Preventing Discriminatory Decision-making in Evolving Data Streams

Zichong Wang
zichongw@mtu.edu
Michigan Technological University
Houghton, Michigan, USA

Nripsuta Saxena
nsaxena@usc.edu
University of Southern California
Los Angeles, California, USA

Tongjia Yu
tongjia2428@columbia.edu
Columbia University
New York, New York, USA

Sneha Karki
snehak@mtu.edu
Michigan Technological University
Houghton, Michigan, USA

Tyler Zetty
tjzetty@mtu.edu
Michigan Technological University
Houghton, Michigan, USA

Israat Haque
israat@dal.ca
Dalhousie University
Halifax, Nova Scotia, Canada

Shan Zhou
shanzhou@mtu.edu
Michigan Technological University
Houghton, Michigan, USA

Dukka Kc
dbkc@mtu.edu
Michigan Technological University
Houghton, Michigan, USA

Ian Stockwell
istock1@umbc.edu
University of Maryland, Baltimore County
Baltimore, Maryland, USA

Xuyu Wang
xuywang@fiu.edu
Florida International University
Miami, USA

Albert Bifet
abifet@waikato.ac.nz
University of Waikato
Hamilton, New Zealan

Wenbin Zhang
wenbinzh@mtu.edu
Michigan Technological University
Houghton, Michigan, USA

ABSTRACT

Bias in machine learning has rightly received significant attention over the past decade. However, most fair machine learning (fair-ML) works to address bias in decision-making systems has focused solely on the offline setting. Despite the wide prevalence of online systems in the real world, work on identifying and correcting bias in the online setting is severely lacking. The unique challenges of the online environment make addressing bias more difficult than in the offline setting. First, Streaming Machine Learning (SML) algorithms must deal with the constantly evolving real-time data stream. Secondly, they need to adapt to changing data distributions (concept drift) to make accurate predictions on new incoming data. Incorporating fairness constraints into this already intricate task is not straightforward. In this work, we focus on the challenges of achieving fairness in biased data streams while accounting for the presence of concept drift, accessing one sample at a time. We present *Fair Sampling over Stream (*FS^2*)*, a novel fair rebalancing approach capable of being integrated with SML classification algorithms. Furthermore, we devise the first unified performance-fairness metric, *Fairness Bonded Utility (*FBU), to efficiently evaluate and compare the trade-offs between performance and fairness across various bias mitigation methods. FBU simplifies the comparison of fairness-performance trade-offs of multiple techniques through one unified and intuitive evaluation, allowing model designers to easily choose a technique. Overall, extensive evaluations show our measures surpass those of other fair online techniques previously reported in the literature.

KEYWORDS

Fairness, Data Stream, Concept Drift, Fairness Drift

ACM Reference Format:
Zichong Wang, Nripsuta Saxena, Tongjia Yu, Sneha Karki, Tyler Zetty, Israat Haque, Shan Zhou, Dukka Kc, Ian Stockwell, Xuyu Wang, Albert Bifet, and Wenbin Zhang. 2022. Preventing Discriminatory Decision-making in Evolving Data Streams. In *The 6th ACM Conference on Fairness, Accountability, and Transparency.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

Machine learning-based decision-making systems are becoming increasingly crucial due to their extensive applications across diverse fields, including criminal justice, job application screening, loan decisions, and resource allocation. However, the growing reliance on automated decision-making systems has led to heightened scrutiny concerning issues of fairness and accountability in these models [12, 37, 53]. For instance, Amazon's decision to abandon an automated hiring tool after discovering its bias against women [29] is just one among many similar cases [11, 31]. The detrimental effects of such biases underscore the importance of developing fairness-conscious classifiers that yield accurate predictions without discriminating against marginalized subgroups in society. Various strategies have been proposed for fair decision-making systems, e.g., to identify and eliminate discrimination [20, 40] and mitigate

ACM FAccT 2023, 12 - 15 June, 2023, Chicago, USA
© 2022 Association for Computing Machinery.
ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/10.1145/1122445.1122456

the inherent bias in historical data [24, 25]. However, these approaches overlook a critical domain that is prevalent in the real world: the online setting.

In real-world streaming applications, data is generated in real-time (online) and its characteristics and distribution change over time. Consequently, it is essential to design non-discriminatory decision systems that consider the unique challenges of the online setting. However, most work in fair-ML approaches fairness as a static issue and assumes that all data can be afforded for multiple scans and that the underlying population characteristics do not evolve [20, 38, 46]. Thus, they fail to address the challenges in the online setting. Specifically, the first significant challenge involves *handling the constantly produced real-time data stream*, i.e., the data is infinite. Since no memory can load infinite data, fairness-aware learning for the online setting must process each incoming instance "on arrival," without requiring storage and reprocessing. Most fair-ML, however, assumes the entire dataset can be scanned multiple times.

The second challenge is that *the target concepts may also evolve*. In other words, the instance at time $t + 1$ can be generated by a different function than the one that generated the instance at time t. For example, in early 2020, patients with symptoms like fever, cough, etc., would likely have a seasonal flu diagnosis, which may not be the case in the Covid-19 era. While initially, Covid-19 would have represented a minority of the cases as compared to the flu, over time, it became the majority class. Thus, the class we need to balance in our dataset can change over time. Addressing bias and changing distribution at the same time is difficult. The third significant challenge is *an intuitive measure capable of quantifying the trade-off between fairness and performance*. So far, fairness evaluations have primarily reported fairness improvement and accuracy loss as two separate metrics without a reliable method for jointly measuring the inherent trade-off between them. However, in real-world applications, business analytics may require a single metric for both performance measures [46]. Therefore, a unified and efficient measurement metric is essential for clear comprehension of the trade-off between fairness and accuracy of the model.

To address the aforementioned challenges, we present *Fair Sampling over Stream* (FS^2), a novel meta-strategy for online stream-based decision-making and *Fair Bonded Utility* (*FBU*), a novel unified fairness-performance trade-off measurement metric. Our approach eliminates the bias in the data stream before applying the SML (Streaming Machine Learning) classification algorithm [3]; thus, the proposed approach is model agnostic. The novel fairness improvement method takes into account continuous data streams, eliminates discrimination, and accounts for evolving data distribution. Additionally, it strikes an effective and robust balance between prediction accuracy and fairness performance. Our major contributions are:

- FS^2: a meta-strategy that builds on the popular data balancing scheme SMOTE [9] and can be integrated with any data stream classifier. Unlike the existing SMOTE-based fair stream processing [3, 19], FS^2 introduces a fairness guarantee without performance degradation and is not restricted to batches.
- A novel fairness-performance metric that assesses the fairness-accuracy trade-off of ML bias mitigation methods in a unified and intuitive manner.
- Qualitative and quantitative experiments on five groups of biased data streams show the effectiveness of the proposed unified metrics and fairness-conscious online learners in streaming environments.

The remainder of the paper is organized as follows. Section 2 and Section 3 review relevant work and the necessary background on discrimination-aware learning. Next, we discuss the limitations of the class balancing technique in Section 4. We present FS^2 and bias-performance measurement metric in Section 5. Section 6 analyzes the experimental results. Finally, we conclude the paper in Section 7.

2 RELATED WORK

2.1 Fairness-aware Learning

A number of approaches have been proposed to address the problem of bias and discrimination in machine learning and can be broadly grouped under three categories: pre-, in-, or post-processing [14, 34, 43, 47, 48, 52]. i) *Pre-processing techniques* focus on mitigating and correcting bias in the data used for training the model, i.e., model-independent. The most popular ones include massaging [25] and reweighting [6], which are also extended in [24]. ii) *In-processing techniques* transform an algorithm to improve fairness, typically by embedding the fairness in the objective function using regularization or other constraints [51]. The authors in [27] first propose this category by integrating discrimination into the model splitting criteria for fair tree induction. Recently, [49] further improved and extended these splitting criteria in the online settings. iii) *Post-processing techniques* are based on either adapting the decision boundary or simply modifying the output of a model, e.g., the prediction labels. With supplementary prediction thresholds in place, [21] works against discrimination, while in [7], the decision boundary of AdaBoost is adjusted for fairness. However, transferring such approaches to an online setting is not trivial, as boundary/prediction might evolve due to the non-stationary spread over time. However, the above approaches fail to tackle bias and discrimination in the pre-processing of stream data. This work proposes a novel pre-processing method that effectively mitigates bias in streaming data and ensures fairness in the resulting machine learning models.

2.2 Stream Learning

The main challenges of learning in a stream setting are concept drift, where the joint data distribution changes over time, and class imbalance [1, 16, 28]. To address the first challenge, learning methods must adapt incrementally by incorporating new information into the model [32, 41, 42] and by forgetting outdated information [17, 50]. Various sampling methods have been proposed for the second challenge, including the most representative (C-SMOTE) [3] to address the class imbalance in a continuous data stream while ensuring the model can adapt to shifts in the data distribution. However, these techniques focus entirely on accuracy without considering fairness.

2.3 Fairness-aware Stream Learning

Despite the wide prevalence of the stream learning setting and the presence of bias and discrimination in real-world applications, surprisingly, little work has explored this environment. The authors in [2] consider an individual fairness approach and deploy an auditor to detect fairness violations. Online SMOTE Boost [39] is made cost-sensitive by adding various parameters of the Poisson distribution for different classes to address bias. In addition, FABBOO [23] adjusts the training distribution on-the-fly, considering both the imbalanced nature of the stream and the model's discriminatory behavior as evaluated from past data. One common drawback to all of these methods is that their balancing approaches are driven by the SMOTE, which can further exacerbate bias (c.f., Section 4). Our method addresses this drawback by introducing an additional clustering for a fairness guarantee.

3 BASIC CONCEPTS

Let D be a sequence of instances $d_1, d_2, \ldots, d_n$, where $d \in \mathbb{R}$, arrives at time $t_1, t_2, \ldots, t_n$. Also, let Y be a sequence of corresponding class labels, such that for each $d \in D$, there exists a corresponding class label $y \in Y$. We assume each data instance in the continuous stream has the form $d = \{A, S, Y\}$, where A, S, and Y represent a set of attributes, the sensitive attribute (e.g., gender/race), and the true class label, respectively. In fair-ML, a sensitive attribute is a characteristic legally protected from discrimination (e.g., race, color, gender, etc.). Though some sensitive attribute classes are historically marginalized (e.g., people of color or women), others are still untouched. We call the former and the latter classes *unprivileged group* and *privileged group*, respectively. Without loss of generality, we assume a binary classifier, $f() : D \longrightarrow y$, such that $y \in \{-1, +1\}$, where the predicted class label is $\hat{Y}$.

Usually, each new data instance from the continuous stream is processed one by one in the *online learning environment*. A characteristic of that learning is that the true class label of each instance is revealed to the learner before the arrival of the next instance. Specifically, if we predict the class label of a data instance d arriving at time t using f_{t-1}, i.e., the classifier from time $t-1$, the true class label of d is revealed to the online learner before the arrival of the next data instance. The model is then updated using this true class label to get f_t. This sequence of events is referred to *first-test-then-train* or *prequential evaluation* setup [15].

Concept drift, the changes in the underlying distribution of the continuous data stream over time, is another characteristic of online learning. We can define the changes as $P_{t_a}(D, Y) \neq P_{t_b}(D, Y)$ for two different time t_a and t_b. Such changes often render the current classifier ineffective since it cannot deal with the changes in the data distribution, thereby necessitating a model update. Also, the continuous data stream can be imbalanced. Machine learning models typically neglect the minority class in such situations to prevent overfitting and loss of generalization [44]. Our technique does not require the minority class to be predefined in advance and assumes that the same class will always be the minority class. In other words, we allow for the possibility that the role of the minority class may alternate between the two classes. We analyze the extent of the imbalance in data by computing the imbalance ratio (IR) using the following equation.

$$IR = \frac{S_{t_{min}}}{S_{t_{maj}} + S_{t_{min}}} \tag{1}$$

where $S_{t_{min}}$ and $S_{t_{maj}}$ are the minority and majority classes at time t, respectively.

The offline fairness-aware classification performs a mapping, $F : (A, S, Y) \Rightarrow Y$, where the goal is to assign a class label to each data instance accurately without discriminating against the unprivileged group. The desirable class label (e.g., receiving a loan in loan decisions) is known as *'favorable' label* and the undesirable one (e.g., being denied a loan) is called *'unfavorable' label.* We focus on parity-based approaches that compare a fair-ML model's behavior towards the unprivileged and privileged groups. We employ statistical parity [26] and equal opportunity [21], which we extend to the online setting.

Statistical parity quantifies the disparity between the probability of the privileged and unprivileged groups receiving a benefit. The Cumulative Statistical Parity Difference ($CSPD$) in Equation 2 represents the cumulative difference in statistical parity between the privileged and unprivileged groups over time. $\lambda \in [0, 1]$ is a decay factor used to regulate the amount of influence of the previous time instance. In other words, the larger the λ (a configurable parameter), the higher the contribution of historical information.

$$\begin{aligned} CSPD_t = (1-\lambda) \times (&\frac{\sum_{i=1}^{t}(F_i(d_i) \wedge Y = 1 | S = \bar{s})}{\sum_{i=1}^{t}(d_i | S = \bar{s}} \\ &- \frac{\sum_{i=1}^{t}(F_i(d_i) \wedge Y = 1 | S = s)}{\sum_{i=1}^{t}(d_i | S = s)}) + \lambda \times CSPD_{t-1} \end{aligned} \tag{2}$$

However, statistical parity may result in the phenomenon of reverse discrimination in certain conditions, i.e., the scenario when data instances that are not deserving of a positive assignment receive one incorrectly, since it does not consider real class labels. Therefore, we also consider Equal Opportunity, which measures the difference in True Positive Rates (TPR) between the privileged and unprivileged groups. Cumulative Equal Opportunity Difference $CEOD$ in Equation 3 is the cumulative difference in equal opportunity between the privileged and unprivileged groups over time.

$$\begin{aligned} CEOD_t = (1-\lambda) \times (&\frac{\sum_{i=1}^{t}(F_i(d_i) \wedge Y = 1 | S = \bar{s}, Y = 1)}{\sum_{i=1}^{t}(d_i | S = \bar{s}, Y = 1)} \\ &- \frac{\sum_{i=1}^{t}(F_i(d_i) \wedge Y = 1 | S = s, Y = 1)}{\sum_{i=1}^{t}(d_i | S = s, Y = 1)}) + \lambda \times CEOD_{t-1} \end{aligned} \tag{3}$$

where $\wedge$ is the logical symbol and λ is used for correction in the early stages of the stream.

4 IMPACT OF CLASS BALANCING ON FAIRNESS

Inequality in class distribution is a hallmark of biased data. As the instances in the underprivileged group are not as prevalent as those in the privileged group, the model can overlook patterns in the classification of the underprivileged group. Thus, models trained on biased datasets can assign favorable labels to the privileged subgroup and do the opposite to the unprivileged one to achieve high accuracy. We can deploy class balancing techniques such as oversampling and undersampling to address unequal representation.

Oversampling increases the minority class representation by creating synthetic instances while undersampling balances representation by reducing majority class instances. We examine the drawbacks of these techniques and how they affect the bias and fairness in a model's output in detail in later. Some key limitations of popular class-balancing techniques include: oversampling can lead to overfitting if samples from a few classes are replicated multiple times [8, 13, 45] and undersampling can lead to hidden information being ignored due to the samples that were removed [9, 35, 50]). However, other under-explored drawbacks exist, particularly in fairness research, while alleviating bias.

Traditional class-balancing techniques aim to equalize the statistical representation of minority and majority class samples. However, since these methods focus on addressing the imbalance solely concerning the target variable (i.e., the class label), they do not account for the inherent properties of the dataset. Thus, the bias in the dataset can increase after balancing the class labels, further degrading the model's fairness. This is particularly problematic in the case of multiple privileged and unprivileged subgroups. Not all privileged subgroups are privileged in the same way and/or to the same extent; the same goes for the unprivileged subgroups. Undersampling can lead to a disproportional increase in the number of instances of the more favorable privileged group. On the other hand, oversampling may result in a disproportionate increase in the instances of the more favored unprivileged group because they may be a bigger proportion of the minority class, leaving the worst favored unprivileged group severely underrepresented.

Figure 1 presents our initial evaluation results, which reveal that common class balancing techniques can lead to further distribution bias in the dataset. In Figure 1, the original Bank Marketing dataset has 24% of positive (subscribe) samples while the rest are negative (non-subscribe). After class balancing with Random Over Sampling (ROS), 11,788 more positive samples are added, of which only 15% are unprivileged subgroup. Thus, the process increases bias in the distribution, as more privileged subgroup samples are added during the balancing process. In other words, the representation of unprivileged favorable groups in the data set declines further.

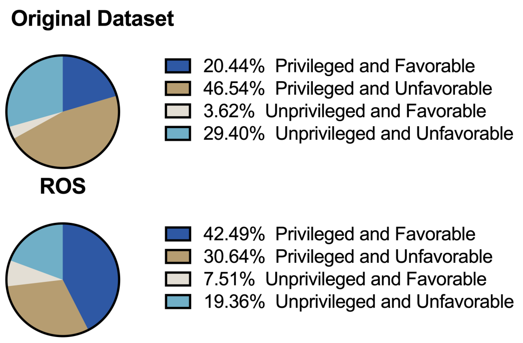

Figure 1: The distribution bias in the Bank Marketing dataset with and without class balancing.

Figure 2 represents another limitation of the common class balancing technique, SMOTE. Specifically, the minority class samples synthesized by SMOTE may belong to the majority class in the feature space. Meanwhile, the random selection of samples can exacerbate in-class bias in the dataset, especially when dealing with multiple sensitive attributes. Our proposed scheme overcomes the above-mentioned issues, which we elaborate on in Section 5.2.2.

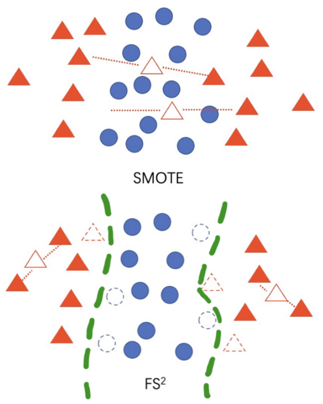

Figure 2: Illustration of oversampling by SMOTE and FS^2. Circle and Triangular nodes represent samples with majority and minority labels, respectively. The hollow nodes are samples created based on the closest neighbors of minority samples within each cluster.

5 METHODOLOGY

This section first introduces the proposed Fairness Bonded Utility (FBU) metric to combine fairness and performance into a single metric. To the best of our knowledge, FBU is the first metric designed to compare various combinations of fairness and performance metrics effectively. Subsequently, we present Fair Sampling over Stream (FS^2), an innovative fair rebalancing technique tailored for biased data streams. This method can be seamlessly integrated with any stream-learning classification technique, thus offering a versatile solution for addressing fairness concerns in stream processing.

5.1 Fairness Bonded Utility

A long-standing challenge in fair-ML is the intricate trade-off between model performance and fairness. Typically, a model's performance degrades as fairness guarantees become stronger. This complicates the decision about which fairness metric to use for what application, as analyzing this trade-off (performance and fairness) is not trivial. Fairness Bonded Utility (FBU) solves this problem by assigning each fairness technique into one of five intuitive "effectiveness" levels ranging from "Jointly advantageous" (fairness and performance both improve) to "Jointly disadvantageous" (both degrade). At a high level, FBU does this by measuring how the performance of a model varies with different fairness techniques in a two-dimensional coordinate space and computing a trade-off baseline that categorizes the various fairness techniques into effectiveness levels. FBU can drastically simplify the agonizing decision-making process of which fairness technique to use for which application scenario.

We first construct the environment for the FBU to measure the performance of different fairness techniques. Specifically, it is a two-dimensional coordinate system (see Figure 3). The x-axis and y-axis represent the pseudo-models with different replacement ratios and the model's performance and fairness, respectively. Our technique is flexible and can be used with any fairness metrics (e.g., CSPD, CEOD) to measure fairness and performance metrics (e.g., accuracy, f1-score). Next, we present the trade-off baseline and the evaluation outcome.

Creating a trade-off baseline. The idea behind the trade-off baseline in FBU is centered around the zero-normalization principle presented by Speicher et al. [36]. It states that bias is minimized when every individual receives the same label, although this may lead to decreased performance and vice versa.

We use this principle to create multiple pseudo-models, M_p, where $p \in \{10\%, 20\%, \ldots, 100\%\}$. We substitute a proportion p of the original model's predictions in each pseudo-model with the same output label. Then, we vary the proportion of substituted predictions in each pseudo model, with p ranging from 10% to 100%. For instance, in M_{10}, we randomly select 10 percent of the predictions from the original predictions to be replaced. It is worth noting that we choose the labels with a higher percentage from the original model as replacement labels. In M_{100}, we replace all the predictions of the original model with the highest percentage of output labels. In general, as the proportion of replaced predictions increases, the model's fairness improves, but its accuracy declines.

Next, we use the original model (with no fairness technique) and the created pseudo models to form the trade-off baseline. We derive the first point for the trade-off baseline from the (performance, fairness) coordinate of the original model (point M_{ori} in Figure 3(b)). Then, we plot the (performance, fairness) coordinates of the ten pseudo models (i.e., M_{80}, M_{90}, $M_{100}, etc.$ in Figure 3(b)). Finally, we connect them to form the trade-off baseline.

Five effectiveness levels. The trade-off baseline categorizes the different bias mitigation techniques being compared into five intuitive effectiveness levels, each with a different bias-performance trade-off. The first level, region 1 in Figure 3(b), is a "Jointly advantageous": A technique belongs in this category if it improves both model performance and fairness relative to the trade-off baseline. Region 2 in Figure 3(b) denotes a "Impressive" trade-off: a technique in region 2 enhances either model performance or fairness compared to the trade-off baseline, making it overall better than the trade-off baseline. If a technique improves model performance but leads to a decrease in fairness, it is considered an "Reversed" trade-off and falls into region 3 (Figure 3(b)). A technique in region 4 in Figure 3(b) is a "Deficient" trade-off: it leads to a decline in either model performance or fairness compared to the baseline, making it overall worse than the trade-off baseline. A region 5 technique (in Figure 3(b)) represents a "Jointly disadvantageous" trade-off since it decreases both model performance and fairness compared to the original model.

Quantitative evaluation of trade-offs. The "Jointly advantageous" (region 1), "Jointly disadvantageous" (region 5), and "Deficient" (region 4) trade-off regions offer clear indications of the effectiveness of the bias mitigation technique. Therefore, we concentrate on quantifying the trade-off quality of bias mitigation techniques in the "Impressive" trade-off region. Specifically, we will measure the strengths and weaknesses of different bias mitigation methods that fall in the "Impressive" region (region 2) based on the area enclosed by the fairness-performance points and the baseline (see Figure 3(b)). Techniques with larger areas have better and desirable fairness-performance trade-offs. Using area as a measure of trade-off instead of other criteria, such as distance from the trade-off baseline, we can have a fair comparison even in cases where the trade-off baseline is curved.

The final output of FBU for a technique will be five percentages: one percentage for each region representing how many cases for that technique fell in that region. The number of cases per region is computed by: $n_r \ x \ n_t \ x \ n_f \ x \ p_m$, where n_r, n_t, n_f, and p_m are the number of run times, techniques compared against, fairness metrics used, and performance metrics used, respectively.

5.2 Fair Sampling over Stream

There are two fundamental challenges facing Fair Sampling over Stream (FS^2): the first is the lack of the entire dataset in the rebalancing phase. The second is synthesizing fair data. We first discuss the online monitoring of the continuous data stream and model updates. Then we elaborate on the challenge of synthesizing fair data.

5.2.1 *Online Monitoring and Model Update.*

In continuous data streams, the status of minority and majority classes can evolve. In other words, a class currently identified as a minority may turn into the majority class, or vice versa [41]. Additionally, approaches focusing solely on fairness in recent outcomes (i.e., in the immediate future) are inadequate in combating discrimination over time. Although individual instances of discrimination may seem minor when evaluated alone, they can accumulate and result in substantial bias in the long term. Hence, monitoring the subgroup distribution in the data stream over time is vital for both the efficiency and fairness of the model. However, the primary challenge faced by FS^2 is its inability to consistently monitor data to address both concept drift and fairness concerns. In streaming machine learning, we assume the data stream to be infinite. Firstly, no memory can physically store infinite data; secondly, this would be against the stream paradigm approach. Therefore, storing every new sample in memory is impossible until the data stream ends.

We employ a different approach from previous non-stationary studies [10, 18] to address this problem. FS^2 employs ADWIN [5] to detect changes in both accuracy and fairness, reflecting both concept and fairness drifts. In other words, drift is detected when either fairness or the data distribution evolves. Specifically, we will use ADWIN to save the most recently seen samples and apply the synthesis algorithm (explained below in Section 5.2.2) using the samples stored in ADWIN. ADWIN maintains a variable-length window of recently seen items and can automatically detect and adapt the window size to the current rate of change in the data. As shown in Equation 4, ADWIN employs a threshold known as δ (i.e., performance threshold and fairness threshold) to configure the error to have two levels automatically: warning level and change level.

$$levelError = \log(\frac{2 \times \log n}{\delta}) \quad (4)$$

We identify the warning level by multiplying δ by 10, and δ helps determine the change level. Since δ is in the denominator,

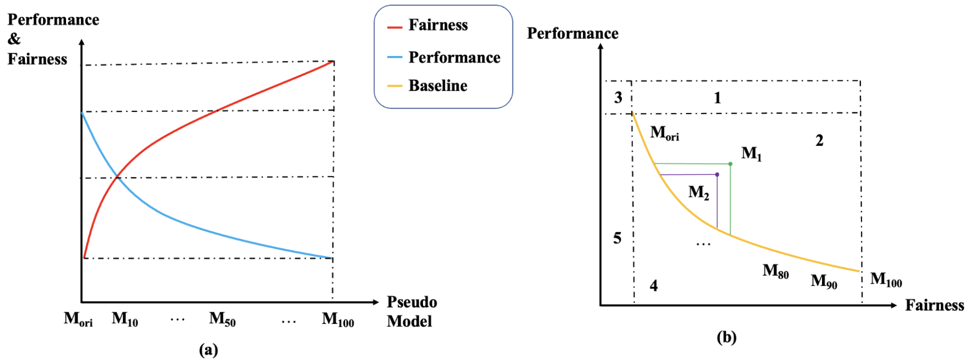

Figure 3: The FBU fairness-accuracy trade-off baseline is depicted by the original trade-off point (M_{ori}) and the points generated by the pseudo models (M_{10},. . . , M_{100}). A bias reduction method is considered effective if it shows a superior trade-off compared to the FBU baseline, i.e., it lies above the red line.

multiplying it by 10 will result in a lower value than the one obtained by using δ alone. Therefore, we will reach the warning level before the change level and can detect changes early to take appropriate action. The window size at any given time is represented by n.

ADWIN monitors the error that occurs over the data in the window. If the error reaches a threshold of the warning level, ADWIN assumes that either a concept drift or fairness drift is starting to occur, and it starts collecting new samples in a new window. If the error exceeds the change level, ADWIN assumes either a concept drift or a fairness drift has occurred, and it substitutes the old window with the new one.

5.2.2 Synthetic fair data. Here we propose a new strategy for synthesizing fair data to address the challenges posed by class balancing and its impact on model fairness. Our proposed approach eliminates the in-class and between-class imbalance in the oversampling part. Unlike traditional class balancing techniques such as SMOTE, our method identifies the true minority classes in the dataset (see Figure 2), and adds fairness constraints to the generation process for synthetic samples.

At a high level, rather than directly running SMOTE, our approach first determines similar samples via clustering, then filters the clusters using silhouette score to ensure a good clustering. Finally, it uses SMOTE within each cluster to generate synthetic minority samples. Notably, our approach not only enhances the fairness of the balancing method but also preserves the accuracy of the prediction. We discuss our approach in detail below.

First, we use a clustering algorithm to identify homogeneous subgroups of feature similarity in the feature space. In our experiments, we use the KNN algorithm. Note that any clustering algorithm can be applied, but different clustering algorithms may have different clustering results and time complexity. By partitioning the dataset according to the selected number of clustering families M, samples with similarities in the feature space will be classified into the same clusters. We measure the quality of clustering by silhouette analysis. The silhouette score measures how similar an object is to its own cluster compared to others, and the value is between $[-1, 1]$. A value of 1 indicates perfect separation, and a value of -1 suggests that the object is in the wrong cluster. Values between 0 and 1 show the degree of separation, with values closer to 1 indicating a stronger separation.

After determining the optimal number of clusters M, the algorithm performs filtering detection for each cluster. The algorithm calculates the silhouette score of each sample and then removes samples with the lowest 20% silhouette score from the dataset. The intuition for this is as follows. Let's imagine a realistic data distribution situation where a sample of a minority group exists near the cluster boundaries. In this case, we can assume that SMOTE selects the minority group sample near the clustering boundary as the template. Then, the synthetic data generated based on this sample will also be close to the clustering boundary. In addition, there is a risk that the generated synthetic data are even closer to the clustering boundary than the original data. This increases the number of samples near the cluster boundaries, thus blurring the boundaries between different clusters, making it difficult to separate these samples in the feature space and a higher probability of being assigned to incorrect clusters. As a result, new instances generated based on these samples may further increase the distribution bias.

After the filtering is complete, the algorithm checks each cluster. Specifically, the algorithm detects the proportion of minority class samples in each cluster to determine its synthetic weight, which is directly related to the proportion of minority samples in the cluster. If the percentage of minority samples in a cluster is high, the cluster is a better representative of minority samples. In other words, clusters with a high percentage of minority class samples are assigned higher synthetic weights. For each cluster, the synthetic weight is calculated according to Equation 5:

$$ClusterWeight = \begin{cases} W_i = \frac{N_i}{N_{Total}} \\ W_1 + W_2 + \cdots + W_n = 1 \end{cases} \tag{5}$$

where N_i is the number of minority class samples in each cluster, N_{Total} is the total number of minority class samples, and W_i is the synthetic weight of cluster i. The sum of all weights is one, and the number of samples selected for each cluster is calculated according to Equation 6:

$$G_i = N_{num} \times W_i \tag{6}$$

where N_{num} is the total number of samples to be synthesized. After calculating the number of samples selected for each cluster, the model will randomly select samples in the clusters, and a new minority sample is generated according to the SMOTE [9] generation algorithm based on the k-nearest neighbors of the minority samples.

5.2.3 FS^2 *Algorithm.* The algorithm is outlined in Algorithm 1, which uses two sliding windows of samples W and W_{label} (line 1). W contains all samples collected recently. W_{label} is an array indicating whether the corresponding samples in W belong to the privileged favorable subgroup, privileged unfavorable subgroup, unprivileged favorable subgroup, or unprivileged unfavorable subgroup. The FS^2 algorithm has nine counters (lines 2-4): C_0, C_1, C_2 and C_3 count the number of instances in each subgroup, while $\overline{C_0}$, $\overline{C_1}$, $\overline{C_2}$ and $\overline{C_3}$ count the number of instances of different subgroups generated by the synthesis algorithm. The C_{gen} counter for each sample d_i in the sliding window W keeps track of the number of times d_i is used to generate synthetic samples.

FS^2 tracks instances of different classes. After new samples are continuously collected, the number of samples of different classes in W can deviate significantly. In this case, the representation of different subgroups in W may change, and FS^2 will synthesize new samples to balance the representation of different subgroups. In line 5, the variable *adwin* is a change detector that ensures the two windows (W, W_{label}) remain consistent with any concept drift by using the class value of the incoming sample. *adwin* is used later for adapting the model to concept and/or fairness drift as needed. In line 6, the variables *balR*, $fairR$ are initialized to 0.

For each new instance d_i (line 7), we first train the pipeline learner l using the new sample (line 8). After this process, the new sample d_i is saved in the sliding window W (line 9). The function $UpdateWindows$ adds a new bit into W_{label} depending on the new sample's class label Y (i.e., 0 if $Y = 0$). In line 11, the function $updateCounters$ increments the relative counter C_0, C_1, C_2, or C_3 depending on the sample's class label Y and sensitive attribute S.

After that, the *adwin* is updated with the sensitive attribute S and class value Y of the incoming sample d_i (line 12), and the $checkDrift$ function uses *adwin* to check for any concept or fairness drift (line 13). The function adapts by reducing W following the window maintained by *adwin* to the occurrence of concept drift or fairness drift. Additionally, it updates W_{label} and the counters C_0, C_1, C_2 and C_3 based on this reduction. Furthermore, if the instances removed from W have been used to create synthetic samples, the function also updates the counter of instances generated for that class ($\overline{C_0}$, $\overline{C_1}$, $\overline{C_2}$ or $\overline{C_3}$). Line 14 determines the actual representation of each subgroup and then saves the associated window and counters into W_{syn}, C_{syn}, and $\overline{C_{syn}}$.

Then, we evaluate whether the instances in $W_{synthetic}$ satisfy or surpass the hyperparameter *minSize* set by the user (line 15). *minSize* denotes the minimum number of data samples already seen before rebalancing can occur. For example, if $minSize = 5$, we must observe at least 5 samples before rebalancing. Thus, the rebalancing starts if both criteria are met; otherwise, the algorithm waits for further samples. The imbalance ratio between the number of minority and majority instances is calculated according to Equation 1, whereas Equation 2 is used to compute the statistical parity difference. Then, the results are stored in *balR* and $fairR$, respectively (line 16). Next, we present the rebalancing task of the algorithm.

W is unbalanced and unfair if *balR* and $fairR$ are less than p_1 and f_1, respectively. Therefore, new synthetic samples $\hat{d_i}$ are generated using the $FairGenerate$ until W becomes fair and balanced (line 18). Specifically, we stop synthesizing samples when *balR* and $fairR$ reach p_1 and f_1, respectively. This process of generating samples is better than SMOTE in terms of fairness.

Before generating new data points, FS^2 calculates the number of instances introduced for each batch subgroup. Then, it clusters the instances in sliding window W, identifies similar individuals through the feature space, filters the boundary samples, and calculates the synthetic weights for the different clusters. FS^2 randomly selects samples in each cluster to synthesize new instances d_i and updates the counter S_{gen} until the synthesis number of that cluster is reached the required value. Also, we deploy the one-time principle, i.e., a sample is used only once to synthesize instances during the rebalancing phase to avoid overfitting. The algorithm violates the one-time principle only when the number of synthesized instances needed to rebalance W is larger than the number of samples. In that case, the function reuses the same samples multiple times during the same rebalancing phase. Finally, we use the new synthesized instances d_i to train the learner l (line 19) and calculate the new *balR* and $fairR$ (line 21).

5.2.4 FS^2 *Classification.* FS^2 is a meta-strategy that solves class imbalance and cumulative discriminatory results in data streams. In particular, we have integrated the Adaptive Hoeffding Trees (AHT) [4] classifier. AHT is a stream-based decision tree induction algorithm that ensures the tree's adaptation to changes in the underlying data distribution by updating the tree with new instances from the stream and replacing underperforming sub-trees. Therefore, for the online setting, AHT is superior to the majority of other classifiers. However, FS^2 can be pipelined with any data stream classifier.

6 EXPERIMENTAL EVALUATION

6.1 Datasets

The lack of access to good datasets hinders online fairness research—the large amount of data required and the necessity for concept drift to be present limit the availability. We use three real and one synthetic dataset (see Table (1 for details). i) Bank Telemarketing Data (BNK) [30]: the objective of the classification task here is to determine whether a client will subscribe to a term deposit. ii) Give Me Some Credit Dataset (GMSC) [22]: to determine whether to approve a loan application. iii) NYPD [33]: we use this dataset to predict whether a suspect was convicted of a felony. iv) Synthetic: we followed the initialization process by authors [24], which involves generating each attribute as a different Gaussian distribution. To simulate class imbalance and concept drift, we introduced these elements into the data stream by shifting the mean of the average of each Gaussian distribution.

Algorithm 1: FS^2 Leaning Algorithm

```
Input: a discriminated data stream $D$, the pipeline learner $l$,
       optional sampling ratio $p_1$ and $f_1$
$W, W_{label} \leftarrow \emptyset$;
$C_0, C_1, C_2, C_3 \leftarrow 0$;
$\overline{C_0}, \overline{C_1}, \overline{C_2}, \overline{C_3} \leftarrow 0$;
$C_{gen} \leftarrow \emptyset$;
$adwin \leftarrow \emptyset$;
$balR, fairR \leftarrow 0$;
for each instance $d_i$ in $D$ do
    $Train\ (d_i, l)$;
    $W \leftarrow Add\ (d_i)$;
    $UpdateWindows\ (d_i, W_{label})$;
    $C_0, C_1, C_2, C_3 \leftarrow updateCounters\ (d_i)$;
    $adwin \leftarrow Add\ (d_i)$;
    $CheckDrift\ (adwin, W, W_{label}, C_1, C_2, C_3, C_4, \overline{C_0}, \overline{C_1}, \overline{C_2}, \overline{C_3}, C_{gen})$;

    $W_{syn}, C_{syn}, \overline{C}_{syn} \leftarrow$
      $CheckRep\ (C_1, C_2, C_3, C_4, \overline{C_0}, \overline{C_1}, \overline{C_2}, \overline{C_3})$;
    if CheckSize(minSize, $C_{syn}$) then
        $balR, fairR \leftarrow$
          $Calculate\ (C_1, C_2, C_3, C_4, \overline{C_0}, \overline{C_1}, \overline{C_2}, \overline{C_3})$ following
          Equation (1) and Equation (2);
        while $f_1 > fairR$ or $p_1 > balR$ do
            $\hat{d_i} \leftarrow FairGenerate(W_{syn}, S_{gen})$;
            $Train\ (\hat{d_i}, l)$;
            $\overline{C}_{syn} += 1$;
            $balR, fairR \leftarrow$
              $Calculate\ (C_1, C_2, C_3, C_4, \overline{C_0}, \overline{C_1}, \overline{C_2}, \overline{C_3})$
              following Equation (1) and Equation (2);
        end
    end
end
```

Table 1: Summary of the dataset used in the evaluations.

Dataset	Sample#	Features#	Sensitive Attribute
BNK	52944	150	Age
GMSC	150,000	29	Age
NYPD	311,367	16	Gender
SYN	1,000,000	21	Synth

6.2 Baselines and Metrics

We compare FS^2 against four state-of-the-art online fairness streaming methods based on four metrics. The four baselines are: i) Fairness-aware Hoeffding Tree [49] (FAHT): FAHT is a fairness-focused variation of the Hoeffding tree algorithm considering both performance and fairness when selecting split attributes, ii) Online Smooth Boosting [39] (OSBoost): it is made cost-sensitive by adding various parameters of the Poisson distribution for different classes, iii) Online fairness and class imbalance-aware boosting [23] (FABBOO): it considers long-term class imbalance to fight class imbalance and discrimination from a boosting approach, and iv) Continuous SMOTE [3] (C-SMOTE): is an approach that addresses class imbalance by resampling the minority class using a sliding window without fairness.

We use two performance and two fairness metrics to evaluate the proposed method. i) *Balanced Accuracy* is the arithmetic mean of Specificity (True Negative Rate) and Sensitivity (True Positive Rate). ii) *Recall* measures the proportion of actual positive cases the model correctly identified. iii) *Cumulative Statistical Parity Difference (CumSPD)* is a measure of the cumulative statistical parity difference between the protected and deprived groups (Equation 2), a value of 0 represents fairness. iv) *Cumulative Equal Opportunity Difference (CumEOD)* is a measure of cumulative equal opportunity difference between the privileged and unprivileged groups (Equation 3).

6.3 Experimental Results

6.3.1 Effect of varying window size. We first evaluated the impact of various window sizes on the proposed solution using sizes 500, 1000, 2000, 5000, and 10000. We present the results of Balanced Accuracy for the predictive performance and CumSPD for fairness. As Figure 5 shows, we observe that when the window size surpasses 2000 both Balanced Accuracy and CumSPD remain constant. This behavior is due to the occurrence of concept drift; the large sliding window (window size > 2000) can not discard outdated instances before the concept drift, while the smaller window (size = 500 or 1000) can accommodate newer instances after concept drift and discards the older ones. Generally, smaller window settings yield better fairness but at the cost of prediction performance degradation. Conversely, excessively large window settings can impede the model's ability to adjust decision boundaries during concept drift, thus negatively affecting model fairness.

6.3.2 Influence of different decay factors. In this section, we analyze the effect of varying decay factors on model fairness and performance in diverse data flow variations (Figure 4). First, we consider *fixed class-imbalance*, i.e., the class ratio is fixed over the data stream. The low decay factor values negatively impact model performance, with Recall dropping below 60% for a decay factor of < 0.4. However, for the decay factor of ≥ 0.5, Balanced Accuracy and Recall do not significantly change. Furthermore, the model's capacity to alleviate unfair outcomes is unchanged for all decay factor values. Next, we consider increasing and decreasing class imbalance and observe a steady rise in Recall as the decay factor increases. Finally, in the case of fluctuating class imbalance, low values of the decay factor decrease the performance as in previous cases, while high values also impact minority classes. In this case, the positive class alternates between minority and majority. High decay factor values allow the model to consider more history and lead to class imbalance weights assigned by FS^2 not being adjusted to recent changes. Overall, the value of the decay factor directly affects the performance of FS^2. Low values of the decay factor produce fluctuating class imbalance weights that decrease the model's performance, and in some cases, values close to 1 are also not appropriate.

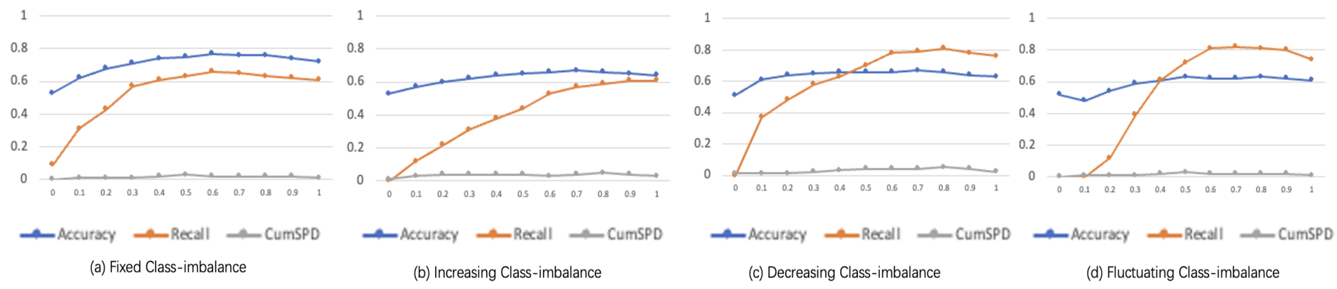

Figure 4: The impact of decay factor on the synthetic dataset of varying class imbalance.

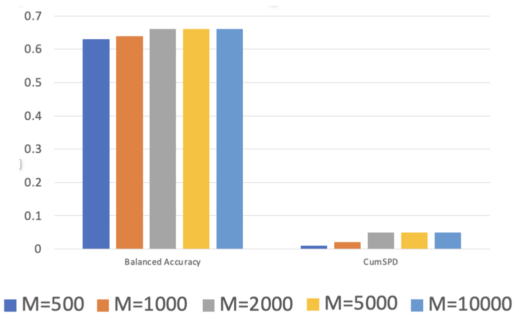

Figure 5: The impact of window size on model performance and fairness.

6.3.3 FS^2 compared to state-of-the-art methods. We compare the performance-fairness of the proposed method (FS^2) with the four baselines. The results are demonstrated in Table 2; the darker hue of red denotes the best model performance, followed by a lighter red-pink shade as the second-best performance. FS^2 performs the best in 12 of the 16 performance-fairness evaluations, and it closely followed the top performer (with a margin of 1-3%) in the remaining 4 measurements. One of the key reasons for this superior performance is the design of our approach, which is specifically tailored to handle both fairness and concept drift. This adaptability gives FS^2 a significant edge over other models that are not equipped to manage these challenges effectively.

In the BNK dataset, FS^2 achieved a maximum Balanced Accuracy of 81%, surpassing the powerful C-SMOTE. The cumulative EOD and SPD are the best for FS^2 with a 0.02 and 0.01 score, respectively, while its recall score of 0.78 is the second highest. FS^2 outperforms existing models on the GMSC dataset regarding Balanced Accuracy, Cumulative EOD, and SPD with scores of 0.83, 0.01, and 0.01. It is the second-best in Recall, with a score of 0.78. Similarly, for the NYPD dataset, FS^2 surpasses all four baseline methods as it offers the best score for all the fairness-performance metrics, i.e., Balanced Accuracy (0.68), Recall (0.54), Cumulative SPD (0.06), and Cumulative EOD (0.03). Finally, in the case of the SYN dataset, FS^2 performs fairly well–it is the second best in Balanced Accuracy and Recall. But it outshines in fairness as it scored a cumulative SPD (0.04) and OPD (0.06). Overall, FS^2 demonstrates superior performance compared to other methods in terms of both accuracy and fairness.

6.3.4 The effectiveness analysis. With the proposed FBU, we can now evaluate the trade-off between fairness and performance based on the proposed metric. Figure 6 shows the overall results. As we can see, FS^2 achieves a Jointly advantageous or Impressive trade-off (i.e., it beats the trade-off baseline constructed by FBU) in most cases, i.e., 79% of the time. The corresponding percentages for FATH, OSBoost, FABBOO, and C-SMOTE were 32%, 25%, 58%, and 31%, respectively. In addition, FS^2 has significantly fewer Jointly disadvantageous trade-off cases (just 1%) than other existing methods. For example, FATH has a Jointly disadvantageous trade-off rate of 15%, 15 times higher than that of FS^2. Overall, FS^2 achieves the best trade-off, significantly improving over all existing methods. The obtained results in Figure 6 are also consistent with the results in Table 2, but are presented in a unified manner that is easy to interpret, verifying the theoretical design of FBU.

6.3.5 Concept and fairness drift detection. We tracked the variation of the model performance on the synthetic dataset using FBU, a choice dictated by the unique advantages of synthetic data. Real-world datasets often fail to provide enough samples for a comprehensive evaluation, while synthetic data allow us to increase the sample size and achieve a more robust evaluation. Furthermore, synthetic data enable precise control over key factors such as fairness and concept drift, providing a clearer picture of our model's response to these variables. The results are shown in Figure 7. Our approach (the orange curve) is designed for streaming data and can adapt and regain its performance after a concept drift occurs (the curve first goes down and then up). Since we can handle fairness drifts, we observe that our model can adapt to both fairness and concept drift. For example, C-SMOTE has a significant decrease in the model's overall performance because of the lack of detection of fairness drift. At the same time, the intuition and validity of FBU are well represented. Note that without FBU we would need to show the changes in performance and fairness separately, complicating the analysis of the specific impact trade-off between them.

7 CONCLUSION

Offering fair predictions in the presence of a continuous and infinite stream of data remains an open challenge in streaming machine learning applications. In this work, we proposed Fair Bonded Utility (FBU), the first fairness metric to unify the comparison of the fairness-performance trade-offs of multiple fairness techniques into one that simplifies the task of choosing a fairness scheme. Then,

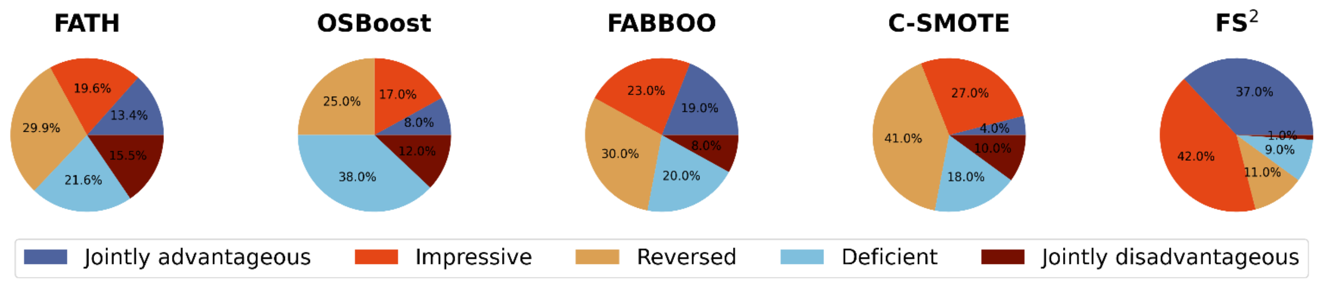

Figure 6: The effectiveness distribution of various data handling techniques, including FS^2, FATH, OSBoost, FABBOO, and C-SMOTE has been evaluated using the FBU. FS^2 proves to be the most effective, with 79% of the cases falling into the "Jointly advantageous" and "Impressive" categories.

Table 2: The predictive and fairness performance comparison.

Dataset	Sensitive Attribute	Methods	Balanced Accuracy	Recall	Cumulative SPD	Cumulative EOD
BNK	Age	FAHT	0.73	0.52	0.16	0.18
		OSBoost	0.73	0.71	0.18	0.19
		FABBOO	0.76	0.54	0.05	0.08
		C-SMOTE	0.80	0.81	0.34	0.14
		FS^2	0.81	0.78	0.02	0.01
GMSC	Age	FAHT	0.62	0.28	0.02	0.04
		OSBoost	0.65	0.57	0.04	0.06
		FABBOO	0.81	0.76	0.02	0.01
		C-SMOTE	0.80	0.80	0.07	0.02
		FS^2	0.83	0.79	0.01	0.01
NYPD	Gender	FAHT	0.55	0.28	0.17	0.28
		OSBoost	0.52	0.07	0.25	0.15
		FABBOO	0.62	0.50	0.07	0.06
		C-SMOTE	0.56	0.42	0.34	0.16
		FS^2	0.68	0.54	0.06	0.03
SYN	synth.	FAHT	0.62	0.28	0.08	0.16
		OSBoost	0.57	0.31	0.08	0.11
		FABBOO	0.67	0.58	0.09	0.07
		C-SMOTE	0.67	0.62	0.26	0.16
		FS^2	0.65	0.61	0.04	0.05

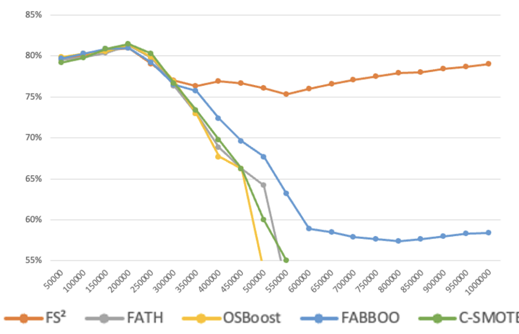

Figure 7: The performance evaluation of concept and fairness drift.

we presented Fair Sampling over Stream (FC^2), a novel fair class-balancing technique capable of handling continuous data streams with concept and fairness drift. Extensive evaluations show our approach gives superior fairness results according to multiple widely-used fairness metrics without sacrificing performance. The pre-processing nature of our method can improve performance and fairness in different applications due to the ease of integration.

REFERENCES

[1] Charu C Aggarwal. 2007. *Data streams: models and algorithms*. Vol. 31. Springer Science & Business Media.

[2] Yahav Bechavod, Christopher Jung, and Steven Z Wu. 2020. Metric-free individual fairness in online learning. *Advances in neural information processing systems* 33 (2020), 11214–11225.

[3] Alessio Bernardo, Heitor Murilo Gomes, Jacob Montiel, Bernhard Pfahringer, Albert Bifet, and Emanuele Della Valle. 2020. C-smote: Continuous synthetic minority oversampling for evolving data streams. In *2020 IEEE International Conference on Big Data (Big Data)*. IEEE, 483–492.

[4] Albert Bifet and Ricard Gavalda. 2009. Adaptive learning from evolving data streams. In *Advances in Intelligent Data Analysis VIII: 8th International Symposium on Intelligent Data Analysis, IDA 2009, Lyon, France, August 31-September 2, 2009. Proceedings 8*. Springer, 249–260.

[5] Albert Bifet and Ricard Gavaldà. 2007. Learning from Time-Changing Data with Adaptive Windowing. *Proceedings of the 7th SIAM International Conference on Data Mining* 7. https://doi.org/10.1137/1.9781611972771.42

[6] Toon Calders, Faisal Kamiran, and Mykola Pechenizkiy. 2009. Building classifiers with independency constraints. In *2009 IEEE international conference on data mining workshops*. IEEE, 13–18.

[7] Toon Calders and Sicco Verwer. 2010. Three naive bayes approaches for discrimination-free classification. *Data mining and knowledge discovery* 21 (2010), 277–292.

[8] Joymallya Chakraborty, Suvodeep Majumder, and Tim Menzies. 2021. Bias in machine learning software: Why? how? what to do?. In *Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. 429–440.

[9] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. 2002. SMOTE: synthetic minority over-sampling technique. *Journal of artificial intelligence research* 16 (2002), 321–357.

[10] Shang-Tse Chen, Hsuan-Tien Lin, and Chi-Jen Lu. 2012. An online boosting algorithm with theoretical justifications. *arXiv preprint arXiv:1206.6422* (2012).

[11] Kate Crawford. 2016. Artificial intelligence's white guy problem. *The New York Times* 25, 06 (2016), 5.

[12] Amit Datta, Michael Carl Tschantz, and Anupam Datta. 2014. Automated experiments on ad privacy settings: A tale of opacity, choice, and discrimination. *arXiv preprint arXiv:1408.6491* (2014).

[13] Georgios Douzas, Fernando Bacao, and Felix Last. 2018. Improving imbalanced learning through a heuristic oversampling method based on k-means and SMOTE. *Information Sciences* 465 (2018), 1–20.

[14] Zichong Wang et al. 2023. FG^{2}AN: Fairness-aware Graph Generative Adversarial Networks.

[15] Joao Gama. 2010. *Knowledge discovery from data streams*. CRC Press.

[16] João Gama, Indrė Žliobaitė, Albert Bifet, Mykola Pechenizkiy, and Abdelhamid Bouchachia. 2014. A survey on concept drift adaptation. *ACM computing surveys (CSUR)* 46, 4 (2014), 44.

[17] Adel Ghazikhani, Reza Monsefi, and Hadi Sadoghi Yazdi. 2014. Online neural network model for non-stationary and imbalanced data stream classification. *International Journal of Machine Learning and Cybernetics* 5 (2014), 51–62.

[18] Heitor M Gomes, Albert Bifet, Jesse Read, Jean Paul Barddal, Fabrício Enembreck, Bernhard Pfharinger, Geoff Holmes, and Talel Abdessalem. 2017. Adaptive random forests for evolving data stream classification. *Machine Learning* 106 (2017), 1469–1495.

[19] Bogdan Gulowaty and Paweł Ksieniewicz. 2019. SMOTE algorithm variations in balancing data streams. In *Intelligent Data Engineering and Automated Learning–IDEAL 2019: 20th International Conference, Manchester, UK, November 14–16, 2019, Proceedings, Part II 20*. Springer, 305–312.

[20] Sara Hajian, Francesco Bonchi, and Carlos Castillo. 2016. Algorithmic bias: From discrimination discovery to fairness-aware data mining. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. 2125–2126.

[21] Moritz Hardt, Eric Price, and Nati Srebro. 2016. Equality of opportunity in supervised learning. *Advances in neural information processing systems* 29 (2016).

[22] https://www.kaggle.com/c/GiveMeSomeCredit/overview. 2021. Give Me Some Credit. https://doi.org/10.34740/KAGGLE/DSV/2242482

[23] Vasileios Iosifidis and Eirini Ntoutsi. 2020. -Online Fairness-Aware Learning Under Class Imbalance. In *Discovery Science: 23rd International Conference, DS 2020, Thessaloniki, Greece, October 19–21, 2020, Proceedings*. Springer, 159–174.

[24] Vasileios Iosifidis, Thi Ngoc Han Tran, and Eirini Ntoutsi. 2019. Fairness-enhancing interventions in stream classification. In *Database and Expert Systems Applications: 30th International Conference, DEXA 2019, Linz, Austria, August 26–29, 2019, Proceedings, Part I 30*. Springer, 261–276.

[25] Faisal Kamiran and Toon Calders. 2009. Classifying without discriminating. In *2009 2nd international conference on computer, control and communication*. IEEE, 1–6.

[26] Faisal Kamiran and Toon Calders. 2012. Data preprocessing techniques for classification without discrimination. *Knowledge and information systems* 33, 1 (2012), 1–33.

[27] Faisal Kamiran, Toon Calders, and Mykola Pechenizkiy. 2010. Discrimination aware decision tree learning. In *2010 IEEE international conference on data mining*. IEEE, 869–874.

[28] Bartosz Krawczyk, Leandro L Minku, João Gama, Jerzy Stefanowski, and Michał Woźniak. 2017. Ensemble learning for data stream analysis: A survey. *Information Fusion* 37 (2017), 132–156.

[29] Nicol Turner Lee, Paul Resnick, and Genie Barton. 2019. Algorithmic bias detection and mitigation: Best practices and policies to reduce consumer harms. *Brookings Institute: Washington, DC, USA* 2 (2019).

[30] Sérgio Moro, Paulo Cortez, and Paulo Rita. 2014. A data-driven approach to predict the success of bank telemarketing. *Decision Support Systems* 62 (2014), 22–31. https://doi.org/10.1016/j.dss.2014.03.001

[31] Marcelo OR Prates, Pedro H Avelar, and Luís C Lamb. 2020. Assessing gender bias in machine translation: a case study with google translate. *Neural Computing and Applications* 32 (2020), 6363–6381.

[32] Jesse Read, Nikolaos Tziortziotis, and Michalis Vazirgiannis. 2019. Error-space representations for multi-dimensional data streams with temporal dependence. *Pattern Analysis and Applications* 22, 3 (2019), 1211–1220.

[33] John F Sargent and Dana A Shea. 2017. *Office of Science and Technology Policy (OSTP): History and Overview*. Congressional Research Service.

[34] Nripsuta Ani Saxena, Wenbin Zhang, and Cyrus Shahabi. 2023. Missed Opportunities in Fair AI. In *Proceedings of the 2023 SIAM International Conference on Data Mining (SDM)*. SIAM, 961–964.

[35] Gaganpreet Sharma. 2017. Pros and cons of different sampling techniques. *International journal of applied research* 3, 7 (2017), 749–752.

[36] Zeyu Sun, Jie M Zhang, Mark Harman, Mike Papadakis, and Lu Zhang. 2020. Automatic testing and improvement of machine translation. In *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. 974–985.

[37] Latanya Sweeney. 2013. Discrimination in online ad delivery. *Commun. ACM* 56, 5 (2013), 44–54.

[38] Sahil Verma and Julia Rubin. 2018. Fairness definitions explained. In *Proceedings of the international workshop on software fairness*. 1–7.

[39] Boyu Wang and Joelle Pineau. 2016. Online bagging and boosting for imbalanced data streams. *IEEE Transactions on Knowledge and Data Engineering* 28, 12 (2016), 3353–3366.

[40] Qianwen Wang, Zhenhua Xu, Zhutian Chen, Yong Wang, Shixia Liu, and Huamin Qu. 2020. Visual analysis of discrimination in machine learning. *IEEE Transactions on Visualization and Computer Graphics* 27, 2 (2020), 1470–1480.

[41] Shuo Wang, Leandro L Minku, and Xin Yao. 2013. A learning framework for online class imbalance learning. In *2013 IEEE Symposium on Computational Intelligence and Ensemble Learning (CIEL)*. IEEE, 36–45.

[42] Shuo Wang, Leandro L Minku, and Xin Yao. 2014. Resampling-based ensemble methods for online class imbalance learning. *IEEE Transactions on Knowledge and Data Engineering* 27, 5 (2014), 1356–1368.

[43] Zichong Wang, Yang Zhou, Meikang Qiu, Israat Haque, Laura Brown, Yi He, Jianwu Wang, David Lo, and Wenbin Zhang. 2023. Towards Fair Machine Learning Software: Understanding and Addressing Model Bias Through Counterfactual Thinking. *arXiv preprint arXiv:2302.08018* (2023).

[44] Gary M Weiss. 2004. Mining with rarity: a unifying framework. *ACM Sigkdd Explorations Newsletter* 6, 1 (2004), 7–19.

[45] Shen Yan, Hsien-te Kao, and Emilio Ferrara. 2020. Fair class balancing: Enhancing model fairness without observing sensitive attributes. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 1715–1724.

[46] Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez-Rodriguez, and Krishna P Gummadi. 2019. Fairness constraints: A flexible approach for fair classification. *The Journal of Machine Learning Research* 20, 1 (2019), 2737–2778.

[47] Wenbin Zhang, Tina Hernandez-Boussard, and Jeremy C Weiss. 2023. Censored Fairness through Awareness. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

[48] Wenbin Zhang, Juyong Kim, Zichong Wang, Pradeep Ravikumar, and Jeremy Weiss. 2023. Individual Fairness Guarantee in Learning with Censorship. *arXiv preprint arXiv:2302.08015* (2023).

[49] Wenbin Zhang and Eirini Ntoutsi. 2019. FAHT: an adaptive fairness-aware decision tree classifier. *arXiv preprint arXiv:1907.07237* (2019).

[50] Wenbin Zhang and Jianwu Wang. 2017. A hybrid learning framework for imbalanced stream classification. In *2017 IEEE International Congress on Big Data (BigData Congress)*. IEEE, 480–487.

[51] Wenbin Zhang and Jeremy C Weiss. 2021. Fair decision-making under uncertainty. In *2021 IEEE International Conference on Data Mining (ICDM)*. IEEE, 886–895.

[52] Wenbin Zhang and Jeremy C Weiss. 2022. Longitudinal fairness with censorship. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 36. 12235–12243.

[53] Indre Zliobaite. 2015. A survey on measuring indirect discrimination in machine learning. *arXiv preprint arXiv:1511.00148* (2015).