Boosting for Fairness-Aware Classification

Daniil Babin [1] **Sudarut Kasemsuk** [1] **Waralak Pariwatphan** [1]

Abstract

Fairness-aware classification is widely used throughout the world and serves a significant purpose in decision-making such as loan credit and/or job hiring. To reduce discrimination among minorities(e.g. in terms of race, gender or nationality). In this problem, We use real-world datasets(UCI KDD Census, etc) and define protect/non-protect groups base on sensitive attributes. We propose different ML's algorithms for imbalanced classification(Adafair, SMOTEBoost).

Github repo: github repo
Video presentation: presentation ink

1. Introduction

Data prediction is based on mathematical models that are used to construct algorithms based on machine learning to forecast accuracy results using training data sets. This is a widely utilized strategy in the banking and financial industries nowadays. online selling platform insurance industry, and many others, in order to predict the outcomes and provide solutions to problems that must be recognized as precisely as possible.

Imbalanced data means that data is contains more data in one category than the number of data in the other. This is generally detected due to the dataset's nature, such as In comparison to non-indebted clients, the number of indebted customers is much lower. or some limitations that make the data unbalanced. Data predictions are more biased towards larger datasets than smaller datasets in general. It has an impact on a minority class; for example, if a dataset comprises 30 percent Group A and 70 percent Group B, a general classification approach is likely to make a difference. The new member is forecast to be in Group B since it has a 70 percent chance of being correct, which means it has a high accuracy (Accuracy score), but it will be unable to forecast Group A. The minority group tended to be more important than the majority group; for example, the diabetics projected group was smaller than the non-diabetics group. And the patient's treatment may be impacted as a result of the miscalculation. Thus, there was a low TPR and a high TNR rate. As a result, Imbalanced classification Algorithms can help resolve data imbalances are important.

AdaFair, our proposed solution, solves this problem and provides fairness while maintaining high TPRs and TNRs for both groups. AdaFair is created on the basis of AdaBoost. AdaFair is a fairness-aware boosting strategy that maintains good TPRs and TNRs while attaining parity between the two groups (thus ensuring justice). In a range of values from 8 percent to 25 percent, AdaFair outperforms approaches in terms of performance (depending on the severity of class-imbalance). Fairness succeeds over a non-cumulative alternative. AdaFair forecasts are more confident when confidence is considered into the weighting.(Iosifidis & Ntoutsi, 2019) In additional, the SMOTEBoost method was also utilized to tackle the problem. The Synthetic Minority Oversampling Technique (SMOTE) and the regular boosting procedure are combined in the SMOTEBoost algorithm. We want to use SMOTE to improve minority class prediction, and we want to use boosting to avoid sacrificing accuracy all over the full data set.(Chawla et al., 2003)

The main contributions of this report are as follows.

- Reproducing algorithms and test models for fairness-aware classification.
- Evaluate and compare those models in terms of predictive and fairness performance on the listed datasets.
- Provide the interpretation of the showed results, how fairly balancing the datasets affects the overall model.

The datasets used in all of the studies are real-world datasets (Adult Census, Bank, COMPAS, KDD Census).

[1]Skolkovo Institute of Science and Technology, Moscow, Russia. Correspondence to: Daniil Babin <Daniil.Babin@skoltech.ru>, Sudarut Kasemsuk <Sudarut.kasemsuk@skoltech.ru>, Waralak Pariwatphan <Waralak.Pariwatphan@skoltech.ru>.

Final Projects of the Machine Learning 2020 Course, Skoltech, Moscow, Russian Federation, 2020.

2. Related Work

To eliminate latent discrimination, On the basis of training data, classification models frequently make predictions. If the training data favors one group or item over another, the learnt model will behave in a discriminatory manner toward that group or object. Some of the learning model's attitudes could lead to skewed findings. To minimize existing discrimination, the discrimination data categorization model has undergone minimal changes. As a result, a function that learns from imbalanced data is required. leaven It is possible to learn non-discriminatory models. Because the model learns from non-discriminatory data, future classifications are less likely to contain behavioral bias. This concept is referred to as a classification without discriminating (CND) (Kamiran & Calders, 2009). However, the alterations should be kept to a minimum and as unobtrusive as feasible. The tradeoff between correctness and indiscrimination is minimal.

Disparate mistreatment(Zafar et al., 2017). Intuitively, any automated decision-making system whose outputs (or decisions) aren't perfectly (i.e., 100 percent) accurate can lead to differential treatments. Consider a decision-making system that employs a logistic regression classifier to generate binary outputs (for example, positive and negative) on a group of people. The system will misclassify (i.e., produce false positives, false negatives, or both) some people if the items in the training data with positive and negative class labels are not linearly separable, as is often the case in many real-world application settings. Misclassification rates may differ for groups of people with varied values of sensitive traits (e.g., males and females; blacks and whites) in this scenario, resulting in unequal mistreatment. For decision tree learning, a discrimination-aware classification strategy improves on prior methods that were based on dataset cleaning (or so-called Massaging)(Kamiran et al., 2012). Reweighing selects a biased sample to negate the impact of discrimination, whereas Massaging modifies the class labels of selected items in the training data to generate a discrimination-free dataset. Model selection strategies exist in addition to preprocessing techniques. Three methods for making Naive Bayes classifiers discrimination-free are proposed.

AdaBoost is a continuous development technique that uses boosting techniques. There's a study paper on its use in machine learning disciplines like Face Detection, which is an example of an Adaboost experiment using a Decision tree model. This is a massively popular model. Because it is adaptive to all learning algorithms and employs the iteration technique. and adjust the weight, decreasing Bias Error and preventing excessive overfitting. AdaBoost's key goal is to transform a weak model into a strong one. The same data collection is processed and analyzed. Use the same model as before. Begin with a simple classify to notice error values before adjusting weight in the data group and continuing to classify until a strong model with the best accuracy emerges.

An alternative sampling method has been proposed, which involves producing synthetic samples rather than adding them to the combination. and label this strategy after the initial data of a random minority group With synthesis, sampling increases the minority (Synthetic Minority Oversampling Technique: SMOTE)(Chawla et al., 2002). On the linking any minority sample, SMOTE generates a new minority sample at random. and one of the closest nearby groups of k by comparing the methods ROS and SMOTE utilizing classification decision trees, as well as a randomized experiment at the default size levels of 100, 300, 400, and 500, which were compared using various data sets. The accuracy of minor forecast is excellent. SMOTE is a widely used acronym. To increase the accuracy of prediction In data disturbed settings, it was also more powerful than random reduction. Unbalanced re-sampling under information conditions performed better in a comparison trial.

Equal opportunity(Hardt et al., 2016), depending on the percentage difference between two groups' true positive percentages. Although it has been suggested that equitable opportunity (also known as disparate mistreatment) has potential. In all classes, equalized odds extends equal opportunity by taking into account the difference in genuine categorized instances between the protected and non-protected groups. Although there are other fairness concepts, we use equalized odds in this study since it appears to be the most promising and has been embraced by recent state-of-the-art methods.

3. Basic Concepts and Definition

Dataset D, which is made up of the following joint distributions: $P(F, S, y) : S$ stands for sensitive characteristics such as age, gender, and marital status, F stands for non-sensitive characteristics, and y stands for the class label. Just for simplicity, we assume that the classification problem is binary $y \in \{+, -\}$, and that there is only one sensitive characteristic S, which is also binary. Class are denoted by the notations $s_+ \left(s_-\right), \bar{s}_+ \left(\bar{s}_-\right)$. Classification's purpose is to find a mapping function $f : (F, S) \rightarrow y$ that can predict the class labels of future unseen instances.

As a fairness metric, we use Equalized Odds. The difference in prediction errors between the protected and non-protected groups is measured by Eq.Odds. Let $\delta FPR(\delta FNR)$ be the difference in false positive (false negative) rates between the protected and non-protected groups, which is defined as follows ($\hat{y}$ is the predicted label):

$$\begin{aligned} \delta FPR &= P\left(y \neq \hat{y} \mid \bar{s}_-\right) - P\left(y \neq \hat{y} \mid s_-\right) \delta FNR \\ &= P\left(y \neq \hat{y} \mid \bar{s}_+\right) - P\left(y \neq \hat{y} \mid s_+\right) \end{aligned} \tag{1}$$

Eq.Odds is a method for minimizing both differences:

$$Eq.Odds = |\delta FPR| + |\delta FNR| \tag{2}$$

In the context of fairness, predictive performance is typically measured in terms of errors rate, which is defined as:

$$ER = \frac{FN + FP}{TP + TN + FN + FP} \tag{3}$$

When there is a class imbalance, however, optimizing for error rate is difficult. In this example, the classifier may misclassify the majority of the minority cases while properly categorizing the majority, resulting in a low error rate ER despite the poor performance in the minority class. In terms of fairness, such a classifier may still be fair, i.e., Eq.Odds, because the difference between the FPRs will result in low FNRs for each group.

The error rate (which is not an acceptable performance measure in an imbalance situation) should be replaced by the balanced error rate, which is defined as follows:

$$\begin{aligned} BER &= 1 - \frac{1}{2} \cdot (\frac{TP}{TP + FN} + \frac{TN}{TN + FP}) \\ &= 1 - \frac{1}{2} \cdot (TPR + TNR) \end{aligned} \tag{4}$$

4. Algorithms and Models

4.1. Adafair

AdaFair(Iosifidis & Ntoutsi, 2019) is a fairness-aware boosting ensemble that updates the distribution of the data at each round, considering not only the class mistakes but also the model's fairness-related performance, It is built on AdaBoost's basis. Boosting is an ensemble strategy in which weak learners are linked to produce a strong learner.AdaBoost(E Schapire, 1999) iteratively calls a weak learner, modifying instance weights in each cycle (boosting round) based on misclassified examples. Because it divides the learning problem into several sub-problems and then combines their solutions (sub-models) into an overall (global) model, we believe boosting is a viable strategy for fairness-aware classification. Intuitively, solving the fairness problem in simpler sub-models is easier than solving it in a global complicated model. To use AdaBoost for fairness, one must carefully alter the underlying data distribution between rounds such that both predicted performance and fairness considerations are taken into account.

4.1.1. Cumulative fairness costs

Let the number of boosting rounds is indicated by T and $1 - T$ be the current boosting round. Let$H_{1:j}(x) = \sum_{i=1}^{j} a_i h_i(x)$be the ensemble model up to round$j$. The model at round$j$ is defined in terms of$|\delta FPR|, |\delta FNR|$ for both protected and non-protected groups, as follows:

$$\begin{aligned} \delta FNR^{1:j} = &\frac{\sum_{i=1}^{\bar{s}_+} 1 \cdot I[\sum_{k=1}^{j} a_k h_k(x_i^{\bar{s}_+}) \neq y_i]}{\bar{s}_+} \\ - &\frac{\sum_{i=1}^{s_+} 1 \cdot I[\sum_{k=1}^{j} a_k h_k(x_i^{s_+}) \neq y_i]}{s_+} \end{aligned} \tag{5}$$

$$\begin{aligned} \delta FPR^{1:j} = &\frac{\sum_{i=1}^{\bar{s}_-} 1 \cdot I[\sum_{k=1}^{j} a_k h_k(x_i^{\bar{s}_-}) \neq y_i]}{\bar{s}_-} \\ - &\frac{\sum_{i=1}^{s_-} 1 \cdot I[\sum_{k=1}^{j} a_k h_k(x_i^{s_-}) \neq y_i]}{s_-} \end{aligned} \tag{6}$$

To avoid misunderstanding, we refer to our fairness-related extra weights as costs. In each round, the expenses associated with fairness are constantly assessed. The following is how the fairness-related cost u_i for an instance x_i in the boosting round j is calculated as follows:

$$u_i = \begin{cases} \left|\delta FNR^{1:j}\right|, \text{if} I\left((y_i \neq h_j(x_i)) \wedge \left|\delta FNR^{1:j}\right| > \epsilon\right), x_i \in s_+, \\ \delta FNR^{1:j} > 0 \\ \left|\delta FNR^{1:j}\right|, \text{if} I\left((y_i \neq h_j(x_i)) \wedge \left|\delta FNR^{1:j}\right| > \epsilon\right), x_i \in \bar{s}_+, \\ \delta FNR^{1:j} < 0 \\ \left|\delta FPR^{1:j}\right|, \text{if} I\left((y_i \neq h_j(x_i)) \wedge \left|\delta FPR^{1:j}\right| > \epsilon\right), x_i \in s_-, \\ \delta FPR^{1:j} > 0 \\ \left|\delta FPR^{1:j}\right|, \text{if} I\left((y_i \neq h_j(x_i)) \wedge \left|\delta FPR^{1:j}\right| > \epsilon\right), x_i \in \bar{s}_-, \\ \delta FPR^{1:j} < 0 \\ 0, \text{ otherwise} \end{cases} \tag{7}$$

4.1.2. AdaFair Algorithm

AdaFair's training phase is depicted in Algorithm 1 (Iosifidis & Ntoutsi, 2019). The instance weights w and expenses u are set up at the start. The error rate $\alpha_j, \delta FNR^{1:j}, \delta FPR^{1:j}$ and u_i are computed for the current round while a weak learner is taught on a particular weight distribution. Instance weights are estimated and normalized by a factor Z once u is determined. We use the confidence score to contribute to the re-weighting procedure, which is different from the original AdaBoost. We allocate higher weights to misclassified examples that are more difficult to learn based on the confidence score than to instances that are close to the decision boundary.

4.2. SMOTEBoost

4.2.1. SMOTEBoost Algorithm

After learning a classifier at iteration t, the synthetically manufactured minority class examples are eliminated. That

Algorithm 1 AdaFair

Input: $D = (x_i, y_i)_1^N, T, \epsilon$
Initialize $w_i = 1/N$ and $u_i = 0$, for $i = 1, 2, ..., N$
for $j = 1$ **to** T **do**
(a) Train a classifier h_j to the training data using weights w_i.
(b) Compute the error rate $err_j = \frac{\sum_{i=1}^{N} w_i I(y_i \neq h_j(x_i))}{\sum_{i=1}^{N} w_i}$
(c) Compute the weight $\alpha_j = \frac{1}{2} \cdot log(\frac{1-err_j}{err_j})$
(d) Compute fairness-related δFNR$^{1:j}$
(e) Compute fairness-related δFPR$^{1:j}$
(f) Compute fairness-related costs u_i
(g) Update the distribution as
$w_i \leftarrow \frac{1}{Z_j} w_i \cdot e^{\alpha_j \cdot \hat{h_j}(x) I(y_i \neq h_j(x_i))} \cdot (1 + u_i)$
//Z_j is normalization factor; $\hat{h_j}$ is the confidence score
end for
Output: $H(x) = \sum_{j=1}^{T} \alpha h_j(x)$

Algorithm 2 SMOTEBoost

Given: Set $S(x_1, y_1), .., (x_m, y_m) x_i \in X$,
with label in $y_i \in Y = 1, ..., C$, where $C_p, (C_p < C)$ corresponds to a minority (positive) class
Let $B = (i, j) : i = 1, ..., m, y \neq y_i$
Initialize the distribution D_1 over the examples, such that $D_1(i) = 1/m$
for $i = 1$ **to** T **do**
(a) Modify distribution D_t by creating N synthetic examples from minority class C_p using the SMOTE algorithm
(b) Train a weak learner using distribution D_t
(c) Compute weak hypothesis $h_t = X \times Y \rightarrow [0, 1]$
(d) Compute the pseudo-loss of hypothesis $h_t : \epsilon_t = \sum_{(i,j) \in B} D_t(i, y)(1 - h_t(x_i, y_i) + h_t(x_i, y))$
(e) Set $\beta_t = \epsilon_t/(1 - \epsilon_t)$ and $w_t = (1/2) \cdot (1 - h_t(x_i, y) + (h_t(x_i, y_i))$
(f) Update $D_t : D_{t+1}(i, y) = (D_t(i, y)/Z_t) \cdot \beta_t^{w_t}$
where Z_t is a normalization constant chosen such that D_{t+1} is a distribution
end for
Output: $h_{fn} = argmax_{y \in Y} \sum_{t=1}^{T} (log \frac{1}{\beta_t}) \cdot h_t(x, y)$

is, they are not included to the original training set, and in each iteration t, new examples are produced by sampling from D_t. The original training set is used to calculate the error estimate after each boosting step.

The combination of SMOTE and a variation of the AdaBoost.M2 boosting technique(Freund & Schapire, 1996). The suggested SMOTEBoost algorithm works in a T -round cycle. Every round, a weak learning algorithm is called and a new distribution D_t is supplied, which is adjusted by prioritizing specific training cases. The weights are adjusted in the distribution to give incorrect classifications a higher weight than accurate categories. Unlike ordinary boosting, which updates the distribution D_t evenly for both majority and minority class examples, the SMOTEBoost technique updates the distribution D_t so that the minority class examples are oversampled by constructing synthetic minority class cases. To compute the weak hypothesis h_t, the weak learner is given the whole weighted training set. Finally, the many hypotheses are integrated into a single final hypothesis, h_{fn}.

5. Experiments and Results

In order to evaluate the performance of fairness-aware predictive models, we compare the fairness-aware algorithms (AdaFair and SMOTEBoost) with AdaBoost and report the model accuracy on accuracy and balance accuracy. Moreover, the fairness of models is indicated on equalized odds, TPR and TNR values for both protected and non-protected groups.

5.1. Experiment setup and dataset description

We evaluate our approach on four real-world datasets which summarized their characteristics in Table 1. As can be shown, and thus provide an intriguing standard for evaluation. The original datasets are available on GitHub.

Adult census income dataset(Kohavi Becker, 2017) contains demographic data from the United States, and the goal of prediction is to determine whether a person's annual income exceeds 50K. The sensitive attribute is set to sex, and the protected class is female. We clean the data by removing a missing value and an ineffective feature. The get dummies method is used to encode all categorical features, and standard scaler is used to normalize all numeric features.

Bank dataset(Cortez Moro, 2012) contains direct marketing data from a Portuguese banking institution, and the goal of prediction is to determine whether or not the customer subscribed to the bank term deposit. The sensitive attribute is set to marital status, and the protected class is married. To clean the data, the get dummies method was used to encode all categorical features. The standard scaler is used to normalize all numerical features.

Compass dataset(Larson et al., 2017) includes criminal history, jail and prison time, demographics, and compas risk scores. The goal is to determine whether or not a person will be arrested again in the next two years (recidivism). The sensitive attribute is set to sex, and the protected class is female. To clean the data, we removed ineffective features

Table 1. An overview of the datasets.

Characteristics	Adult	Bank	Compas	KDD
Instances	32,561	49,732	7,214	199,523
Attributes	15	17	53	42
Sensitive Attr.	Gender	Marital	Gender	Gender
Class ratio (+:-)	1:2.02	1:1.52	1:4.17	1.09:1
Positive class	50K+	yes	1	50000+

and those with a high number of missing values. The get dummies method was used to encode all categorical features, and standard scaler is used to normalize all numeric features and fill in the missing value as 0.

KDD dataset(Lane Kohavi, 2000) and the adult census dataset have the same predictive target. In the KDD census, the class designations were obtained first from the total person income field rather than the adjusted gross income. The sensitive attribute is set to sex, and the protected class is female. To clean the data, we substitute an indicator for missing values in categorical features. The get dummies method was used to encode all categorical features, and standard scaler is used to normalize all numeric features.

We perform all dataset experiments as a 70:30 random split stratified by target feature to obtain train and test data, and a random seed was set to fix reproducibility. To achieve the fairness-aware classification in the experiments, a zero tolerance for discrimination was set. For all approaches compared to AdaBoost, we set the number of boosting rounds to 10 and the maximum depth of a weak learner (Decision Tree Classifier) to 5.

5.2. Predictive and fairness performance

5.2.1. Adult Census Income

Figure 1 shows the performance of different approaches we tested on the dataset. AdaFair and SMOTEBoost outperform AdaBoost in terms of balance accuracy, with SMOTEBoost achieving the highest accuracy score. Despite the fact that the accuracy and equalized odds of two fairness-aware approaches are lower than AdaBoost, the TPR of both protected and non-protected classes is significantly higher. TNR protected classes of AdaBoost are slightly more than fairness methods in protected classes but significantly more in non-protected classes. Referring to the figure, both fairness-aware approaches can improve the balance accuracy and TPR of protected and non-protected classes of this dataset.

5.2.2. Bank dataset.

As shown in Figure 2, both fairness-aware approaches perform nearly as well as AdaBoost in terms of accuracy and

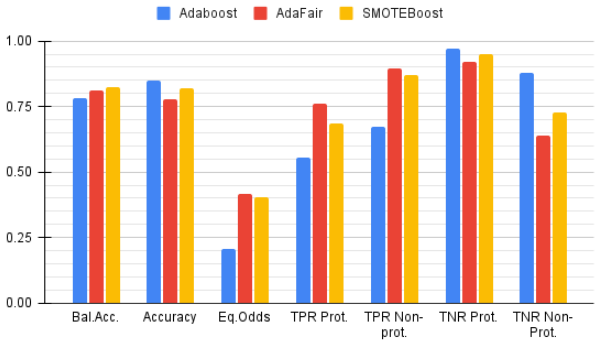

Figure 1. Performance of algorithms on Adult Census dataset.

TNR of both protected and non-protected classes. Furthermore, both techniques outperform AdaBoost in terms of balance accuracy and TPR while falling short in terms of TNR. According to the figure, both fairness-aware approaches can preserve accuracy and TNR while approving the dataset's balance accuracy and TPR.

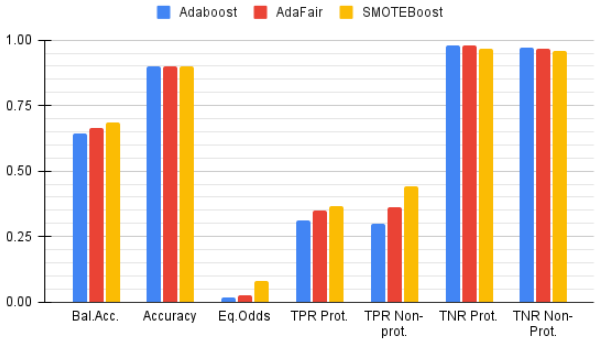

Figure 2. Performance of algorithms on Bank Census dataset.

5.2.3. Compas dataset.

Figure 3 shows that all approaches perform nearly identically in this dataset, with AdaFair scoring slightly higher than the others in almost all cases. According to the figure, both fairness-aware approaches can be utilize to detect the unfair classification in the sensitive attribute.

5.2.4. KDD Census dataset.

KDD dataset. As shown in Figure 4, AdaFair has the highest balance accuracy and TPR scores in this dataset, whereas AdaBoost and SMOTEBoost have better accuracy and TNR scores. In equalized odds, SMOTEBoost has the best score. According to the graph, both fairness-aware approaches can

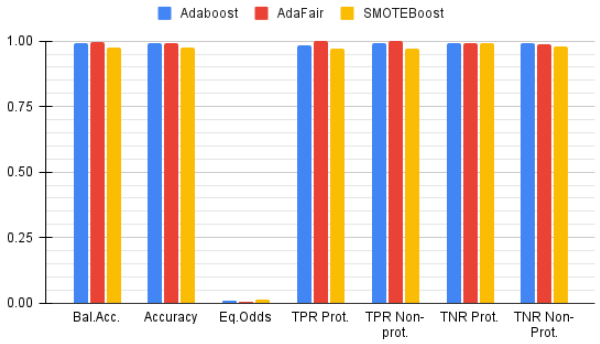

Figure 3. Performance of algorithms on Compas dataset.

be utilized to achieve classification fairness, with AdaFair outperforming the others in TPR.

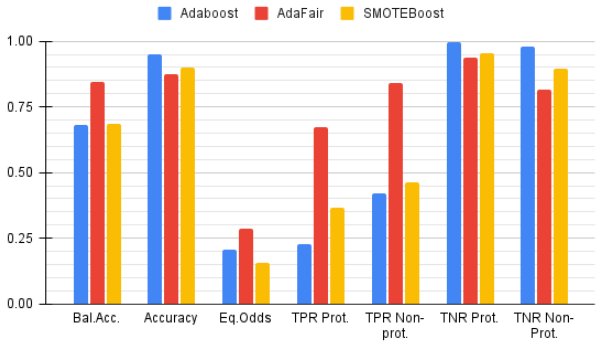

Figure 4. Performance of algorithms on KDD Census dataset.

6. Conclusion

In this work, we replicate the AdaFair and SMOTEBoost algorithms in order to reduce their bias. Following the implementation of both algorithms, we evaluate their predictive and fairness performance in terms of balance accuracy, accuracy, TPR, and TNR of protected and non-protected classes.

The outcome of analyzing all algorithms with four real-world datasets reveals a significant difference in TPR score for both protected and non-protected classes. TPR and balance accuracy can be outperformed by AdaFair and SMOTEBoost. On the other hand, improving unfair classification algorithms can have an effect on accuracy, TNR, and equalized odds scores in some cases.

Moreover, AdaFair is able to achieve the accuracy and fairness (balance accuracy and TPR) in both extreme class-imbalance and nearly class-balance.

References

Benjamin Fish, Jeremy Kun, and Ad´ am D Lelkes. 2016. A confidence-based ´ approach for balancing fairness and accuracy. In Proceedings of the 2016 SIAM International Conference on Data Mining. SIAM, 144–152.

Faisal Kamiran and Toon Calders. 2009. Classifying without discriminating. In Computer, Control and Communication. IEEE, 1–6.

Faisal Kamiran and Toon Calders. 2012. Data preprocessing techniques for classification without discrimination. KAIS 33, 1 (2012), 1–33.

Jeff Larson and Marjorie Roswell and Vaggelis Atlidakis, 2017. URL https://raw.githubusercontent.com/propublica/compas-analysis/master/compas-scores-two-years.csv

Moritz Hardt, Eric Price, Nati Srebro, et al. 2016. Equality of opportunity in supervised learning. In NIPS. 3315–3323.

Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez Rodriguez, and Krishna P Gummadi. 2017. Fairness beyond disparate treatment disparate impact: Learning classication without disparate mistreatment. In Proceedings of the 26th International Conference on World Wide Web. WWW, 1171–1180.

Nitesh V Chawla, Aleksandar Lazarevic, Lawrence O Hall, and Kevin W Bowyer. 2003. SMOTEBoost: Improving prediction of the minority class in boosting. In ECML PKDD. Springer, 107–119.

Paulo Cortez (Univ. Minho) and Sérgio Moro (ISCTE-IUL), 2012. URL https://www.kaggle.com/datasets/prakharrathi25/banking-dataset-marketing-targets?select=train.csv

Robert E Schapire. 1999. A brief introduction to boosting. In IJCAI, Vol. 99.1401–1406.

Ronny Kohavi and Barry Becker. Data Mining and Visualization, Silicon Graphics, 2017. URL https://www.kaggle.com/uciml/adult-census-income

Sahil Verma and Julia Rubin. 2018. Fairness Definitions Explained. (2018).

Terran Lane and Ronny Kohavi. Data Mining and Visualization Silicon Graphics, 2000. URL https://archive.ics.uci.edu/ml/datasets/Census-Income+(KDD)

Vasileios Iosidis and Eirini Ntoutsi. 2016. AdaFair: Cumulative Fairness Adaptive Boosting.

Y. Freund, R. Schapire, Experiments with a New Boosting Algorithm, Proceed-ings of the 13th International Conference on Machine Learning, 325-332, 1996.

A. Team member's contributions

Explicitly stated contributions of each team member to the final project.

Daniil Babin (20% of work)

- Reviewing literate on the topic (1 papers)
- Replicating Classifiers Without Disparate Mistreatment algorithm (Unsuccess)
- Reviewing the paper overall completeness and presentation video

Sudarut Kasemsuk (40% of work)

- Reviewing literate on the topic (2 papers)
- Replicating SMOTEBoost algorithm
- Preparing the GitHub Repo
- Preparing the Section 1-4 of this report
- Build the presentation 50%
- Make the presentation video 50%

Waralak Pariwatphan (40% of work)

- Reviewing literate on the topic (1 papers)
- Replicating AdaFair algotithm
- Data pre-processing
- Preparing the GitHub Repo
- Preparing the Section 4-6 of this report
- Build the presentation 50%
- Make the presentation video 50%

B. Reproducibility checklist

Answer the questions of following reproducibility checklist. If necessary, you may leave a comment.

1. A ready code was used in this project, e.g. for replication project the code from the corresponding paper was used.

   ☑ Yes.
   ☐ No.
   ☐ Not applicable.

   General comment: If the answer is **yes**, students must explicitly clarify to which extent (e.g. which percentage of your code did you write on your own?) and which code was used.

   Students' comment: For AdaFair and SMOTEBoost, we implement the codes from the corresponding paper, and around 70% of the total code is adjusted, e.g., all of the code in the calculate fairness function is modified from the original to improve overall accuracy.

2. A clear description of the mathematical setting, algorithm, and/or model is included in the report.

   ☑ Yes.
   ☐ No.
   ☐ Not applicable.

   Students' comment: Adafair and SMOTEBoost algorithms are included in 4.1 and 4.2.

3. A link to a downloadable source code, with specification of all dependencies, including external libraries is included in the report.

   ☑ Yes.
   ☐ No.
   ☐ Not applicable.

   Students' comment: All these things could be found in section: Github repo.

4. A complete description of the data collection process, including sample size, is included in the report.

   ☑ Yes.
   ☐ No.
   ☐ Not applicable.

   Students' comment: Data description is listed in Table 1 and 5.1.

5. A link to a downloadable version of the dataset or simulation environment is included in the report.

   ☑ Yes.
   ☐ No.
   ☐ Not applicable.

   Students' comment: In 5.1 section, we provide all of the download links for the datasets.

6. An explanation of any data that were excluded, description of any pre-processing step are included in the report.

   ☑ Yes.
   ☐ No.
   ☐ Not applicable.

   Students' comment: All explanation of data and preprocessing step are included in the 5.1.

7. An explanation of how samples were allocated for training, validation and testing is included in the report.

   ☑ Yes.
   ☐ No.
   ☐ Not applicable.

   Students' comment: The percentage of sample split is indicated at the end of 5.1.

8. The range of hyper-parameters considered, method to select the best hyper-parameter configuration, and specification of all hyper-parameters used to generate results are included in the report.

   ☑ Yes.
   ☐ No.
   ☐ Not applicable.

   Students' comment: We don't have range of hyperparameters but we have the specifific parameters to compare each algorithms in part 5.1.

9. The exact number of evaluation runs is included.

   ☑ Yes.
   ☐ No.
   ☐ Not applicable.

   Students' comment: We declare the number of evaluation runs at the of part 5.1.

10. A description of how experiments have been conducted is included.

    ☑ Yes.
    ☐ No.
    ☐ Not applicable.

    Students' comment: For description of Adafair experiment is shown in part 4.1 and SMOTEBoost experiment is shown in part 4.2.

11. A clear definition of the specific measure or statistics used to report results is included in the report.

- ☐ Yes.
- ☑ No.
- ☐ Not applicable.

Students' comment: We don't have all definition of the specific measure or statistics used to report results but some measures are included in 3.

12. Clearly defined error bars are included in the report.

    - ☐ Yes.
    - ☑ No.
    - ☐ Not applicable.

    Students' comment: None

13. A description of the computing infrastructure used is included in the report.

    - ☐ Yes.
    - ☐ No.
    - ☑ Not applicable.

    Students' comment: None