A Group-sensitive Optimal/Bayesian Classifier

Consider our group-wise exponential loss as:

$$
\begin{aligned}
\mathcal{L}(H_T; \{\mathbb{D}^s\}, \{\tau^s\}) &= \tau^{p+}L^{p+} + \tau^{p-}L^{p-} \\
&\quad + \tau^{np+}L^{np+} + \tau^{np-}L^{np-} \\
&= \sum_s \sum_{i=1}^{|D|} \frac{\tau_s}{|D^s|} \mathbb{I}(x_i \in D^s \& y_i = 1) e^{-H(x_i)} \\
&\quad + \sum_s \sum_{i=1}^{|D|} \frac{\tau_s}{|D^s|} \mathbb{I}(x_i \in D^s \& y_i = -1) e^{H(x_i)}
\end{aligned}
$$

(A.1)

The ensemble classifier H_T is optimal if and only if $\frac{\partial \mathcal{L}(H_T; \{\mathbb{D}^s\}, \{\tau^s\}.)}{\partial H_T} = 0$, we have:

$$
\begin{aligned}
&\sum_{i=1}^{|D|} [\frac{\tau^{p+}}{|D^{p+}|} \mathbb{I}(x_i \in D^{p+}) + \frac{\tau^{np+}}{|D^{np+}|} \mathbb{I}(x_i \in D^{np+})] e^{H(x_i)} \\
&= \sum_{i=1}^{|D|} [\frac{\tau^{p-}}{|D^{p-}|} \mathbb{I}(x_i \in D^{p-}) + \frac{\tau^{np-}}{|D^{np-}|} \mathbb{I}(x_i \in D^{np-})] e^{-H(x_i)}
\end{aligned}
$$

(A.2)

Assume the ratio $\sum_{i=1}^{|D|} \frac{\mathbb{I}(x_i \in D^s)}{|D|}$ can be the approximation of the real group probability $p(x \in D^s|x)$, then:

$$
H(x) = \frac{1}{2} \ln \frac{\frac{\tau^{p+}}{|D^{p+}|} p(x \in D^{p+}|x) + \frac{\tau^{np+}}{|D^{np+}|} p(x \in D^{np+}|x)}{\frac{\tau^{p-}}{|D^{p-}|} p(x \in D^{p-}|x) + \frac{\tau^{np-}}{|D^{np-}|} p(x \in D^{np-}|x)}
$$

(A.3)

It shows that the sensitivity of the optimal classifier with respect to group s is depending on the $\frac{\tau_s}{D^s}$, where τ_s is the initial group weight and $|D^s|$ is the sample size for group s.

B Effectiveness of Penalty Intensity Selection

Fig. 1 shows that if just solve the optimization problem with fairness constraint by adding a constant Lagrange multiplier λ at each boosting iteration, the model will collapse at the very beginning, where the model will be stuck in a local optimal point, characterizing as oscillations around a certain point. Therefore, although the grid search for λ cost more time, the algorithm will collapse easily without this operation.

C Details for Datasets and Competors

We introduce the details of the datasets we use as follows.

- **Bank** dataset is to decide if a person subscribes to the financial product. We set the maritial status (married or single) as the sensitive attribution for this dataset.

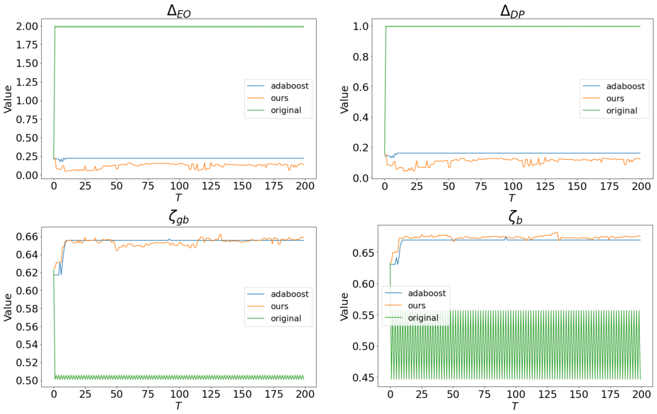

Figure 1: Performance and fairness metrics for every boosting iteration via Compas dataset for use rectified fairness penalty and penalty intensity selection. In these figures, 'original' denotes that just activate the fairness constraint at every boosting with a constant Lagrange multiplier λ, which may cause the model stuck in a local optimal point showing as the lines fluctuating back and forth. Here T means the boosting iteration.

- **Kdd census income** dataset (KDD) was collected from Current Population Surveys conducted by the U.S. Census Bureau from 1994 to 1995. The label of this dataset is if a person receives more than 50,000 US dollars annually or not. In our experiments, we consider the gender as the sensitive attribution.

- **Compas** dataset was published by ProPublica14 in 2016. The data have been used for crime recidivism risk prediction by 52 attributes (31 categorical, 6 binary, 14 numerical and a null attribute). We consider recidivism as the positive class and gender as the sensitive attribution.

- **Credit** dataset was quite similar to the bank dataset, which aims to determine whether it is risky to grant credit to a person or not. We use the gender value as the sensitive attribution.

- **Adult** dataset contains data from different demographics in the United States of America and the label of each sample is defined by whether the annual income of exceed 50K dollars or not. The sensitive attribution of this dataset in our experiments is gender.

Details of these datasets are shown in Table 1.

We introduce the details of the competitors we use as follows.

- **AdaBoost** It is the vanilla AdaBoost that does not consider fairness issue. We set the number of estimators as 100 in our experiments.

	$\vert D^{p+}\vert$	$\vert D^{p-}\vert$	$\vert D^{np+}\vert$	$\vert D^{np-}\vert$	IMG
Credit	2823	9015	3763	14349	0.2622
Kdd	3968	151807	14600	128910	0.0243
Bank	2755	24459	1912	10878	0.0781
Compas	373	658	2110	2137	0.1745
Adult	1669	13011	9533	20962	0.0796

Table 1: Current benchmarks for fairness. $|D^{p+}|$, $|D^{p-}|$, $|D^{np+}|$, $|D^{np-}|$ denote the number of samples for positive protected group, negative protected group, positive non-protected group and negative non-protected group, respectively.

- **GBDT** Similar to AdaBoost, GBDT trains many base learners in a gradual, additive and sequential manner, where each base learner are supposed to fit the gradient generated by the forehead ensemble classifier.
- **AdaFair**: It is a fairness-aware classifier based on AdaBoost that updates the weights of the instances heuristically in each boosting round taking into account a cumulative notion of fairness metric.
- **SMOTEBoosting**: It is a modified version for ababoost dealing with the class imbalance data applying SMOTE to each boosting iteration
- **Group DRO**: This work applies distributionally robust optimization with a increased regularization to neural network, which can increase the performance of the worst-case group. In this paper, we use a two-block neural network as the classifier for Group DRO with 80 training epochs. Structural details can be found in the codes we plan to release in the future.
- **FTL**: In this work, researchers propose a compound splitting criterion which combines threshold-free (i.e., strong) demographic parity with ROC-AUC, and it can be extends to bagged fair random forest. In our experiments, we set the number of base learners of this fair random forest as 100.
- **ExpGradient**: It is a post-processing method that reduce fair classification problem to a sequence of cost-sensitive classification problems, and then yield a randomized classifier with the lowest error subject to the designed constraints.
- **FairBatch**: FairBatch is a recent work achieving SOTA performance on fairness for deep neural network by adjusting the sample ratio of each group per batch as a min-max optimization problem. As same as GroupDRO, we train a two-block neural network as the classifier.