further separated into training and validation with $90/10$ to compute early stopping criteria during training. The downstream classifier used is a one-layer deep multi-layer perceptron (MLP) with 20 hidden layers, ReLu activation function in the hidden layer and softmax activation function in the final layer. Unless stated otherwise, `FWC` uses the L^1 to compute the distance from the original datasets in the optimization problem. For the downstream classifier, we use Adam optimizer [38] with a learning rate set to 10^{-3}, a batch size of 32, a maximum number of epochs set to 500 with early stopping evaluated on the separate validation set with a patience of 10 epochs and both the features X and the protected attribute D are used for training the classifier. Note that due to the size of the Adult dataset, `Fairlet` coresets [4] could not be run due to the RAM memory required exceeding the machine capacity (32GB). Finally, all uncertainties are reported at $\pm 1\sigma$ (one standard deviation) in both figures and tables. Uncertainties are computed over a set of 10 runs where the random seed for the algorithm initialization and train/test split was changed, but consistent across all methods (i.e., all methods in the first run were presented the same train/test split across each datasets).

Table 3: Wasserstein distance of the weighted coresets with respect to the original dataset, with averages and standard deviations obtained over 10 runs. In bold, coresets with the closest distance to the original dataset (i.e., smallest Wasserstein distance) in each coreset size and dataset combination.

	Wasserstein Distance (↓)											
Method	*Adult* ($\times 10^6$)			*Credit* ($\times 10^6$)			*Crime*			*Drug*		
Coreset Size m	0.5%	1%	2%	5%	10%	20%	5%	10%	20%	5%	10%	20%
FWC (ϵ: 0.01)	5.72 ± 0.95	$\mathbf{3.76 \pm 0.80}$	$\mathbf{2.70 \pm 0.56}$	$\mathbf{0.40 \pm 0.10}$	0.75 ± 0.21	1.07 ± 0.28	$\mathbf{1.65 \pm 0.08}$	1.76 ± 0.09	1.89 ± 0.08	$\mathbf{3.23 \pm 0.23}$	3.57 ± 0.24	3.98 ± 0.22
FWC (ϵ: 0.05)	$\mathbf{5.57 \pm 0.73}$	3.90 ± 0.82	2.79 ± 0.58	1.07 ± 0.28	0.48 ± 0.24	0.72 ± 0.17	1.87 ± 0.11	$\mathbf{1.66 \pm 0.12}$	1.72 ± 0.06	3.95 ± 0.30	$\mathbf{3.31 \pm 0.34}$	3.49 ± 0.19
FWC (ϵ: 0.1)	6.18 ± 1.07	4.15 ± 0.82	3.04 ± 0.63	0.71 ± 0.15	1.05 ± 0.30	0.47 ± 0.19	1.70 ± 0.07	1.85 ± 0.11	$\mathbf{1.62 \pm 0.08}$	3.50 ± 0.21	3.95 ± 0.29	$\mathbf{3.25 \pm 0.24}$
Fairlet (K-Means)	-	-	-	10.02 ± 2.14	7.47 ± 2.77	5.00 ± 2.48	2.10 ± 0.42	2.10 ± 0.44	2.00 ± 0.36	4.29 ± 0.44	4.10 ± 0.45	3.83 ± 0.43
Fairlet (K-Medoids)	-	-	-	3.16 ± 0.81	4.48 ± 0.89	5.47 ± 0.65	4.21 ± 0.17	4.14 ± 0.14	4.10 ± 0.06	6.50 ± 0.45	5.90 ± 0.47	5.17 ± 0.43
Ind. Fair. Cor.	12.96 ± 5.83	23.91 ± 25.39	11.48 ± 8.19	$\mathbf{0.41 \pm 0.29}$	0.72 ± 0.46	$\mathbf{0.23 \pm 0.12}$	2.35 ± 0.16	2.60 ± 0.25	2.01 ± 0.09	4.84 ± 0.33	5.51 ± 0.50	4.17 ± 0.21
K-Median Cor.	22.11 ± 17.42	9.37 ± 11.27	12.04 ± 9.11	0.66 ± 0.74	$\mathbf{0.26 \pm 0.17}$	0.36 ± 0.17	2.58 ± 0.20	2.01 ± 0.11	2.35 ± 0.12	5.38 ± 0.54	4.16 ± 0.22	4.80 ± 0.15
Uniform Subsampling	18.01 ± 18.34	14.39 ± 14.90	7.96 ± 5.42	0.74 ± 0.53	0.30 ± 0.16	0.80 ± 1.15	2.60 ± 0.26	1.95 ± 0.13	2.28 ± 0.24	5.46 ± 0.59	4.01 ± 0.30	4.75 ± 0.42
K-Means	43.96 ± 18.82	77.04 ± 18.06	74.50 ± 12.53	13.23 ± 1.42	10.76 ± 1.55	7.59 ± 1.58	2.00 ± 0.12	1.93 ± 0.11	2.04 ± 0.08	3.94 ± 0.17	3.66 ± 0.20	3.62 ± 0.32
K-Medoids	50.87 ± 4.30	53.31 ± 1.85	51.77 ± 1.86	2.59 ± 0.42	4.25 ± 0.68	5.18 ± 0.07	3.15 ± 0.08	3.12 ± 0.11	2.83 ± 0.06	5.14 ± 0.08	4.53 ± 0.28	3.91 ± 0.09

Closeness to the original dataset and clustering performance Tables 3 and 4 include the numerical values (means and standard deviations computed over 10 runs) for all methods in

Table 4: Clustering cost of the coresets with respect to the original dataset, with averages and standard deviations obtained over 10 runs. In bold, coresets with the smallest clustering cost (i.e., smallest sum of square distances of original dataset samples from the closest generated coreset sample) in each coreset size and dataset combination.

	Clustering Cost (↓)											
Method	*Adult* ($\times 10^6$)			*Credit* ($\times 10^4$)			*Crime* ($\times 10^2$)			*Drug* ($\times 10^2$)		
Coreset Size m	0.5%	1%	2%	5%	10%	20%	5%	10%	20%	5%	10%	20%
FWC (ϵ: 0.01)	14.73 ± 6.43	5.44 ± 2.29	2.07 ± 0.39	**1.07 ± 0.21**	2.61 ± 0.59	4.31 ± 0.92	**3.86 ± 0.27**	4.44 ± 0.21	4.79 ± 0.19	**5.54 ± 0.41**	6.45 ± 0.32	7.02 ± 0.31
FWC (ϵ: 0.05)	13.59 ± 4.21	5.81 ± 2.37	2.26 ± 0.30	4.41 ± 1.52	1.39 ± 0.96	2.44 ± 0.21	4.72 ± 0.36	**3.96 ± 0.41**	4.37 ± 0.14	6.94 ± 0.54	**5.70 ± 0.63**	6.35 ± 0.24
FWC (ϵ: 0.1)	20.29 ± 12.27	6.31 ± 2.36	2.19 ± 0.29	2.48 ± 0.20	4.00 ± 1.28	1.22 ± 0.41	4.36 ± 0.15	4.72 ± 0.34	3.89 ± 0.30	6.36 ± 0.26	6.94 ± 0.51	5.62 ± 0.45
Fairlet (K-Means)	-	-	-	1.36 ± 0.09	0.76 ± 0.11	0.60 ± 0.31	4.76 ± 0.46	4.36 ± 0.52	3.92 ± 0.56	7.08 ± 0.30	6.45 ± 0.40	5.76 ± 0.63
Fairlet (K-Medoids)	-	-	-	5.79 ± 1.36	5.40 ± 1.08	5.04 ± 0.92	6.27 ± 0.28	6.00 ± 0.27	5.59 ± 0.20	8.58 ± 0.41	7.93 ± 0.42	7.09 ± 0.54
Ind. Fair. Cor.	1.23 ± 0.37	4.67 ± 9.46	0.70 ± 0.04	1.50 ± 0.38	2.46 ± 0.77	0.74 ± 0.10	5.24 ± 0.27	5.64 ± 0.48	4.53 ± 0.22	7.29 ± 0.35	7.92 ± 0.62	6.30 ± 0.41
K-Median Cor.	10.45 ± 14.21	0.70 ± 0.04	1.06 ± 0.19	2.47 ± 0.90	0.76 ± 0.06	1.37 ± 0.29	5.64 ± 0.39	4.55 ± 0.27	5.21 ± 0.16	7.75 ± 0.66	6.26 ± 0.36	7.26 ± 0.20
Uniform Subsampling	10.16 ± 15.13	3.49 ± 1.18	1.93 ± 0.67	3.38 ± 1.42	1.09 ± 0.26	2.74 ± 2.71	5.61 ± 0.45	4.46 ± 0.34	5.07 ± 0.37	7.74 ± 0.73	6.06 ± 0.51	7.02 ± 0.50
K-Means	**0.72 ± 0.01**	**0.58 ± 0.01**	**0.48 ± 0.01**	1.28 ± 0.21	**0.68 ± 0.12**	**0.51 ± 0.29**	4.52 ± 0.16	4.09 ± 0.25	**3.76 ± 0.39**	6.86 ± 0.28	6.13 ± 0.41	**5.61 ± 0.67**
K-Medoids	83.53 ± 13.86	75.10 ± 7.22	56.95 ± 9.17	4.93 ± 0.53	4.70 ± 0.48	4.19 ± 0.13	5.52 ± 0.07	5.15 ± 0.28	4.49 ± 0.25	7.65 ± 0.12	6.83 ± 0.43	5.93 ± 0.30

terms of Wasserstein distance from the original dataset and clustering cost, for the three coreset sizes $m = [5\%, 10\%, 20\%]$ (apart from the Adult dataset, in which coreset sizes are set to $m = [0.5\%, 1\%, 2\%]$ due to the large size of the original dataset). Clustering cost is computed as the sum of the squared distance of each point in the original dataset from the closest coreset representative, while the Wasserstein distance is computed solving the optimal transport between the empirical distribution of the original dataset and the one of the coresets, using the L^1 norm as cost function. FWC consistently provides the closest distributional distance to the original dataset in Wasserstein distance, with the only exception of the Credit dataset, in which the large number of discrete features makes the optimization non-smooth in feature space, resulting in a potentially imprecise solution of Equation (9). In addition, FWC, while not naturally minimizing clustering costs, seems to achieve competitive clustering costs in smaller datasets while not performing as well on larger datasets such as Adult. Finally, we note that the Wasserstein distance might not always decrease with a higher coreset size, which is due to the parity violation constraint ϵ. In other words, the coreset samples not only have to be close to the original dataset distribution but also respect the hard fairness constraint; the lower the ϵ, the tighter this constraint is (Equation (4)). Indeed, when ϵ is the largest ($\epsilon = 0.1$), coresets of size 20% (or 2% for the Adult dataset, i.e., the largest) consistently has a smaller Wasserstein distance to the original dataset than coresets of size 5% (0.5% for the Adult dataset, i.e., the smallest).

Fairness-utility tradeoff when using coresets for training downstream models Figure 6 expands the results provided in Figure 1 and shows all the fairness-utility tradeoffs for all methods across the four datasets, both with (right column) and without (left column) using a pre-processing fairness approach [34] (excluding FWC , to which no fairness modification is applied after coresets have been generated). For each method, the coreset size that achieves the best fairness-utility tradeoff is shown (which is not necessarily the coreset with the largest size). FWC achieves a competitive fairness-utility tradeoff with respect to other competing methods, when using the generated coresets to train a downstream MLP classifier model. FWC consistently reduces disparity in the downstream classification with respect to other approaches, and often maintains the same utility (indicated by the AUC). For completeness, Figure 7 also reports standard deviations for the fairness-utility tradeoff; standard deviations for the Adult and Credit datasets are large due to the MLP classifier becoming trivial (i.e., always returning 0s or 1s), which yields very low performance but has no demographic disparity (by definition, as all test samples are assigned the same outcome). Tables 5 and 6 report the numbers shown in Figures 6 and 7, including one number to quantify the fairness-utility tradeoff, computed as the Euclidean distance in the Figure from the $(0, 1)$ point (which would be a fair classifier with perfect performance). In other words, for the k-th method achieving a disparity of d_k with uncertainty Δd_k and performance a_k with uncertainty Δa_k, the tradeoff t_k and associated uncertainty Δt_k are quantified as:

$$t_k = \sqrt{(1 - a_k)^2 + d_k^2} \quad , \quad \Delta t_k = \sqrt{\left(\frac{d_k}{t_k}\Delta d_k\right)^2 + \left(\frac{a_k - 1}{t_k}\Delta a_k\right)^2} \tag{26}$$

Finally, the average reduction in disparity was computed from Tables 5 and 6, by averaging the improvement obtained by FWC samples against all methods and across datasets. FWC result in an

average reduction in disparity of 53% and 18% for the scenario without and with fairness pre-processing, respectively.

Table 5: Demographic disparity (Equation (25)), AUC and fairness-utility tradeoff (Equation (26)) of downstream MLP classifier trained using all fair coresets/clustering methods across the four real datasets. The best method across the 3 different coreset sizes is shown, and the best performing method for each metric in each dataset is bolded. Averages and standard deviations taken over 10 runs. For the Credit dataset, K-means reaches low disparities due to the classifier being trivial, i.e., returning the same prediction regardless of input features.

Method	*Adult Dataset*			*Credit Dataset*			*Crime Dataset*			*Drug Dataset*		
	DD (↓)	AUC (↑)	Tradeoff (↓)	DD (↓)	AUC (↑)	Tradeoff (↓)	DD (↓)	AUC (↑)	Tradeoff (↓)	DD (↓)	AUC (↑)	Tradeoff (↓)
FWC (ϵ: 0.01)	**0.02 ± 0.02**	0.67 ± 0.10	0.33 ± 0.10	0.03 ± 0.04	0.67 ± 0.11	0.33 ± 0.11	**0.13 ± 0.05**	0.83 ± 0.02	**0.22 ± 0.04**	**0.05 ± 0.04**	0.79 ± 0.02	**0.21 ± 0.02**
FWC (ϵ: 0.05)	0.07 ± 0.03	**0.75 ± 0.08**	**0.26 ± 0.07**	0.02 ± 0.02	0.64 ± 0.11	0.36 ± 0.11	0.15 ± 0.04	0.84 ± 0.02	**0.22 ± 0.03**	0.06 ± 0.02	0.80 ± 0.02	**0.21 ± 0.02**
FWC (ϵ: 0.1)	0.06 ± 0.03	0.70 ± 0.08	0.30 ± 0.08	0.03 ± 0.03	0.66 ± 0.12	0.34 ± 0.12	0.16 ± 0.03	0.85 ± 0.01	**0.22 ± 0.03**	0.08 ± 0.03	0.80 ± 0.01	**0.21 ± 0.02**
Fairlet (K-Means)	-	-	-	**0.01 ± 0.01**	0.57 ± 0.11	0.43 ± 0.11	0.38 ± 0.09	0.90 ± 0.02	0.39 ± 0.09	0.16 ± 0.06	0.80 ± 0.02	0.25 ± 0.04
Fairlet (K-Medoids)	-	-	-	0.04 ± 0.06	0.60 ± 0.09	0.41 ± 0.09	0.24 ± 0.16	0.86 ± 0.04	0.28 ± 0.14	0.10 ± 0.09	0.76 ± 0.05	0.26 ± 0.06
Ind. Fair. Cor.	0.08 ± 0.07	0.73 ± 0.08	0.28 ± 0.08	0.05 ± 0.05	0.65 ± 0.11	0.35 ± 0.11	0.39 ± 0.10	0.88 ± 0.02	0.40 ± 0.10	0.21 ± 0.06	**0.81 ± 0.02**	0.28 ± 0.05
K-Median Cor.	0.12 ± 0.08	0.72 ± 0.09	0.30 ± 0.09	0.07 ± 0.07	0.66 ± 0.12	0.35 ± 0.12	0.45 ± 0.05	**0.91 ± 0.01**	0.46 ± 0.05	0.16 ± 0.10	0.76 ± 0.06	0.29 ± 0.08
Uniform Subsampling	0.12 ± 0.06	0.71 ± 0.10	0.31 ± 0.10	0.07 ± 0.09	**0.69 ± 0.11**	**0.31 ± 0.11**	0.45 ± 0.08	0.89 ± 0.02	0.46 ± 0.08	0.14 ± 0.11	0.78 ± 0.03	0.26 ± 0.06
K-Means	0.08 ± 0.04	0.68 ± 0.10	0.33 ± 0.10	0.00 ± 0.00 *	0.51 ± 0.02	0.49 ± 0.02	0.40 ± 0.03	**0.91 ± 0.01**	0.41 ± 0.03	0.15 ± 0.04	**0.81 ± 0.02**	0.25 ± 0.03
K-Medoids	0.08 ± 0.04	0.71 ± 0.12	0.30 ± 0.11	0.05 ± 0.05	0.60 ± 0.07	0.40 ± 0.07	0.28 ± 0.10	0.88 ± 0.04	0.30 ± 0.09	0.12 ± 0.05	0.80 ± 0.02	0.23 ± 0.03

Table 6: Demographic disparity (Equation (25)), AUC and fairness-utility tradeoff (Equation (26)) of downstream MLP classifier trained using all fair coresets/clustering methods across the four real datasets. All methods apart from FWC have been corrected for fairness using a preprocessing fairness technique by [34]. The best method across the 3 different coreset sizes is shown, and the best performing method for each metric in each dataset is bolded. Averages and standard deviations taken over 10 runs.

Method	*Adult Dataset*			*Credit Dataset*			*Crime Dataset*			*Drug Dataset*		
	DD (↓)	AUC (↑)	Tradeoff (↓)	DD (↓)	AUC (↑)	Tradeoff (↓)	DD (↓)	AUC (↑)	Tradeoff (↓)	DD (↓)	AUC (↑)	Tradeoff (↓)
FWC (ϵ: 0.01)	**0.02 ± 0.02**	0.67 ± 0.10	0.33 ± 0.10	0.03 ± 0.04	0.67 ± 0.11	**0.33 ± 0.11**	0.13 ± 0.05	0.83 ± 0.02	0.22 ± 0.04	**0.05 ± 0.04**	0.79 ± 0.02	**0.21 ± 0.02**
FWC (ϵ: 0.05)	0.07 ± 0.03	**0.75 ± 0.08**	**0.26 ± 0.07**	0.02 ± 0.02	0.64 ± 0.11	0.36 ± 0.11	0.15 ± 0.04	0.84 ± 0.02	0.22 ± 0.03	0.06 ± 0.02	0.80 ± 0.02	**0.21 ± 0.02**
FWC (ϵ: 0.1)	0.06 ± 0.03	0.70 ± 0.08	0.30 ± 0.08	0.03 ± 0.03	0.66 ± 0.12	0.34 ± 0.12	0.16 ± 0.03	0.85 ± 0.01	0.22 ± 0.03	0.08 ± 0.03	0.80 ± 0.01	**0.21 ± 0.02**
Fairlet (K-Means)	-	-	-	0.02 ± 0.03	0.60 ± 0.08	0.40 ± 0.08	0.23 ± 0.05	0.87 ± 0.02	0.26 ± 0.04	0.12 ± 0.06	**0.82 ± 0.02**	**0.21 ± 0.04**
Fairlet (K-Medoids)	-	-	-	0.04 ± 0.05	0.59 ± 0.09	0.41 ± 0.09	**0.11 ± 0.14**	0.81 ± 0.08	0.22 ± 0.10	0.09 ± 0.07	0.76 ± 0.04	0.26 ± 0.04
Ind. Fair. Cor.	0.05 ± 0.03	0.68 ± 0.08	0.33 ± 0.08	0.04 ± 0.05	0.63 ± 0.09	0.37 ± 0.09	0.17 ± 0.04	0.85 ± 0.02	0.23 ± 0.03	0.08 ± 0.08	0.80 ± 0.03	**0.21 ± 0.04**
K-Median Cor.	0.04 ± 0.04	0.69 ± 0.08	0.31 ± 0.08	**0.01 ± 0.01**	0.60 ± 0.09	0.40 ± 0.09	0.23 ± 0.07	0.85 ± 0.02	0.27 ± 0.06	0.10 ± 0.05	0.79 ± 0.02	0.23 ± 0.03
Uniform Subsampling	0.04 ± 0.03	0.68 ± 0.10	0.32 ± 0.10	0.03 ± 0.05	**0.68 ± 0.12**	**0.33 ± 0.12**	0.29 ± 0.05	**0.88 ± 0.01**	0.31 ± 0.05	0.06 ± 0.05	0.79 ± 0.05	0.22 ± 0.05
K-Means	0.06 ± 0.03	0.66 ± 0.09	0.34 ± 0.09	0.02 ± 0.04	0.53 ± 0.04	0.47 ± 0.04	0.23 ± 0.05	**0.88 ± 0.01**	0.26 ± 0.04	0.09 ± 0.03	**0.82 ± 0.01**	**0.21 ± 0.02**
K-Medoids	0.05 ± 0.03	0.70 ± 0.12	0.30 ± 0.11	0.01 ± 0.02	0.61 ± 0.08	0.39 ± 0.08	0.12 ± 0.05	0.85 ± 0.04	**0.20 ± 0.04**	0.11 ± 0.05	0.81 ± 0.02	0.22 ± 0.03

Fairness-utility tradeoff when using coresets for data augmentation We also evaluate the performance of FWC in reducing the downstream demographic disparity when doing data augmentation, i.e., adding the synthetic representatives to the training data when training a downstream model. We use the data augmentation scheme adopted by [68, Section 2.1], which first uses k-means on the original dataset and then sorts the synthetic representatives based on the distance of each synthetic representative to the nearest k-mean centroid with the same combination of protected attribute and outcome D and Y. We generate a set of synthetic datasets of size equal to 50% of the original dataset and look at the fairness-utility of a downstream model trained augmented with such synthetic representative in increments of 5% (20% and 2.5% respectively for the Adult dataset, given the large dataset size). Figures 3 and 4 show the fairness-utility tradeoff of the downstream MLP classifier when doing data augmentation, selecting the best model across various degrees of data augmentation, along with the performance of the baseline MLP classifier with no data augmentation (averages and standard deviations over 10 runs). Table 7 shows the numerical values for the downstream fairness-utility tradeoff, including the tradeoff value computed as in Equation (26). In all datasets the data augmentation with FWC seems to either increase the performance or reduce the demographic disparity, with the only exception being the Drug dataset. Upon further investigation, Figure 5 shows this effect does not appear if the protected attribute D (gender) is not included in the features used to train the downstream MLP classifier. This phenomenon indicates the protected attribute provides a strong predictive effect on the outcome (whether the individual has tried cannabis or not), which might potentially be mediated by unmeasured confounders, i.e., other features regarding the recorded individuals that are not available in the Drug dataset. This would require either in-training or post-processing fairness approaches to be alleviated; see [29] for a comprehensive review on different

potential approaches. Finally, as in Figure 7, the standard deviations for Adult and Credit dataset are large due to the downstream model becoming trivial.

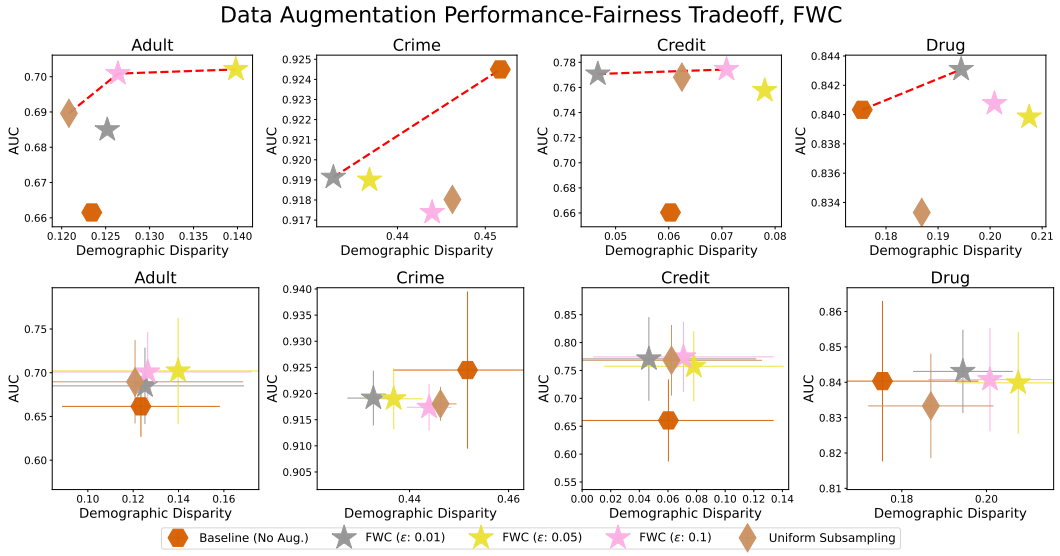

Figure 3: Fairness-utility tradeoff of downstream MLP classifier trained using the original training set augmented with coresets representatives, following the augmentation strategy from [68]. Each point shows the best model in terms of fairness-utility tradeoff over various degrees of data augmentation, in addition to the baseline model with no augmentation. Means and standard deviations taken over 10 runs, with the computed Pareto frontier indicated by the dashed red line.

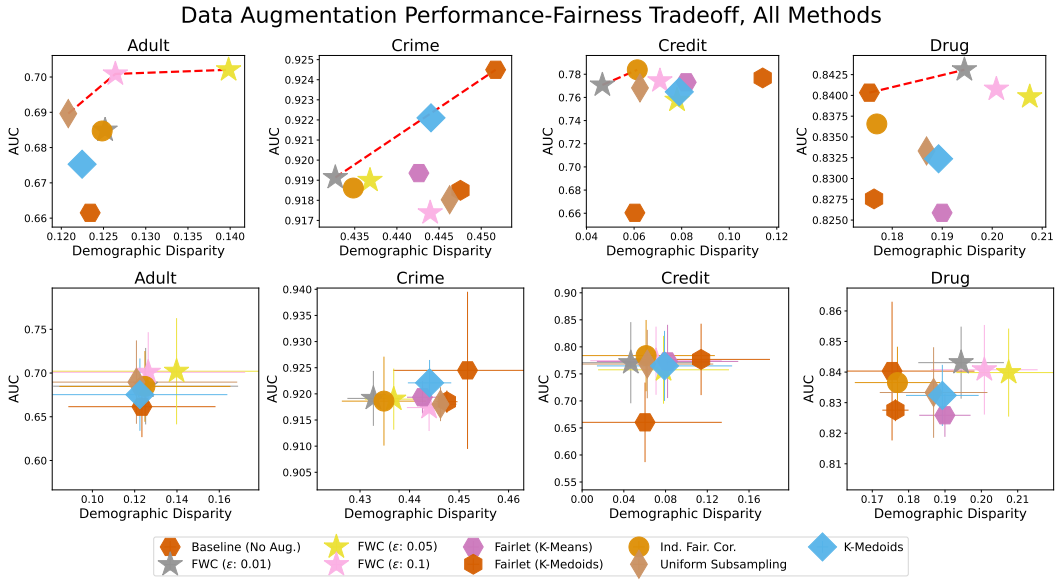

Figure 4: Fairness-utility tradeoff of downstream MLP classifier trained using the original training set augmented with coresets representatives, following the augmentation strategy from [68], including all methods mentioned in Section 7. Each point shows the best model in terms of fairness-utility tradeoff over various degrees of data augmentation, in addition to the baseline model with no augmentation. Averages and standard deviations computed over 10 runs, with the top panel showing just means and the bottom panel combining both means and standard deviations, with the computed Pareto frontier indicated by the dashed red line.

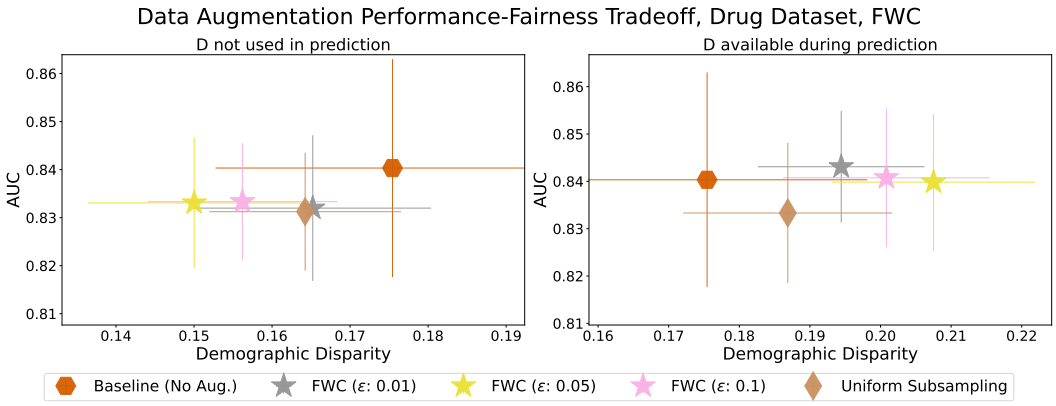

Figure 5: Data augmentation fairness-utility tradeoff of downstream MLP classifier for the Drug dataset when the protected attribute D (gender) is either not included (left) or included (right) as feature in the learning process. As in Figure 3, the best model across various data augmentation degrees is reported, with averages and standard deviations obtained over 10 runs. `FWC` manages to successfully reduce the demographic disparity when gender is not used as a feature, but fail to do so when gender is used, indicating that gender provides strong predictive power for the outcome in question, which would require enforcing fairness either during model training or by post-processing the outputs.

Table 7: Demographic disparity (Equation (25)), AUC and fairness-utility tradeoff (Equation (26)) of downstream MLP classifier trained via data augmentation with all fair coresets/clustering methods, across the four real datasets. The best methods across various degrees of data augmentation is shown. The best performing methods for every column is bolded, with averages and standard deviations taken over 10 runs.

Method	*Adult Dataset*			*Credit Dataset*			*Crime Dataset*			*Drug Dataset*		
	DD (↓)	AUC (↑)	Tradeoff (↓)	DD (↓)	AUC (↑)	Tradeoff (↓)	DD (↓)	AUC (↑)	Tradeoff (↓)	DD (↓)	AUC (↑)	Tradeoff (↓)
Baseline (No Aug.)	**0.12 ± 0.03**	0.66 ± 0.05	0.36 ± 0.05	0.06 ± 0.06	0.66 ± 0.11	0.34 ± 0.11	0.45 ± 0.05	**0.924 ± 0.015**	0.46 ± 0.04	**0.18 ± 0.03**	**0.84 ± 0.02**	**0.24 ± 0.03**
FWC (ϵ: 0.01)	0.13 ± 0.03	0.68 ± 0.07	0.34 ± 0.06	**0.05 ± 0.03**	0.77 ± 0.11	**0.23 ± 0.11**	**0.43 ± 0.04**	0.919 ± 0.005	**0.44 ± 0.04**	0.19 ± 0.03	**0.84 ± 0.01**	0.25 ± 0.02
FWC (ϵ: 0.05)	0.14 ± 0.03	**0.70 ± 0.09**	0.33 ± 0.08	0.08 ± 0.05	0.76 ± 0.09	0.25 ± 0.09	0.44 ± 0.03	0.919 ± 0.006	**0.44 ± 0.03**	0.21 ± 0.02	**0.84 ± 0.01**	0.26 ± 0.02
FWC (ϵ: 0.1)	0.13 ± 0.02	**0.70 ± 0.07**	**0.32 ± 0.06**	0.07 ± 0.05	0.77 ± 0.09	0.24 ± 0.09	0.44 ± 0.03	0.917 ± 0.004	0.45 ± 0.03	0.20 ± 0.03	**0.84 ± 0.01**	0.26 ± 0.02
Fairlet (K-Means)	-	-	-	0.08 ± 0.06	0.77 ± 0.10	0.24 ± 0.10	0.44 ± 0.03	0.919 ± 0.003	0.45 ± 0.03	0.19 ± 0.04	0.83 ± 0.01	0.26 ± 0.03
Fairlet (K-Medoids)	-	-	-	0.11 ± 0.07	**0.78 ± 0.10**	0.25 ± 0.09	0.45 ± 0.04	0.918 ± 0.002	0.45 ± 0.04	**0.18 ± 0.05**	0.83 ± 0.00	0.25 ± 0.04
Ind. Fair. Cor.	**0.12 ± 0.03**	0.68 ± 0.06	0.34 ± 0.06	0.06 ± 0.04	**0.78 ± 0.10**	0.22 ± 0.10	**0.43 ± 0.04**	0.919 ± 0.009	**0.44 ± 0.04**	**0.18 ± 0.02**	**0.84 ± 0.01**	**0.24 ± 0.02**
Uniform Subsampling	**0.12 ± 0.02**	0.69 ± 0.07	0.33 ± 0.07	0.06 ± 0.05	0.77 ± 0.09	0.24 ± 0.09	0.45 ± 0.04	0.918 ± 0.003	0.45 ± 0.04	0.19 ± 0.03	0.83 ± 0.01	0.25 ± 0.02
K-Medoids	**0.12 ± 0.04**	0.68 ± 0.06	0.35 ± 0.06	0.08 ± 0.05	0.76 ± 0.10	0.25 ± 0.09	0.44 ± 0.03	0.922 ± 0.004	0.45 ± 0.03	0.19 ± 0.02	0.83 ± 0.01	0.25 ± 0.02

Using `FWC` to correct biases in LLMs To query GPT models, test data with 200 samples is provided with a base parity of 0.5 (similar to [76]) and additionally, examples are provided in the case of the few shot settings. Data is fed in as text. One of the tabular data, for example, is: "*A person in 1996 has the following attributes: Age:* 21.0*, workclass: Private, education: Some-college, highest education level:* 10.0*, marital status: Never-married, occupation: Other-service, relationship: Own-child, race: White, sex: Female, capital gain:* 0.0*, capital loss:* 0.0*, hours per week:* 25.0*, native country: United-States*".

We use the following prompts:

- **Zero shot**: "*Using the provided data, will this person from 1996 be hired at greater than 50,000 USD per year? You must only respond with the word 'yes' or 'no'. Here are 0 examples with the correct answer.*"
- **Few shot**: "*Given the provided data, will this person from 1996 be hired at greater than 50,000 USD per year? You must only respond with the word 'yes' or 'no'. Here are* n *examples with the correct income level for a person in 1996. Make sure you use the examples as a reference*", where n is the number of demographically balanced samples.
- **Few shot** (`FWC`): "*Given the provided data, will this from 1996 be hired at greater than 50,000 USD per year? You must only respond with the word 'yes' or 'no'. Here are* n *examples with the correct income level for a person in 1996, along with weights in the*

column weight. Weights are between 0 (minimum) and 1 (maximum). The more the weight, the more important the example is. Make sure you use the examples as a reference."

In the few shot settings, 16 examples are provided in both cases due to the LLM token limitation; passing fewer examples yields similar results to the ones in Table 1. For FWC, we run a separate coreset generation run (differently from other experiments in Section 7), where we selected $m = 16$ and ensured that the positive class ($Y = 1$) has an equal number of male and female samples. We also note that while the results from GPT-4 for the zero shot and few shot cases are similar to what was observed by [76], the accuracies reported for the GPT-3.5 Turbo model appear to be lower in our experiments, which points to a potential difference in the exact backend LLM model used for inference.

D Limitations of FWC

Coreset support and non-convex feature spaces FWC representative do not need to be within the original n samples, but they do need to share the same support. As shown in Section 4.2, solving line 5 in Algorithm 1 in cases 1. and 2. means the synthetic representatives could fall outside the original dataset. In case 3., the solution has to be selected from the data points already existing in the original dataset (akin to k-medoids). In general, non-convex feature spaces $\mathcal{X}$ might represent a challenge, as representatives might be generated in zero-density regions (a simple example could be a dataset distributed as a hollow circle or two moons). However, this criticism is also more generally applicable to the existing fair coresets/clustering literature, as well as the k-means algorithms, for which specific adjustments have been developed [63] and could be indeed extended to FWC .

Limiting the total number of iterations FWC is not of polynomial time complexity and the total number of iterations might grow faster than linear time when the dataset size is very large, as shown in the synthetic data experiment in Section C.1 and Table 2. This phenomenon is shared with k-means, which is also not of polynomial time complexity and is known to potentially take an exponentially large number of iterations to terminate [73]. As mentioned in Section C.1, a common practice for clustering is set a fixed number of maximum iterations, after which the algorithm is stopped.

Computational bottlenecks The main complexity term for FWC is $\mathcal{O}(mn)$, which comes from establishing the cost matrix in the beginning of the solution of problem (8). This complexity is comparable with what occurs in Lloyd's algorithm for k-means and k-medians. This might be problematic if the cost matrix is too large to be stored directly in memory. In practice, we do not actually need to store the entire matrix, as we only need to compute the largest component for each row of C for solving problem (8) (see Lemma A.1), so one could further improve the cost of storing the cost matrix. Another option would also be leverage the same approaches used for k-means such as, e.g., cost matrix sketching [79]. Finally, FWC would also benefit from GPU implementations akin to k-means and k-medians, which would substantially accelerate the runtime speed of FWC .

Connection between ϵ and downstream learning In our definition of demographic parity in Equation 3, the hyper-parameter ϵ effectively controls how different the outcome rates across sensitive feature groups D of the weighted coreset distribution $p_{\hat{Z},\theta}$ can be from the overall outcome rates in the original dataset p_{Y_T}. In our experiments (Section 7), we empirically show that limiting the fairness violation in the coresets results in a fairer downstream model. However, when training a downstream model using FWC we induce a distribution shift between the train set and the test set, as the coresets distribution is never identical to the original dataset distribution. Although we have provided some results on the generalization properties of FWC (Proposition 5.5) as well as some intuition about developing downstream learning bounds for FWC in non-i.i.d. settings (Section B.1), theoretically characterize the connection between the fairness violation parameter ϵ remains an open question. In essence, the analysis is challenging as the induced distribution shift is dependent on the biases in the original dataset distribution, the coreset size m, the metric chosen for the cost matrix and, ultimately, the fairness violation parameter ϵ. For this reason, although we have shown that restricting the fairness violation improves downstream models fairness, an explicit characterization of the downstream effects

of ϵ, as well as other hyper-parameters, would require significant further work, beyond the scope of this paper.

Implications for other fairness measures As highlighted by [5, Chapter 3], fairness notions in classification settings can be categorized into notions of independence, separation, and sufficiency. Demographic parity falls in the class of the independence notion, and hence, other measures of fairness that are closely related, e.g., disparate impact, would also improve when optimizing for demographic parity. However, other notions of fairness such as separation or sufficiency may not simultaneously be satisfied [5]. As FWC targets demographic parity, it cannot guarantee an improvement in these other measures. To test this, we compute the equalized odds, which falls under the notion of separation, for the downstream classifier in Section 7 and check the performance of FWC compared to the other approaches. Table 8 indicates the datasets in which FWC is part of the Pareto frontier for both demographic parity and equalized odds. When considering equalized odds, FWC is not a part of the Pareto frontier for the Drug dataset, and more generally, FWC performance is not as competitive. This is in contrast to demographic parity: in Figure 1, FWC sits on the Pareto frontier across all datasets for fairness-performance trade-off in downstream classification.

Table 8: Presence on the Pareto frontier for FWC across different fairness violation hyper-parameter values ($\epsilon = \{0.01, 0.05, 0.1\}$), for both demographic parity (left) and equalized odds (right) in the downstream learning settings of Section 7. As equalized odds is not an independence notion of fairness as demographic parity, constraints on demographic parity do not guarantee an improvement in equalized odds, resulting in FWC not performing as well for downstream performance-fairness tradeoff when using equalized odds.

	Pareto Frontier, Demographic Parity			**Pareto Frontier, Equalized Odds**		
Dataset	FWC ($\epsilon = 0.01$)	FWC ($\epsilon = 0.05$)	FWC ($\epsilon = 0.1$)	FWC ($\epsilon = 0.01$)	FWC ($\epsilon = 0.05$)	FWC ($\epsilon = 0.1$)
Adult	✓	✓		✓		
Drug	✓	✓	✓			
Crime	✓	✓	✓			✓
Credit	✓			✓		✓

E Broader Impact

Our work presents a novel approach to obtain coresets (synthetic representative samples) of a given dataset while reducing biases and disparities in subgroups of the given dataset. As other approaches in the field of algorithmic fairness, our efforts may help populations that would otherwise face disadvantages from a model or decision process. Importantly, our approach refrains from exploiting biases inherent in the data itself; rather, it seeks to mitigate biases in data-driven decision systems. It is crucial to note that our method does not claim to address all sources or types of bias. In addition, while our tools enable a malicious modeler to manipulate algorithmic fairness methods to amplify disparities instead of reducing them, for instance, by reversing the fairness constraint (replacing $\leq$ with $\geq$), the unfairness of a trained model can be detected by assessing it over a separate test set from the original dataset.

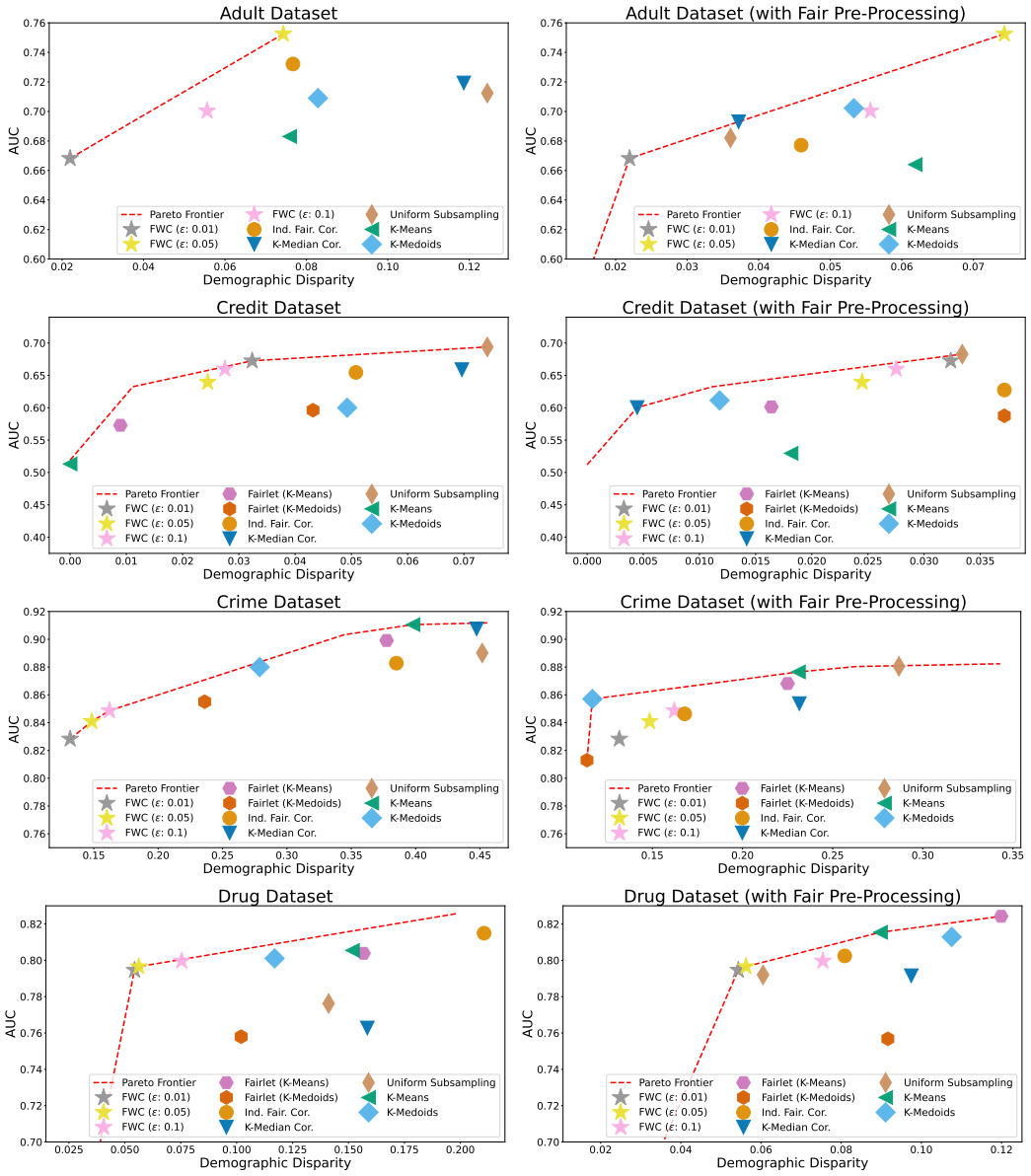

Figure 6: Fairness-utility tradeoff of all methods, indicated by AUC and demographic disparity of a downstream MLP classifier across all datasets (rows) and without (left column) or with (right column) fair pre-processing [34] (excluding `FWC` , to which no fairness modification is applied after coresets have been generated). The Pareto frontier, indicated with a dashed red-line, is computed across all models and coreset sizes. We report the averages over 10 separate train/test splits.

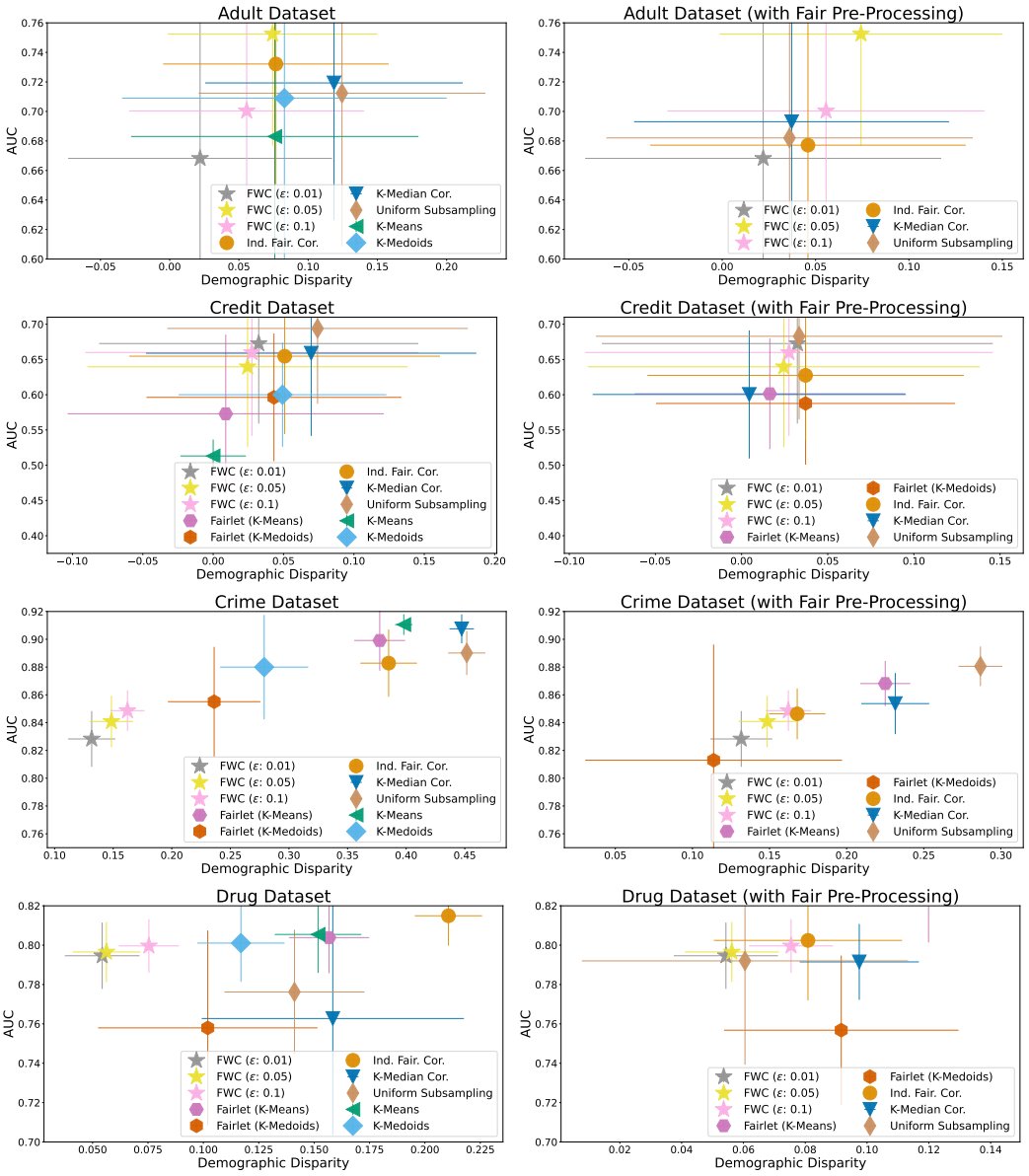

Figure 7: Similarly to Figure 6, we report the means and standard deviations over 10 runs of the fairness-utility tradeoff of all methods, indicated by AUC and demographic disparity of a downstream MLP classifier across all datasets (rows) and without (left column) or with (right column) fair pre-processing [34] (excluding `FWC` , to which no fairness modification is applied after coresets have been generated).