Minimax Group Fairness: Implementation and Extensions

Karan Sampath, Rohan Gupta, Campbell Phalen and Brian Williams

CIS 523 Final Project

1 Introduction

Machine Learning paradigms today are increasingly having to grapple with understanding group errors and dealing with their potential ramifications in any model. Not only are there empirical model performance issues, but they also have long-lasting impacts on different populations. Existing research [8] has shown that different data sources often help with building in fairness improvements, but reach an upper bound in error rates very quickly. To deal with the issue, they have often aimed toward equalizing group error rates to minimize group bias. However, as pointed out in [4], this not only leads to an artificial increase in error for other groups but also pushes overall error higher counteracting optimizations made by the model.

As first expounded on in [4] (our reference document for this paper), we instead aim to use minimax group fairness, which by definition Pareto dominates any model aiming to equalize error rates. Crucially, we aim to do the following throughout the course of this paper:

1. **Algorithmically Define Better Groups**: Current literature has mostly aimed to define groups through easily perceived and hand-drawn differences, replacing actual distinctions with perceived heuristics. This approach is flawed for three reasons. First, it is extremely subjective, and entirely controlling for individual biases is not possible. Second, it is often a crude approximation of real group differences and therefore leads to inaccuracies both in error calculation and final estimation. Finally, it does not deal with intersectional harms appropriately since it relies on singular, disjoint group definitions without accounting for overlapping members of multiple affected groups. Instead, we define a better adaptive heuristic to find groups that at least partially account for the flaws listed above.
2. **Use Non-Convex Approaches**: Our reference document primarily uses a linear regression model to allow for easy and efficient computation of $\arg\min_h \epsilon(h)$. However, this approach is not easily generalizable: linear regression is not an industry standard, and even when used is done in a regularized manner. To account for this, we aim to understand generalizability by working on developing suitable non-convex parametrizations to existing group problems. We tackle the problem of the tradeoff between a guaranteed *Weighted Empirical Risk Minimizer (WERM)* as opposed to an 'approximation' model (potentially offering more model generalizability), foregoing theoretical guarantees for empirical results.
3. **Developing an Overall Framework for Group Fairness**: By comparing both non-convex and convex approaches, we aim to develop an overall framework for the best parameterizations for group fairness paradigms to help better support future research in the area.

1.1 Motivation

Our Ethical Algorithm Design class has served as the primary inspiration for this project. There was an emphasis on first understanding the real harms of incorrect Machine Learning approaches through the COMPAS v ProPublica case. Observing the clear and pernicious flaw with the model, we were motivated to further investigate existing literature to understand how we could contribute to the body of the work. Here the class supported us well by giving us the tools to understand contemporary Fair ML approaches, from

group fairness to individual fairness approaches. The Bias Bounties project here was an important practical application of our learning and gave us an insight into the differing effectiveness of various models on group fairness paradigms.

As mentioned above, we were not satisfied by the equal error rates paradigm. To the contrary, we were inspired and fascinated by the **Minimax Fairness** algorithm described in class. We looked to not only

- Implement the algorithm for our own, attempting to reproduce the paper's research on our own dataset
- Attempt extensions to the algorithm's procedure, identifying key areas for improvement.

Our project experience helped us realize the immense potential for error with hand-drawn group distinctions and the risk of drawing conclusions from such results. Further, we were interested in understanding whether we could incorporate aspects of individual and group fairness into our approach. Therefore drawing from average individual fairness, we aimed to develop an average guarantee per group for our non-convex model in this group. We prioritized this over a per iteration approach to help us develop a more generalized view of the effectiveness of non-convex models.

We hypothesize, therefore, that while our foray into non-convex optimization might risk foregoing some of the theoretical guarantees per iteration, upon output of the algorithm on non-convex cases, the tuple $(\bar{h}, \bar{\epsilon})$ would give us a randomized model that still has $\epsilon-$approximate guarantees for accuracy over groups.

Our ultimate hope is that our project serves as a natural and innovative extension of ideas touched upon in class throughout the semester, explored further in our research and research done by faculty and Prof. Kearns.

2 Related Work

There are several recent related papers to our work in the area. [10] shows the vitality of relaxing learning constraints and allow learning on transformed outcome spaces. This work is also related to the fair ML frameworks in [10] [7], where they define equity frameworks as composed of equal access, equal outcomes and equal opportunities. Importantly, they bypass existing notions of individual and group fairness to measure the number of 'obstacles' individuals face while making the decision to modify Fair ML algorithms. Further, this work is similar to our group adaptive heuristic model in two ways: it builds on the relations between equal access and equal outcomes for individuals to ensure fair modelling spans the entire pipeline, and it uses proxy models to check for various fairness attributes instead of building in modifications to the existing model framework. More importantly, both papers define various metrics of fairness: equal opportunity, equal utilization, equal outcomes, and equal access. We use such notions in our framework as well.

Our work on subgroup selection draws from work done in [6] [5] [12] [9], particularly [6] [5], with an algorithmic approach developed in both papers as well. [6] aims to deal with the identification issue by specifying a heuristic for their particular use-case. We extend their predictions to a generalized case, better allowing for the development of a heuristic model that could deal with a wide range of cases. [5] instead defines fairness to include uniformity across groups, and therefore introduces some manual class imbalance to correct for algorithmic biases. We extend work in this area by developing a generalized classifier that is able to better account for possible biases.

Minimax models are extensively present in current frameworks: [3] proposes a novel minimax framework to quantify the statistical price of fairness, but primarily applies theoretical predictions to a linear model. We aim to further extend such frameworks to the non-convex realm as well. [1] [11] also compare minimax fairness approaches to other models, with [11] proposing a 'federated' minimax approach to better improve convergence by including an alternating attribute into the model. Finally, we draw inspiration from [2] which shows the optimality of minimax models to solve a wide range of problems.

3 Methods

Our first extension to the methods described in the paper is to adaptively define groups that we want to minimax over. While we recognize the paper's efforts in defining potentially marginalized groups as per what 'makes sense', given the context of the dataset, we wish to let the data speak for itself in telling us which groups are underperforming in algorithmic analysis as well as in representation among the data items.

Specifically, we follow a broad three-step procedure for identifying groups that a model class H does poorly on:

- Train an unbiased model $h^* \in \mathcal{H}$ on an unweighted dataset $\{\vec{x}, \vec{y}\}$ to find $\arg\min_{h \in \mathcal{H}} L(h(x), y)$
- Using this trained model h^*, identify those groups that are most prominently misclassified using an algorithmic tool
- Run `MinimaxFair` using the same $\mathcal{H}$, adapted from the original paper [4] (studied in class) by *Diana, Gill, Kearns et. al* to attempt to rectify error rates over these groups.

More concretely, we will describe in depth our hypothesis classes H as well as our adaptive group finding function to identify misclassified groups by some $h^* \in \mathcal{H}$.

3.1 Group Finding

For this section, we'll suppose we have some 'sufficiently trained, close to convergent' $h^* \in H$ that is representative of the entire dataset. Of course, in classical Machine Learning, one would stop here. Once you have a tuned model that you have trained for enough epochs, what is there to do?

However, we're exploring the realm of Fair ML, so a natural question is: how well does this represent the original dataset? Since the quality of our model is implicitly conditioned on the quality of our dataset, we ask no questions on the representativeness of our dataset to actual populations, nor do we question other factors about the dataset such as size, parameters, and scope.

With our assumptions laid out, we note two key observations:

- We care about the quality of our **mislabels**, not our correct labels. Since by assumption h^* is as good as it can be within H, we want to know where h^* is underperforming.
- Specifically, we care most about distributions within groups. We want to see, for a specific group label, is the distribution of values *sufficiently close* to that of the original dataset? For example, if in the original dataset we find that 20% of the population is deaf (a binary quality), but in the **misclassified** samples, 30% of the population is deaf, we know that our model is performing poorly on deaf individuals and we need to recalibrate.

Keeping these observations in mind, we define the following procedure named `GroupFinder` (formalized in an algorithm available in the Appendix)

- Precompute distributions of group values in the original dataset
- Using h^* as defined above, find those $\hat{y} \neq y$ and corresponding $X^{'}$ that are wrongly classified.
- For the corresponding features, find the distribution of group values, per group.
- Compare this distribution to that of the wrongly classified features $(X^{'}, \hat{y} \neq y)$ by taking the (normalized) difference, thresholding the value, and computing the group value with the maximum difference.
- This gives (group, value) pairs that are most underperforming according to h^*
- Output the top N such pairs, sorted by distributional difference in descending order.

All together, this procedure gives us the groups to use in `MinimaxFair`.

3.2 Minimax Fairness

Ideas in this section tie very closely to our primary source [4]. We preface this section by giving credit where credit is due, but for clarity we will reiterate some key ideas from the paper. Firstly, in order to minimax fairness we use the `MinimaxFair` algorithm from the paper. We select this over the Lagrangian, gradient-ascent based `MinimaxFairRelaxed` for ease of implementation and the fact that our group finding algorithm can produce overlapping groups. In particular, the algorithm is of the following form:

Algorithm 1: MinimaxFair

Input: $\{x_i, y_i\}_{i=1}^n$, adaptive learning rate η_t, populations G_k with relative sizes $p_k = \frac{|G_k|}{n}$, iteration count T, loss function L, model class H;

Let $\epsilon_k(h) := \frac{1}{|G_k|} \sum_{(x,y) \in G_k} L(h(x), y)$;

Initialize $\lambda_k := p_k \; \forall k$;

for $t = 1$ ***to*** T **do**

  Find $h_t := \text{argmin}_{h \in H} \sum_k \lambda_k \cdot \epsilon_k(h)$;

  Update each $\lambda_k := \lambda_k \cdot exp(\eta_t \cdot \epsilon_k(h_t))$;

end

Output: Uniform distribution over the set of models $h_1, ..., h_T$

Figure 1: `MinimaxFair`, [4]

Our implementation is practically identical, however we did face some uncertainty in the choice of the learning rate η. we tried setting η to be a constant (specifically, $\max(\frac{\log(N)}{T}, 0.01)$, but we found it better to define $\eta_t = \dfrac{1}{\sqrt{t}}$ as suggested in the source.

Our choice of T was highly dependent on the constraints of the machine we were running on. We noticed that we could see convergent trends for most $\mathcal{H}$ in about 300-400 steps, so we give a generous estimate for T at 1000. Our purpose in this exploration is not to actually find a fully convergent model, but simply estimate whether theoretical guarantees of the algorithm hold without a guaranteed *WERM* and on differently defined, overlapping groups.

Importantly, as we will elucidate further below, our tuning and manual adjustments were most prominent when it came to designing and defining models and hypothesis classes $h \in \mathcal{H}$.

3.3 Model Selection

One of the goals in selecting this project was to explore how theory stacks up with empirical performance. In particular, we look at reasonably non-convex optimisation settings and ask two questions:

- Do we obtain a final result $\bar{h} \sim \mathcal{U}(h_1, h_2 \cdots h_T)$ resembling an $\epsilon-$approx equilibrium?
- On the granularity of a per-iteration level, do we find noise in error rates?

To that end, our first model is **Linear Regression on Log Loss**. It is well known that Weighted Linear Regression is convex, and this will serve as a 'control' for our experiments, to ensure that our results are similar to those in the source paper [4] as well as validate correctness of our algorithms.

The second model we use is **Logistic Regression on 0/1 Loss**. With logistic regression, 0/1 loss is non-convex. We selected this combination of hypothesis and loss $(\mathcal{H}, \mathcal{L})$ in order to deviate somewhat far from Linear Regression but also have a reasonably convergent approximation to weighted regression. We set the max number of training iterations as 100 for computational complexity reasons.

The third model we use is the **Gradient Boosting Classifier on 0/1 Loss**. This is out of the realm of linear classifiers, but accepts weighted samples without trouble. We use this since it is increasingly being

popularized in the worlds of competitive machine learning on Kaggle, and its empirical performance has been outstanding. We wish to compare this more 'complex' model class against the simpler ones above.

3.4 Dataset Selection

We use an American Community Survey (ACS) 1-year dataset from 2018 to help test our hypotheses. It is provided by the US Census Bureau on both a 1 and 5-year time horizon. The dataset is particularly suited to our needs in a few ways. First, it is easily accessible and verifiable. We use the Python Folktables package to access the data, and hence there is both a high degree of certainty with regards to its accuracy and lesser need to transform the dataset to our needs. There is also a high degree of familiarity with this dataset given earlier work with it in our CIS 523 class.

Second, the ACS is a survey which approximates data to the entire population using a representative sample size. This means that given the inherent issues with estimation, there are likely to be inaccuracies and biases in the dataset. We choose a 1-year version here to ensure such inaccuracies can be accounted for by our model and do not become undetectable when combined when many years.

Finally, given that the ACS contains data across a number of important group-defining indicators such as race, nativity and sex, it is ideal for testing our hypotheses on the effectiveness of our group fairness model.

4 Empirical Analysis

We implement the algorithms discussed above in code, and run them on the ACS dataset as described. We set our group finder to output 5 groups for ease of readability and ease of observing minimax trends.

4.1 Linear Regression

For the Linear Parameterization, we observed that *GroupFinder* returns the top 5 affected (group, value) tuples as ('NATIVITY', 2.0), ('PINCP', 0.0), ('ANC', 1.0), ('RAC1P', 2.0), ('CIT', 4.0), which are (respectively), *Foreign Born, No Income, Single Ancestry, Asian, Naturalized Citizens*. We train the model using log-loss, which is provably convex. Further, for Linear Regression we see the following per iteration MSE loss curve.

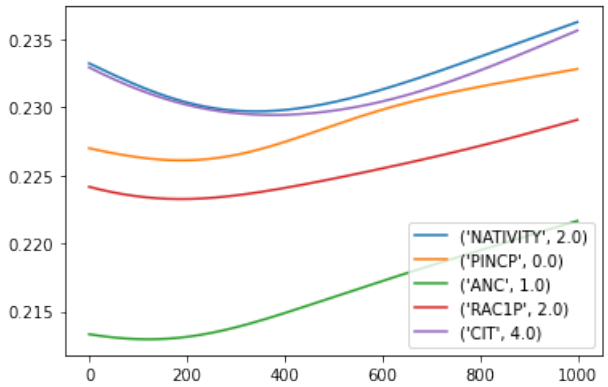


Figure 2: `Linear Regression MSE Losses`

As observed in the original implementation, about two groups tend to have the same error but other groups have lower error values on the whole. Furthermore, the relationship between errors and T (iteration) is interesting. While we observe an average global minimum around $T = 300$, errors continue to uniformly increase. We found this to be strange, since it appeared the model at time $T = 600$ was Pareto dominated (using the loss function) by the model at time $T = 300$.

This could also just be a consequence of the polynomial weights algorithm giving only guarantees about the empirical average of play, and not the per-iteration losses. However, our suspicion was that (given we used hyperparameters as alluded to in the source paper), since one of the key differing points was the fact that we did not enforce **disjoint groups**, we could have overlapping weight updates, which could afffect the algorithm's guarantees.

To validate this, we plotted MSE Losses for disjoint (group, value) tuples: ('RAC1P', 1) through ('RAC1P', 5)

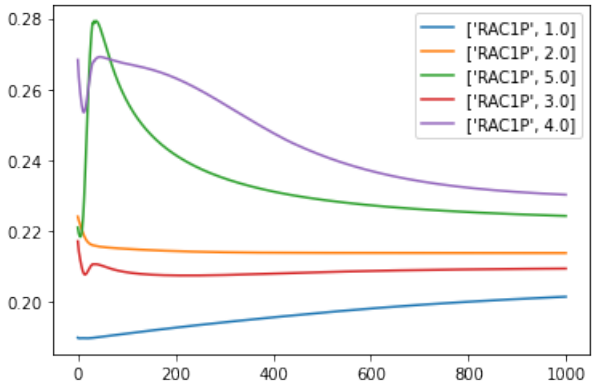


Figure 3: `Linear Regression MSE Losses for disjoint groups`

To our surprise, we found that the oddity we saw above did not repeat itself here! In fact, the algorithm performed exactly as expected, smoothing out error rates and showing convergence without any visible Pareto domination. In the original set of groups, as above, upon closer inspection we found around 90000 total entries, 48000 of which were distinct.

As a final nail in this coffin, we wondered what would happen if we artificially made the groups disjoint. That is, suppose (NATIVITY, 2.0) and (RAC1P, 2.0) had some overlap. We would arbitrarily remove group membership from one of these groups. Repeating for all pairs, we'd have a disjoint set of groups. In this way, we designed a naive algorithm to disjoint-ify our groups. We then ran `MinimaxFair` on these artificially disjoint groups and recorded our results below.

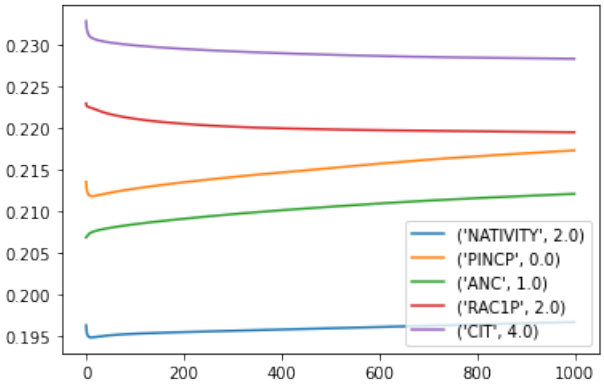


Figure 4: `Linear Regression MSE Losses for artificially disjoint groups`

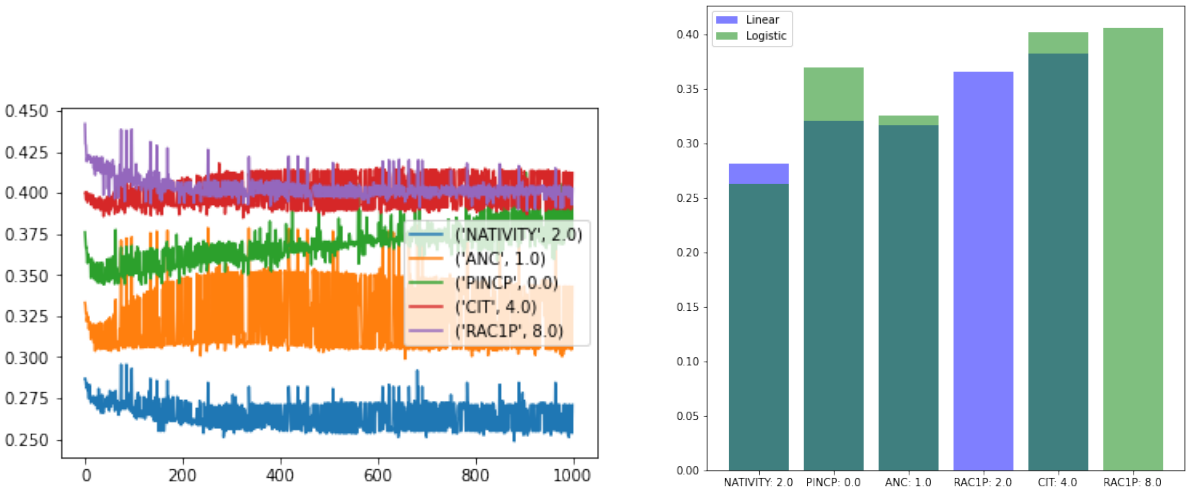
The algorithm now performed as expected 4, showing convergence and error 'trade-off', with no Pareto domination. This strongly implied, to us, that disjoint groups were an implicit requirement to the algorithm. However, this was in its own right puzzling as the source paper claimed that "The minimax approach does

not require that groups of interest be disjoint". Furthermore, an amendment to support group disjointness is made only to `MinimaxFairRelated`, so it is assumed that `MinimaxFair` has no such constraint.

We will analyse why group disjointness might affect `MinimaxFair` in detail in Section 5, but for the next few experiments, we'll artificially disjoin groups as per our algorithm above in order to provably retain guarantees of `MinimaxFair`.

4.2 Logistic Regression

For the Logistic Regression parameterisation, we train it on 0/1 loss which we know is non-convex. We see the following results.



(a) 0/1 loss over time T

(b) Group 0/1 error comparison for $\bar{h}$

Figure 5: Logistic Regression Analysis

Evidently, this curve is very noisy and shows large fluctuations in errors. However, we realize two key properties of the curve

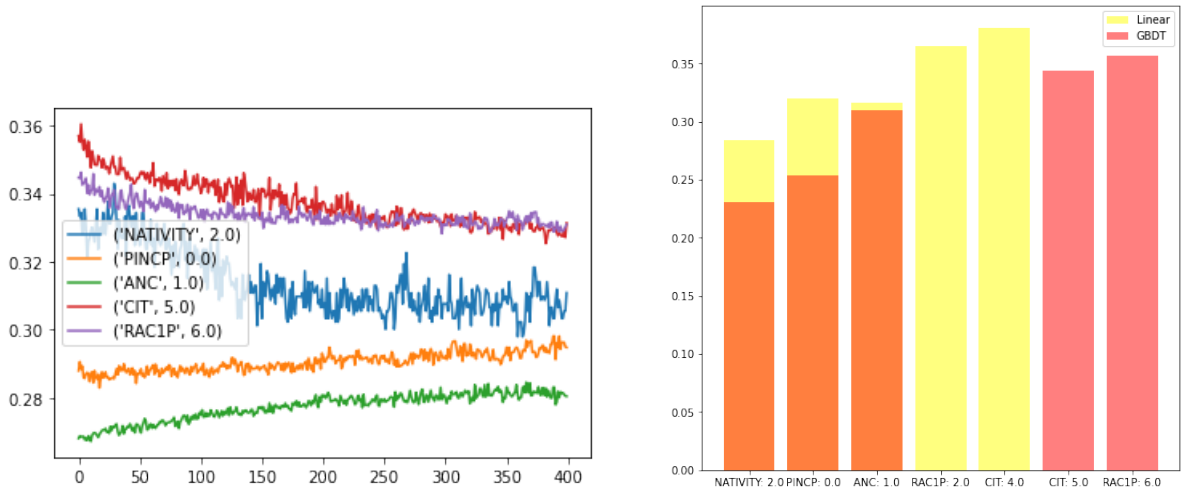
- While there is noise in per-iteration loss, there appears to be a convergent trend for all curves.
- Minimax properties are realized, where we see some error 'trade off' and no enforcement of equal error rates.

These properties lead us to hypothesise that, since at the end of model training we care about the empirical average of play $\bar{h}$ (and this is the real quantity we have good guarantees over), the performance of $\bar{h}$ shouldn't be too far from that of a model optimized over a convex loss function (such as Linear Regression). Figure 6) b) demonstrates exactly this. In our implementation, we cache the models at every round and finally give a randomized hypothesis $\bar{h}$, which is simply a uniform sampling over all models $\{h_1, h_2 \cdots h_T\}$.

It turns out, that while logistic regression on 0/1 loss is evidently non-convex and noisy on a per iteration granularity, the group error for $\bar{h}$ is largely comparable to that of Linear Regression. We observe that in Figure 5b, error rates for groups that are common between the two models are very close. Note that RAC1P = 2.0 and RAC1P = 8.0 are groups output by `GroupFinder` that are not common between the two $\mathcal{H}$, so their respective barplots have no overlap.

4.3 Gradient Boosted Decision Trees

The final model we use is a Gradient Boosted Decision Tree. We train models of depth 3 and for 100 iterations each on 0/1 Loss, which is non-convex. This was by far our most computationally complex model, and it took about 1 hour to fully run `MinimaxFair` for 400 iterations. Note that we reduced T from 1000 to 400 since it would have taken too long (and potentially overloaded resources/memory of our local machine).



(a) 0/1 loss over time T

(b) Group 0/1 error comparison for $\bar{h}$

Figure 6: GBDT Analysis

We observe similar trends as in Logistic Regression, except with less noise and more clear-cut convergence. More importantly, the empirical average of play for GBT consistently outperforms Linear parameterisations on overlapping groups. This is evidence to suggest that it might be worth exploring more complex models (like GBDTs) with non-convex cost functions over sticking to theoretical guarantees.

5 Evaluation

With these experiments, we overarchingly set out to

- **Objective 1:** Validate the procedure outlined in the source alongside an algorithmic group finding function
- **Objective 2:** Test the framework on non-convex cost functions over varying $\mathcal{H}$

Objective 1

Evidently, our models match up to the source as they show convergence, minmax properties, and improved group error rates. However, our finding with the framework malfunctioning on overlapping groups was very interesting to us. At a high level, we hypothesise this occurs due to to repeated weight updates.

Suppose at time t we have some data items d with weight $w_t(d)$ in multiple groups $G_1, G_2, \cdots$. Note that $w(d)$ is a function of each group it is in, and at time t, $w(d)$ will be updated to $w(d) + k$, where $k \geq 0$ since $e^y \geq 0$, $\forall y \in R$. Therefore every such d will have $w_{t+1}(d) >> w_t(d)$. Those items that appear in multiple groups will have a higher weight as t increases and, consequently, since we weight samples as per an implicit distribution according to w, other samples will have lower weight.

In this way, the PW algorithm will 'act as if' data items d are heavily underperforming and weight them highly. Furthermore, it's likely that these data items that consitute multiple groups are a small part of the dataset. These two results imply that for larger t, the majority of the dataset will have lower weight, so naturally we end up optimizing for accuracy over d, which is not particularly relevant and appears to come with a sacrifice of overall group error as seen in 2

With this established, we are led to wonder about how we can achieve both **overlapping groups** and **minimax fairness**. We have currently artificially solved the problem by artificially disjoining the groups; however, intersectionality is growing to be increasingly important, and it's important that this is reflected in Fair ML.

Objective 2

Modern ML is the most famous non-convex class of problems that exists. Therefore, when we saw the paper's analysis limited to convex problems, we were intrigued to see whether the results held for more general classes.

Our empirical tests revealed that, in the non-convex case, while the per-iteration loss seems to fluctuate, average guarantees hold. In particular, $\bar{h} \sim \mathcal{U}(h_1, h_2 \cdots h_T)$ appears to resemble an $\epsilon-$approx equilibrium. This follows from the assumption that the Linear Regression model resembles an ϵ-approx equilibrium, and other models (Logistic, GBDT) produce $\bar{h}$ that perform very similarly.

Notably, we believe that there is cause for future research in this area as we found our GBDT's empirical average model to do better than Linear Regression's equivalent (Fig 6b). While accounting for the fact that we trained GBDT for $T = 400$ and Linear Regression for $T = 1000$, this is even more surprising. We observe that the tradeoff between a **more complex** and a **simpler, yet convex** system can be positive!

Note. It is important to highlight that while it is certainly rosy to think about non-convex optimisation in this new light, in practical cases it may not be the most feasible. This is simply because, if it takes time O to train a model $h \in \mathcal{H}$, we have an overall running time of $\mathcal{O}(T \cdot O)$ since `MinimaxFair` calls the training step as a subroutine. The more complex the model, the larger O is, and the multiple of T might cripple feasibility for achieving great minimax guarantees.

6 Conclusion

In this paper, we have shown that convex parametrizations of minimax fairness in many cases can do better in terms of noise, convergence and empirical average of play in comparison to convex parametrizations. We provide a thorough discussion of the limits of our parametrizations particularly in terms of feasibility. Further, we show that `MinimaxFair` does not achieve similar performance outcomes when groups overlap, which is a vital issue as detailed in our introduction. While we aimed to define better artifically disjoint groups, this is clearly not easily applicable to real world scenarios where intersectionality is not a factor that can be ignored. Finally, we also show the limits of `GroupFinder` and discuss possible improvements to the algorithm.

There remains immense potential for further exploration in this area. On our group finding algorithm, we believe an immediate further extension on our artifically disjoint groups would be clearly delineate intersections and generate results on non-intersecting subsets and intersecting subsets separately. Currently, we are unable to distinguish between the two, reducing the quality of our conclusions. A natural further extension of this would be to mathematically understand what causes the erroneous behavior of overlapping groups, thereby better helping us pinpoint and potentially rectify the behavior from an algorithmic standpoint.

On our models, we believe two immediate extensions would be try other models such as Neural Networks and also increase the number of iterations T for each model. We were severely limited by the computing power available to us (training a single neural network took about 2 mins, and for $T = 400$, we were looking at over 10 hours) but we believe that a distributed version of this algorithm in the cloud could allow for a

far higher dimensional result. Finally, we've also observed that we reach upper bounds in terms of feasibility while exploring different convex models, raising important further exploration on the exact nature of the tradeoff between feasibility and accuracy.

References

[1] Blum, A., Stangl, K., and Vakilian, A. Multi stage screening: Enforcing fairness and maximizing efficiency in a pre-existing pipeline, 2022.

[2] Chen, S., Jiang, S., Ma, Z., Nolan, G. P., and Zhu, B. One-way matching of datasets with low rank signals, 2022.

[3] Chzhen, E., and Schreuder, N. A minimax framework for quantifying risk-fairness trade-off in regression. *arXiv preprint arXiv:2007.14265* (2020).

[4] Diana, E., Gill, W., Kearns, M., Kenthapadi, K., and Roth, A. Minimax group fairness: Algorithms and experiments. In *Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society* (2021), pp. 66–76.

[5] Dullerud, N., Roth, K., Hamidieh, K., Papernot, N., and Ghassemi, M. Is fairness only metric deep? evaluating and addressing subgroup gaps in deep metric learning, 2022.

[6] Liu, J., Lyu, Y., Zhang, X., and Xie, S. Subgroup fairness in graph-based spam detection, 2022.

[7] Naggita, K., and Aguma, J. C. The equity framework: Fairness beyond equalized predictive outcomes, 2022.

[8] Redmond, M., and Baveja, A. A data-driven software tool for enabling cooperative information sharing among police departments. *European Journal of Operational Research 141*, 3 (2002), 660–678.

[9] Shaham, S., Ghinita, G., and Shahabi, C. Models and mechanisms for fairness in location data processing, 2022.

[10] Shen, A., Han, X., Cohn, T., Baldwin, T., and Frermann, L. Optimising equal opportunity fairness in model training, 2022.

[11] Tarzanagh, D. A., Li, M., Thrampoulidis, C., and Oymak, S. Fednest: Federated bilevel, minimax, and compositional optimization, 2022.

[12] Zhu, H., and Wang, S. Learning fair models without sensitive attributes: A generative approach, 2022.

A Appendix: Code

All our code is available on GitHub at `https://github.com/rohangpta/523-final-project`

Specifically, you can find the precise algorithms, helper methods, and test functions we used in `algos.py` and you can run the notebook `experiments.ipynb` to recreate/modify our experiments from scratch.