Fairness Using Discrepancy Minimization

October 26, 2020

Abstract

With the rise of the era of artificial intelligence and machine learning in the last decade there has been an increasing interest in developing a strong theory and implementation of *Algorihtmic Fairness* which has eventually resulted in a large volume of work over the past few years. Despite the huge amount of work done on the topics over a very short period of time, there has been little consensus of a unifying theory of algorithmic fairness. In this paper we present a notion of implementing algorithmic fairness that is based on the *discrepancy theory* and provide justifications towards its usefulness.

1 Introduction

NG: So we are basically targetting equal opportunity. In the past decade we have witnessed a rapid progress and growth of machine learning algorithms and artificial intelligence due to which there has been a growing interest to develop algorithms that will make policy oriented decisions like recruitment, selection in Universities, promotion for a job title, crime recidivism etc. Due to the growth of machine learning algorithm people believe that machines will eventually make better decision in these scenarios than humans who are very prone to error and bias. Because of this there has been a lot of research done to define a notion of *fairness* for algorithms. In the context of decision-making, fairness in a subjective sense is defined as *the absence of any prejudice or favoritism toward an individual or a group based on their inherent or acquired characteristics.* But converting this idea to an implementable and mathematical definition understandable in the language of algorithms is a challenging task and the machine learning community is still struggling to find a unifying solution to this problem. Several algorithms that are used in US for making decisions have been found to be unfair. For example the software COMPAS (Correctional Offender Management Profiling for Alternative Sanctions) has been found to be biased [17]. In another case of a hiring application, it was recently exposed that Amazon discovered that their AI hiring system was discriminating against female candidates, particularly for software development and technical positions [17]. These examples show that the present technology has not been able to give a fully satisfying solution to the problem of implementing algorithmic fairness. In this paper we study a notion of fairness that is based on the notion of discrepancy of set systems, a widely studied topic in the theory of computer science and combinatorics [7].

2 Fairness in Machine Learning

In this section we describe the notions of fairness that have been proposed over the past few years and have been popular in the research community of machine learning. Along with the definitions of fairness there are also the notions of *biases* in this domain which reflect the kind of biases that an algorithm can be prone to. We mention them in brief in this section, for details the reader may refer to [4]. In what follows we will assume S represents the protected attribute (e.g., race or gender), $S = 1$ is the privileged group, and $S \neq 1$ is the unprivileged group. $\hat{Y} = 1$ means that the prediction is positive. Let us note that if $\hat{Y} = 1$ represents acceptance (e.g., for a job), then the condition requires the acceptance rates to be similar across groups. A higher value of this measure represents more similar rates across groups and therefore more fairness. Note that this notion relates to the "80 percent rule" in disparate impact law, which requires that the acceptance rate for any race, sex, or ethnic group be at least 80% of the rate for the group with the highest rate.

2.1 Measures of Algorithmic Bias

The following are considered as some stantard notions of fairness which have been developed over the years. Here we provide a brief description and the reader can refer to the details in numerous surveys available on fairness like [17].

- **Disparate Impact**: This measure was designed to mathematically represent the legal notion of disparate impact. It requires a high ratio between the positive prediction rates of both groups. This ensures that the proportion of the positive predictions is similar across groups. The constraint can be formulated as

  $$\mathbb{P}[\hat{Y} = a | S \neq 1] \geq \mathbb{P}[\hat{Y} = a | S = 1](1 - \epsilon)$$

  where a can be either 1 or -1 (representing a selection or not, deemed to offened or not) etc. Here ϵ is a fraction close to zero. For eg. in the 80% rule ϵ will be 0.2 and for complete fairness ϵ can be chosen to be 0.

- **Demographic Parity**: This measure is similar to disparate impact, but the difference is taken instead of the ratio [28, 44]. This measure is also commonly referred to as statistical parity. Formally, this measure is computed as follows:

  $$\Big| \mathbb{P}[\hat{Y} = a | S \neq 1] - \mathbb{P}[\hat{Y} = a | S = 1] \Big| \leq \epsilon$$

- **Equalized Odds**: This measure was designed to overcome the disadvantages of measures such as disparate impact and demographic parity. The measure computes the difference between the false positive rates (FPR), and the difference between the true positive rates (TPR) of the two groups.

$$\Big|\mathbb{P}[\hat{Y}=1|S\neq 1, Y=1]-\mathbb{P}[\hat{Y}=1|S=1, Y=1]\Big|\leq\epsilon$$
$$\Big|\mathbb{P}[\hat{Y}=1|S\neq 1, Y\neq 1]-\mathbb{P}[\hat{Y}=1|S=1, Y\neq 1]\Big|\leq\epsilon$$

- **Equal Opportunity**: This requires true positive rates (TPRs) to be similar across groups (meaning the probability of an individual with a positive outcome to have a positive prediction). This measure is similar to equalized odds but focuses on the true positive rates only. This measure is mathematically formulated as follows:

$$\Big|\mathbb{P}[\hat{Y}=1|S\neq 1, Y=1]-\mathbb{P}[\hat{Y}=1|S=1, Y=1]\Big|\leq\epsilon$$

- **Individual Fairness**: This requires that similar individuals will be treated similarly. Similarity may be defined with respect to a particular task. Individual fairness may be described as follows:

$$\Big|\mathbb{P}[\hat{Y}^i=a|X^i, S^i]-\mathbb{P}[\hat{Y}^j=a|X^j, S^j]\Big|\leq\epsilon$$

where we assume that $d(i,j)\leq\delta$ according to some metric $d(.,.)$ and S^i represent the sensitive features of i and X^i represent the associated features of i.

2.2 Representation Learning Approach

Representation learning is the task to perform on high dimensional data inputs and get approximation of it to low dimensional feature space. Usually Principal Component Analysis and Autoencoders performs similar tasks that can be mathematically formulated as $X, Y \rightarrow Z$ where X is the input data, Y is the corresponding labels to the input and Z represents the reduced feature space corresponds to X. The notion of fairness from representation learning seems to achieve by disentanglement of sensitive features from X in transformed feature space Z. The representation learning approaches are widely used for disentangling the sensitive features to reduce the cause for unfairness []. As we disentangle these sensitive features in higher dimensional spaces, mostly done by VAE's, GAN's and other deep learning model to achieve fairness [] [] . Apart from the intuition and results of it on some data sets [] [] [] , it's still facing criticism for not delivering a generalized fairness as it had promise [].

3 Discrepancy Theory

Discrepancy theory is an important topic of study that basically originated from questions in analytic number theory which were asked by pioneer mathematicians like van der Corput [10], Schmidt and Erdos [22] in early Twentieth century who studied the combinatorial and analytic properties of infinite sequences of integers in a particular range. This lead to the development of several variants and questions of number theory which were collectively called discrepancy theory. In computer science, nowadays there has been extensive use of discrepancy theory including areas in complexity theory, probabilistic algorithms, pseudo-randomness, computational geometry, machine learning, communication complexity, mathematical finance and computer graphics [7]. In this paper we will develop an application of discrepancy theory in the newly developing field of algorithmic fairness.

3.1 Combinatorial Discrepancy

In this paper, we mostly deal with the notion of combinatorial discrepancy [7] that is defined for set-systems in combinatorics. Given a universal set X consisting of n elements and a collection $S = \{S_1, S_2 \dots S_m\}$ of subsets of X. The objective is to find a coloring $\chi : X \rightarrow \{-1, 1\}$ such that the *discrepancy* of the set system is minimized. Where the discrepancy of a particular set S_j w.r.t. a coloring χ is defined as

$$D(S_j) = \Big|\sum_{a \in S_j} \chi(a)\Big|$$

Our aim is to find

$$\begin{aligned} D(X, S) &= \min_{\chi \in \{-1,1\}^n} \max_{S_i \in S} D(S_i) \\ &= \min_{\chi \in \{-1,1\}^n} \max_{S_i \in S} \Big|\sum_{a \in S_j} \chi(a)\Big| \end{aligned}$$

This problem has been studied from both algorithmic and hardness point of views for over two decades and very recently there has been a plethora of results obtained for this problem because of emphasis on its research in theoretical computer science community. More formally, in a very seminal paper Spencer [21] showed that there always exist a coloring that achieves a discrepancy of $6\sqrt{n}$. This result is called the six-standard deviation result. Although this result is non-constructive, constructive versions started coming up beginning from the work of Bansal [3] who gave an SDP based algorithm to find a coloring of low discrepancy. This result was simplified by Lovett and Meka [16]; several constructive results appeared after these work most importantly the work of Larsen [15] who provided an implementable (in Python) algorithm to find a low discrepancy coloring. Although theoretically his result isn't as sound as previous constructive algorithms, the implementability of his algorithm for large values of n makes it more important in the machine learning set-up where we are interested in implementability of algorithms and not just theoretical guarantees. Kasper's algorithm finishes in a reasonable amount of time on matrices of sizes up to 10000×10000 where the matrix is the matrix corresponding to the adjacency properties of the given set system. For several datasets in machine learning we have less than 10000 records and hence the algorithm is implementable on those sets. Some hardness results are also known for this problem [6].

4 Discrepancy as a Measure of Fairness

In this section we describe how the notion of combinatorial discrepancy can be used to define a notion of fairness for machine learning algorithms. Given a particular dataset we first create an instance of the discrepancy problem by defining (X, S) as follows: X will be the set of records in the dataset s.t. $|X| = n$ and for each sensitive attribute i (like race, sex , religion) we construct $S_i^j \in S$ which consist of all the elements of X that have a fixed value of j of the sensitive attribute i. For example if i is 'sex' then there can be two possible values of j corresponding to 'Male' and 'Female'; S_i^1 will consists of all the elements of X which are Male and S_i^2 will consists of the females. Thus we have created an instance of the discrepancy problem. We can also represent the collection S of sets by a $k \times n$ matrix R whose rows represent the characteristic vector of each $S_i \in S$.

Meaning of a Coloring: As we have seen that there is a natural instance of the discrepancy problem for a given dataset with certain sensitive attributes. Now we need to understand the meaning of a low discrepancy coloring for the set system obtained as above. Intuitively a low discrepancy coloring will balance the colors for each of the categories of the sensitive attributes. For example if a coloring represents whether a person gets a job or not then a low discrepancy coloring will ensure that the number of selected females is roughly same as number of non-selected ones and number of selected males is roughly same as the rejected ones i.e. formally we can put it as

$$\left| |\sum_{a \in S_j^i, \chi(a)=1} \chi(a)| - |\sum_{a \in S_j^i, \chi(a)=-1} \chi(a)| \right| \leq \epsilon$$

The above inequality will be true for any set S_j^i if the discrepancy of the overall coloring is bounded by ϵ. Thus a low discrepancy coloring ensures a certain notion of balancing among the protected attributes in the datasets. We will incorporate this coloring to measure the fairness of a classification algorithm A. We will propose three different ways in which we can use the coloring returned by the algorithm solving the discrepancy problem. First is the randomized approach, second is the linear programming based approach and third is a non-convex programming approach.

4.1 Randomized Approach

Definition 1 *Let A_1 be a binary classification algorithm on a dataset D and A_2 be an algorithm that returns a coloring of discrepancy bounded by ϵ of the corresponding instance of D. We generate new labels called Fair$(A_1, A_2, \alpha, \epsilon)$ labelling, if on all the records in which A_1 disagrees with A_2, with probability α we choose the label assigned by A_2 and with probability $1 - \alpha$ we choose the labelling of A_1.*

As with the prior work on algorithmic fairness [23] it has been noticed that a trade off w.r.t *accuracy* arises when we try to implement a notion of fairness. We try to develop similar trade-offs in the above definition as well. Since α measures the amount of fairness in the final labelling, we need to derive a relation between α and Acc_f which is the accuracy of the final labelling. The following result justifies that.

Lemma 1 *$\mathbb{E}[Acc_f] = (1-\alpha)Acc_{A_2} + \alpha Acc_{A_1}$ where Acc_{A_2} is the training accuracy of A_2 and Acc_{A_1} is the accuracy of A_1. Thus the expected accuracy of the final coloring is a linear combination of the accuracies of A_1 and A_2.*

Proof: Since $\mathrm{Fair}(A_1, A_2, \alpha, \epsilon)$ is a randomized algorithm, that returns a final binary (± 1) labelling f on the records that has actual labellings as r, Acc_f is a random variable and we need to find its expected value.

$$\begin{aligned}
\mathbb{E}[\mathrm{Acc}_f] &= \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}[\mathbb{1}f(x_i) = r(x_i)] \\
&= \frac{1}{n}\sum_{i=1}^{n}\mathbb{P}[f(x_i) = r(x_i)] \\
&= \frac{1}{n}\sum_{i=1}^{n}\{\alpha\mathbb{1}[A_1(x_i) = r(x_i)] + (1-\alpha)\mathbb{1}[A_2(x_i) = r(x_i)]\} \\
&= \alpha\{\sum_{i=1}^{n}\mathbb{1}[A_1(x_i) = r(x_i)]\} + (1-\alpha)\{\sum_{i=1}^{n}\mathbb{1}[A_2(x_i) = r(x_i)]\} \\
&= (1-\alpha)\mathrm{Acc}_{A_2} + \alpha\mathrm{Acc}_{A_1}
\end{aligned}$$

□

4.2 Linear Programming Based Approach

In this section we describe how can we use a linear programming relaxation for the problem of discrepancy minimization. Here we use the solution if an LP to arrive at a coloring of low discrepancy. Although in general the discrepancy problem is NP-hard, we are interested in getting a LP relaxation to the problem that can give us a coloring of low enough discrepancy. Recall that our discrepancy problem was

$$\begin{aligned}
D(X, S) &= \min_{\chi\in\{-1,1\}^n}\max_{S_i\in S} D(S_i) \\
&= \min_{\chi\in\{-1,1\}^n}\max_{S_i\in S}\left|\sum_{a\in S_j}\chi(a)\right|
\end{aligned}$$

This optimization problem is an integer linear programming that involves the minimization of a modulus function and it is known that such an optimization can be relaxed to linear programming problem as follows: we introduce a new variable to the system say z and perform the following optimization.

$$\begin{aligned} & \min z \\ z &\geq \sum_{a \in S_j} \chi(a) \ \forall S_j \in S \\ z &\geq -\sum_{a \in S_j} \chi(a) \ \forall S_j \in S \\ z &\geq 0 \\ -1 &\leq \chi(a) \leq 1 \ \forall a \in X \end{aligned}$$

We will call this linear program **LP-1**. This LP relaxation has been studied extensively from the theoretical point of view in the so called Beck-Fiala [5] setting for discrepancy with the aim of getting relevant theoretical bounds for the problem. But here we are more interested in its implementability for large number of variable. We can use well known libraries of Python like PuLP to solve the above Linear Programming problem. It turns out that this LP runs much faster than the algorithm due to Kasper and hence is more useful for our set up.

4.2.1 Prioritizing Sensitive Attributes

In several applications we notice that each of the sensitive attribute is not of equal sensitivity for eg. in case of the german dataset, "Age" is more sensitive than "Gender" attribute. Thus, there is need to define a notion of priority among the sensitive attributes in order to screen certain constraints of the LP. To ensure such a prioritizing we introduce the notion of a *priority vector* $\mathbf{p}$ along with LP that decides how the constraints in the LP are to be selected. The vector $\mathbf{p} = (p_1, p_2, \ldots, p_m)$ is given as part of the input by the user and $0 \leq p_i \leq 1$ represents the probability of selecting the constraints, corresponding to i^{th} sensitive attribute in our LP. In general, $\sum_i p_i \neq 1$, and p_i represents the importance of the i^{th} sensitive attribute. Thus to check whether the constraints corresponding to i^{th} sensitive attribute is put in the LP or not we do the following;

$$p_i = \begin{cases} 0, & \text{Constraints are not put in the LP} \\ 1, & \text{Constraints are put in the LP} \\ \beta, & \text{Pick a random number } r \text{ in (0,1),} \\ & \text{if } r \leq \beta \text{ put the constraints otherwise not} \end{cases}$$

Thus based on the $\mathbf{p}$ vector we have given a randomized procedure to select the relevant constraints and solve for the value of z in the aforementioned linear program. We can have another interpretation of the vector $\mathbf{p} = (p_1, p_2, \ldots, p_m)$ and have a deterministic algorithm for the problem by simply sorting the p_i's and selecting the top k in this sorted list where k will be given as part of the input.

4.2.2 Controlling Discrepancy

We can also conceive of applications in which we need to control the discrepancy rather than minimizing it. Think of an examination in which we want to enforce that the ratio of selected and non-selected candidates is not 1 but a very small number like 0.01. This scenario arises in several highly competitive examinations (SAT, JEE etc.) [13]. In these scenarios it won't be a good idea to find a coloring that minimizes discrepancy because that may lead to a a 50/50 split (or a ratio close to 1). Consider a set system with discrepancy $\epsilon \geq 0$; as the value of ϵ tends to zero the corresponding coloring χ ensure that each set consists of almost equal numbers of +1s and -1s which may not be needed in every scenario. Thus in order to retrieve a coloring that can control the discrepancy we can use the following variant of the above linear program.

$$\begin{aligned} & \min z \\ z &\geq \sum_{a \in S_j} \chi(a) \;\; \forall S_j \in S \\ z &\geq -\sum_{a \in S_j} \chi(a) \;\; \forall S_j \in S \\ \epsilon_2 n &\geq z \geq \epsilon_1 n \\ -1 &\leq \chi(a) \leq 1 \; \forall a \in X \end{aligned}$$

We will call this optimization problem **LP-2**. This LP allows us to vary ϵ_1, ϵ_2 to control the amount of discrepancy/fairness needed in our application.

4.2.3 Alternate Definition of Discrepancy

We have been minimizing maximum discrepancy, we can consider other notions of discrepancy in which we minimize the sum of discrepancies over all the sets in the set system. More formally we define

$$\hat{D}(X, S) = \min_{\chi \in \{-1,1\}^n} \sum_{S_i \in S} \Big| \sum_{a \in S_j} \chi(a) \Big|$$

as the new discrepancy measure. We can easily write a linear program that is a relaxation to the above problem.

$$\begin{aligned} & \min \sum_i z_i \\ z_j &\geq \sum_{a \in S_j} \chi(a) \;\; \forall S_j \in S \\ z_j &\geq -\sum_{a \in S_j} \chi(a) \;\; \forall S_j \in S \\ z_j &\geq 0 \; \forall S_j \\ -1 &\leq \chi(a) \leq 1 \; \forall a \in X \end{aligned}$$

This program, which we refer to as **LP-3** is similar to the previous LP can be solved using standard LP solvers.

4.3 Non-Convex Programming Based Approach

In this section we propose another framework that is based on non-convex optimization. In this approach we do a combined optimization of the loss function of a particular classifier like SVM along with the objective function corresponding to the discrepancy problem. More formally if $\mathbf{w}$ is variable vector corresponding to SVM classifier and $f(\mathbf{w})$ is the objective function of the SVM and $g(\mathbf{w}) \in C$ be the feasible set of the corresponding optimization problem; then we define a new objective function that involves both $f(\mathbf{w})$ and discrepancy objective. Recall according to the the *kernel trick* of SVM optimization, we assume that the points to be classified are mapped to a higher dimensional space via a mapping ϕ such that the similarity in that space is governed by the *kernel function* and the data is linearly classifiable by a hyperplane $y = \langle \mathbf{w}, \phi(\mathbf{x}) \rangle - b$ in that space. Concisely, the optimization problem solved in that setting is as follows [19].

$$\mathbf{w} = \sum_{i=1}^{n} c_i y_i \phi(\mathbf{x}_i)$$

where the c_i are solved by the following dual optimization problem

$$\begin{aligned} \min \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} y_i c_i \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle c_j y_j - \sum_{i=1}^{n} c_i \\ \sum_{i=1}^{n} c_i y_i = 0 \;\; \forall i \\ 0 \leq c_i \leq \frac{1}{2n\lambda} \;\; \forall i \end{aligned}$$

The value of b can be solved as

$$\begin{aligned} b &= \langle \mathbf{w}, \phi(\mathbf{x}_j) \rangle - y_j \\ &= \left[\sum_{i=1}^{n} c_i y_i \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle \right] - y_j \end{aligned}$$

Here in the entire optimization we will use that fact that $K(\mathbf{x}_i, \mathbf{x}_j) = \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle$ because of the kernel trick. Thus the coloring of a record $\mathbf{x}_j$ can be predicted as

$$\operatorname{sgn}\left(\left[\sum_{i=1}^{n} c_i y_i K(\mathbf{x}_i, \mathbf{x}_j) \right] - b \right)$$

In our approach we will use this coloring directly in the discrepancy objective function and find a value of $\mathbf{w}$ that minimizes both the terms. The main idea is to write the discrepancy objective as

$$\begin{aligned} D(X, S) &= \min_{\chi \in \{-1,1\}^n} \max_{S_i \in S} \Big| \sum_{a \in S_j} \chi(a) \Big| \\ &= \min_{c_1, c_2, \ldots c_n} \max_{S_i \in S} \Big| \sum_{\mathbf{x}_j \in S_i} \operatorname{sgn}\left(\left[\sum_{i=1}^{n} c_i y_i K(\mathbf{x}_i, \mathbf{x}_j) \right] - b \right) \Big| \end{aligned}$$

Thus our combined objective function becomes minimization of

$$
\begin{aligned}
&(1-\alpha)\Big\{\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} y_i c_i K(\mathbf{x}_i,\mathbf{x}_j) c_j y_j - \sum_{i=1}^{n} c_i\Big\} \quad + \\
&\alpha\Big\{\max_{S_i \in S}\Big|\sum_{\mathbf{x}_j \in S_i} \operatorname{sgn}\Bigg(\Bigg[\sum_{i=1}^{n} c_i y_i K(\mathbf{x}_i,\mathbf{x}_j)\Bigg] - b\Bigg)\Big|\Big\}
\end{aligned}
$$

$$
\begin{aligned}
\sum_{i=1}^{n} c_i y_i &= 0 \ \forall i \\
0 \le c_i &\le \frac{1}{2n\lambda} \ \forall i
\end{aligned}
$$

Now we know that $\operatorname{sgn}(x)$ function is not a continuous function; we approximate it with a non-convex smooth function like $\tanh(x)$ or $\frac{x}{\sqrt{x^2+1}}$ and use the similar techniques in the previous section to remove the modulus sign from the optimization which eventually gives us a non-convex optimization problem over the variables $(c_1, c_2, \cdots c_n)$. In general SVM deals with a quadratic objective function but because of the joint optimization approach we need to solve a non-convex optimization problem. Since non convex optimization suffers from problem related to global and local optimization; there are solvers like Natasha [1], Natasha2 [2] which can give a local minima as output.

4.4 Understanding the Three Approaches

Here we try to give a motivation for the three approaches introduced by us by showing that these approaches are trying to achieve notions of fairness by trading with other factors. In the first approach we use a randomized algorithm to choose the coloring between Kasper and SVM and notice that expected accuracy of the final classifier is a weighted linear combination of the accuracies of the two algorithms. In the second approach we use the same technique to merge the values but A_2 is chosen as the LP based coloring. This choice might give a coloring of discrepancy value lower than Kasper but the LP based coloring might be faster than Kasper for large values of m, n. Theoretically, Kasper's algorithm runs in time $O((m+n)n^2)$ and best LP algorithm will run in $O((m+n)n^{3+\frac{1}{18}})$ [12] that is theoretically worse than Kasper's running time but in practice the running of solvers like PuLP is significantly faster than the algorithm of Kasper. The LP based approach also helps us to achieve a framework for the non-convex optimization. The non-convex programming formulation doesn't do a merging of two independent algorithms A_1 and A_2 but rather does a joint optimization of the two objective functions to learn a single $\mathbf{w}$. This approach might lead to difficulty in solving the optimization problem but gives a single model to incorporate fairness using discrepancy.

5 Datasets

In our study we have used 3 different data set such as ProPublica risk assessment, ProPublica violent risk assessment and German credit data set . All results are obtained using 66%/33% train/test splits.

ProPublica recidivism The ProPublica data is obtained from COMPAS risk assessment system[]. It includes 6167 records along with 13 attributes for and The target variable for each criminal to predict whether criminal recidivated (rearrested) or not within two years from his last release from the prison. The data have attributes such as crime charge degree, decile score, score text etc. with sensitive attributes are gender and race.

ProPublica violent recidivism The violent recidivism[] is another version of ProPublica data. The data having records of 4010 criminals with same attributes as of ProPublica data. The target variable is whether the criminal rearrest or not within two years only for violent crimes. The sensitive attributes are race and gender same as for ProPublica data.

German The German credit data set obtained from publicly available UCI repository, dealing with gender and age inequality for credit related issues. The data includes 1000 records with 20 attributes and the target variable that determine whether an individual having predicted credit risk score as good or bad. The attributes are credit history, purpose, credit amount, saving amount, property, job, housing etc. with Personal status, sex, and age as the sensitive attributes.

6 Baselines

Although there has been a plethora of work done in fairness with regard to classification algorithms [8, 9, 14, 18, 11], we restrict our attention to the work of Zafar et al [23] in which the results obtained are similar in spirit to our results. The main idea in that paper is to work with a standard classification algorithm like SVM and code the fairness constraints in the SVM optimization problem itself resulting in quadratic programming problem with convex concave constraints also called DCCP (Disciplined Convex Concave Problem) which can be solved to arbitrary accuracy [20]. Our approach is similar in spirit but instead of changing the constraints we change the objective function to enforce the fairness conditions thereby obtaining our LP and Non-Convex formulations. Since in our LP based approach the variables of the LP are independent of the variables of the SVM we can perform the two optimizations independently and merge them using the randomized approach. In this paper we have focused our analysis on LP based framework and keep the non-convex analysis for future work.

7 Experiments

In the experiments we have used linear SVM (Support Vector Machine) classifier for prediction calling it A_1 algorithm for training the model along with this we have used Kasper's coloring algorithm as A_2 to return a low discrepancy coloring for sensitive attributes. So in this randomized approach we have constrained accuracy over α. In this method α is compared with randomized z, if $\alpha > z$ then final label of the record is given according the coloring obtained from A_2 algorithm otherwise it will choose the label according to A_1 algorithm. In this approach we have used Kasper's discrepancy coloring with α-constraints as fairness criteria.

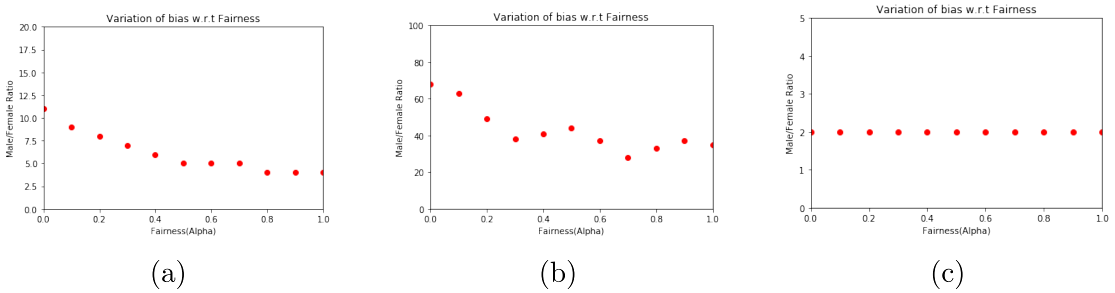

Figure 1: Variation of Bias in Sensitive Attributes w.r.t Fairness parameter (α) (a) ProPublica (Gender) (b) ProPublica Violent (Gender) (c) German (Gender)

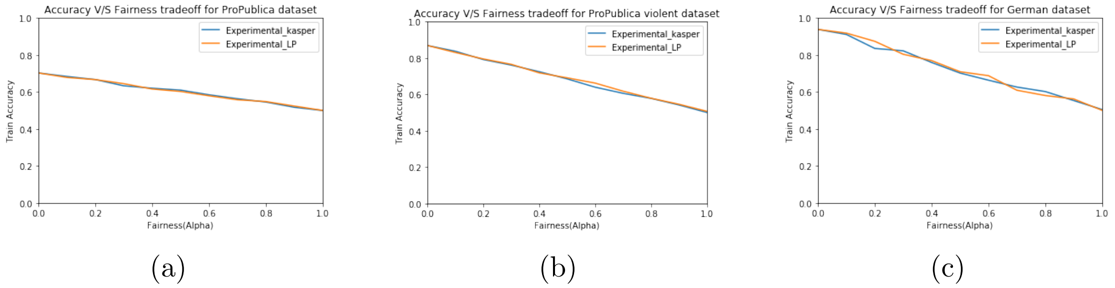

Figure 2: Accuracy vs. Fairness Tradeoffs for Kasper and **LP-1** algorithms with sensitive attributes as (a) Propublica (Race , Gender); (b) Propublica Violent (Race, Gender) (c) German (Age, Marital Status, Gender)

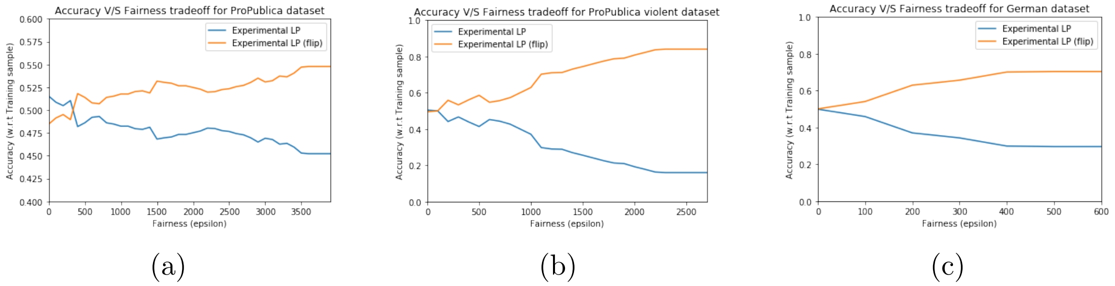

Figure 3: Accuracy vs. Discrepancy (ϵ) trade-offs for **LP-2** (a) Propublica (Race , Gender); (b) Propublica Violent (Race, Gender) (c) German (Age, Marital Status, Gender)

8 Realism of Our Definitions

Since in the very beginning we had noticed that fairness is a subjective concept and any definition of fairness would be incomplete without a philosophical discussion. In this section we give a philosophical and scientific perspective of the definitions we have provided in the paper and analyse whether our definitions are realistic and implementable in the society or not. In this paper we have mostly been focused with the fairness of classification algorithms and hence our sensitive attributes are sensitive in the "legal" sense. It has been found by neuro-scientists that the need of fairness is similar to the need of food i.e. similar neurons are fired when questions about fairness and food are raised []. This observation lets us to believe that fairness is more of an individual need and may vary from one person to other. Thus a definition of fairness that allows individuals to choose the amount of fairness they want should be more realistic. Also too much freedom to people might create chaos and hence some constraints should also be imposed. These issues are answered by our first randomized approach where the fairness parameter α can be thought to be controlled by the government and it allows the people to choose the amount of fairness they want from the classification tasks. Thus our notions of fairness incorporate both individual and group fairness ideas and are somewhat *democratic* in nature. [23]

9 Conclusion

References

[1] Zeyuan Allen-Zhu. Natasha: Faster non-convex stochastic optimization via strongly non-convex parameter. *arXiv preprint arXiv:1702.00763*, 2017.

[2] Zeyuan Allen-Zhu. Natasha 2: Faster non-convex optimization than sgd. In *Advances in neural information processing systems*, pages 2675–2686, 2018.

[3] Nikhil Bansal. Constructive algorithms for discrepancy minimization. In *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, pages 3–10. IEEE, 2010.

[4] Solon Barocas, Moritz Hardt, and Arvind Narayanan. Fairness in machine learning. *NIPS Tutorial*, 1, 2017.

[5] József Beck and Tibor Fiala. "integer-making" theorems. *Discrete Applied Mathematics*, 3(1):1–8, 1981.

[6] Moses Charikar, Alantha Newman, and Aleksandar Nikolov. Tight hardness results for minimizing discrepancy. In *Proceedings of the twenty-second annual ACM-SIAM symposium on Discrete Algorithms*, pages 1607–1614. SIAM, 2011.

[7] Bernard Chazelle. *The discrepancy method: randomness and complexity.* Cambridge University Press, 2001.

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

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

[10] JG Corput. van der: Verteilungsfunktionen. In *Proc. Ned. Akad. v. Wet*, volume 38, pages 813–821, 1935.

[11] Michael Feldman, Sorelle A Friedler, John Moeller, Carlos Scheidegger, and Suresh Venkatasubramanian. Certifying and removing disparate impact. In *proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pages 259–268, 2015.

[12] Shunhua Jiang, Zhao Song, Omri Weinstein, and Hengjie Zhang. Faster dynamic matrix inverse for faster lps. *arXiv preprint arXiv:2004.07470*, 2020.

[13] Shreeharsh Kelkar. The elite's last stand: negotiating toughness and fairness in the iit-jee, 1990–2005, 2013.

[14] Jon Kleinberg, Jens Ludwig, Sendhil Mullainathan, and Ashesh Rambachan. Algorithmic fairness. In *Area papers and proceedings*, volume 108, pages 22–27, 2018.

[15] Kasper Green Larsen. Constructive discrepancy minimization with hereditary l2 guarantees. *arXiv preprint arXiv:1711.02860*, 2017.

[16] Shachar Lovett and Raghu Meka. Constructive discrepancy minimization by walking on the edges. *SIAM Journal on Computing*, 44(5):1573–1582, 2015.

[17] Ninareh Mehrabi, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan. A survey on bias and fairness in machine learning. *arXiv preprint arXiv:1908.09635*, 2019.

[18] Aditya Krishna Menon and Robert C Williamson. The cost of fairness in classification. *arXiv preprint arXiv:1705.09055*, 2017.

[19] Bernhard Scholkopf and Alexander J Smola. *Learning with kernels: support vector machines, regularization, optimization, and beyond.* Adaptive Computation and Machine Learning series, 2018.

[20] Xinyue Shen, Steven Diamond, Yuantao Gu, and Stephen Boyd. Disciplined convex-concave programming. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, pages 1009–1014. IEEE, 2016.

[21] Joel Spencer. Six standard deviations suffice. *Transactions of the American mathematical society*, 289(2):679–706, 1985.

[22] Terence Tao. The erdos discrepancy problem. *arXiv preprint arXiv:1509.05363*, 2015.

[23] Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez-Rodriguez, and Krishna P Gummadi. Fairness constraints: A flexible approach for fair classification. *J. Mach. Learn. Res.*, 20(75):1–42, 2019.