Generating Fair Tabular Data using Generative Adversarial Networks

Kyriakos Aristides

Master of Science
Data Science
School of Informatics
University of Edinburgh
2019

Abstract

The use of automated decision making systems has a wide range of applications. From movie and TV recommender systems to legal, financing, health and employment. These systems are replacing humans and their final decisions are often accepted without any human intervention. Because these systems are trained to make decisions with historical data which could possibly contain past decisions that placed certain groups at a disadvantage they could unintentionally be unfair. This realization has led to an increased popularity of the topic of fairness in Machine Learning as scientists understand the importance of accounting for and avoiding unfairness. For the purposes of this project I proposed a Generative Adversarial Network which can take any tabular dataset with continuous and discrete attributes and a binary output label and produce a fair synthetic dataset based on a chosen binary sensitive attribute which when used in Machine Learning algorithms can produce classification results that are as accurate as the original. By the end of the project my final model was able to produce a fair synthetic dataset based on the original training dataset which had almost the same training utility of the original when training a classifier to predict classes on the original test set.

Acknowledgements

Immense gratitude is due to my supervisors, John Pate and Tania Bakhos who were there to support me during the project proposal process and the project itself, answering all my questions and providing useful insights. Also, to my friends and coworkers here in Edinburgh with whom I shared my struggles and finally my family who always believes in me and pushes me to pursue my dreams.

Table of Contents

Chapter 1

Introduction

The topic of fairness in Machine Learning has become increasingly popular over the past few years. There has been an increased research interest among the scientific community on matters of measuring and removing unfairness from algorithmic models. New papers on the topic are published on a weekly basis [1]. These are an indication of the presence of the topic of fairness in the forefront of research in the scientific community.

Apart from research, fairness in Machine Learning could pose an issue in our every day lives due to the increased presence of automated decision making systems that are driven by historically collected data. Such systems are being used to decide whether you could be getting a loan, a job, accepted at a University and they are trained using these historical data to minimize their misclassification errors. The problem with such systems regarding fairness is that their decisions could put people that belong in specific groups regarding gender, race, age, religion or sexual orientation at a disadvantage as a result of misclassification of more people with sensitive values under these groups. For example, more women could be classified as not suitable for a job than men without any other reason apart from the fact that they are women [2]. These issues could arise unintentionally since the historical data we use to train such classifiers could be inherently biased towards certain groups.

The implementation of the General Data Protection Regulation[1] (GDPR) in May 25 2018 which limits the circumstances in which decision making that has legal or similarly significant effect on individuals is solely automated is an indication that classifiers should not be trained with a purely performance related objective but also with a fairness one.

[1] https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A32016R0679

According to a 2017 survey on the state of Data Science and Machine Learning by Kaggle, relational data is the most popular type of data used in business industries [3]. Their role in business is integral as they can provide valuable intelligence and insight through the application of Machine Learning and as things stand, they have taken oil's position as the most valuable resource [4]. The importance and relevance of tabular data and their wide usage in automated decision making systems has motivated their usage in this project.

Generative Adversarial Networks (GAN) [5] can be used to generate such tabular data by implicitly learning the probability distribution of the given dataset and then drawing samples for it. This happens by pitting a generator that generates data against a discriminator that recognizes fake data with the purpose of training the generator to produce synthetic data that the discriminator recognizes as real. The model built for the purposes of this project is based on this notion with the addition of a fairness component working in parallel with the discriminator. The motivation for creating such a GAN model is that GANs have appealing performance capabilities and design flexibility and the datasets generated from it can be applied immediately in any classification task.

The structure of the thesis is as follows: In Chapter 2 the background of the project is set. This involves Generative Adversarial Networks and the several fairness notions used in the project. The next chapter focuses on the details of the Tabular GAN (TGAN) model, its function, how it handles continuous and discrete data and how it generates data using a generator, discriminator and relevant loss function as well as the Fair TGAN model built for the project and how it is used in the context of fairness and classification. Chapter 4 describes the dataset used in the project as well as the model training setup and the subsequent classification and fairness tests used to analyse the results. Chapter 5 presents the results of different models that were trained during the project, explains the process which led to training each model and analyses and compares the results. The final chapter outlines the conclusions from the project and sets the future work that is possible on the topic of generating fair datasets using GANs.

Chapter 2

Background

The background section goes over the integral parts of the project. These are the Generative Adversarial Networks (GAN), the tabular GANs used to generate tabular data, how they can be used in a fairness context and the various relevant notions of fairness.

2.1 Generative Adversarial Networks

GANs are a type of generative model which tries to build a probabilistic model of each class in the data. Essentially, they are structured probabilistic models. In simple terms, the purpose of GANs is to estimate the distribution of the input data and then use that distribution to generate synthetic samples that are similar to the input data [6].

A GAN consists of a generator and a discriminator. They are represented by their own differentiable functions with respect to their inputs and parameters. Generator function G takes as input a latent variable z and is parameterized by ϕ. Discriminator function D takes as input observed variables x and uses θ as parameters. Each of them tries to minimize their respective cost function as defined by their parameters. The generator's cost function is $L^G(\theta, \phi)$ and the generator's cost function is $L^D(\theta, \phi)$. The noticeable difference from any other optimization problem is that both cost functions depend on parameters of both the generator and discriminator while each of them can control only their own. So, instead of an optimization problem with a local minimum solution we have a minimax game which is solved by the Nash Equilibrium. The Nash Equilibrium is a tuple with parameter values for both cost functions and is also a local minimum for both of them with respect to their parameter values [7]. The minimax game that the GAN ultimately tries to solve is to minimize the generator cost function and maximize the discriminator cost function so as to learn to generate fake

data that fool the discriminator. Since the generator loss function is tied to that of the discriminator this game can be expressed as a value function of the discriminator's payoff as $V(\theta,\phi) = -L^D(\theta,\phi)$ with an optimization functions as such [6]:

$$\phi^* = \arg\min_{\phi}\max_{\theta} V(\theta,\phi)$$

2.1.1 Tabular Generative Adversarial Networks

While the majority of recent research on GANs focuses on images and video ([8], [9]) the popularity of tabular data warrants the development of models such as the ones used in this project. Several different models have been built with the purpose of creating synthetic tabular datasets. Howe *et al.* created DataSynthezier [10] which generates fake data using differential privacy mechanisms [11]. Zhang *et al.* [12] and Dong *et al.* [13] use Bayesian Networks. Several GAN examples deal with medical tabular data ([14], [15]) but do not solve the problem of generating multimodal continuous variables.

Two recent GAN models that generate synthetic tabular data have been found in the literature. The first called TableGAN by Park *et al.* [16] uses convolutional neural networks for its generator and optimizes prediction accuracy by minimizing cross entropy loss. The second model is called Tabular GAN (TGAN), by Xu and Veeramachaneni [17] and it learns the marginal distribution of each feature in the original dataset by minimizing the Kullback-Leibler (KL) divergence. This project utilizes the latter model to build a new Fair TGAN model.

2.2 Fairness Notions

Several notions of fairness have been proposed in the literature. A simple distinction between such notions could be given based on how fairness is measured. This could be either by groups called group-based fairness where different groups are distinguished between different values of a chosen binary sensitive variable or on an individual basis where individuals that have similar features should receive the same treatment. Notions that fall under group-based fairness are **Disparate Impact**, **Disparate Mistreatment**, **Equal Opportunity** and **Equalized Odds**. The notation used in the following fairness notions is: s for the sensitive attribute, y for the true output label and $\hat{y}$ for the label prediction.

2.2.1 Disparate Impact

Under the doctrine of disparate impact ([18], [19], [20]) a decision making process suffers from it if its outcome disproportionately hurts or benefits a group of people with a specific sensitive attribute value. According to it and assuming without loss of generality a binary sensitive attribute and binary label, fairness can be achieved when the following equation stands [21]:

$$\mathrm{p}(\hat{y} = 1|s = 0) = \mathrm{p}(\hat{y} = 1|s = 1)$$

This equation shows that positive predictions (usually $\hat{y} = 1$) for both groups of the sensitive attribute should be equal.

2.2.2 Disparate Mistreatment

The notion of disparate mistreatment as formalized by Zafar *et al.* [2] differs from disparate impact in that it depends on the ground truth label y instead of just the prediction. Assuming a binary label y which can take truth values of 1 and -1 then disparate mistreatment will arise if misclassification rates are disproportionate between different subgroups with different sensitive attribute values. The different misclassification rates are presented in the confusion matrix in table 2.1, adapted from Zafar *et al.* [2]. These misclassification rates can be measured as fractions over class distribution in both the ground truth y and predicted $\hat{y}$ labels. As a result, disparate mistreatment is absent from binary classification when there is equality between them for different values of the sensitive attribute as follows:

Overall Misclassification Rate (OMR):

$$\mathrm{p}(\hat{y} \neq y|s = 0) = \mathrm{p}(\hat{y} \neq y|s = 1)$$

False Positive Rate (FPR):

$$\mathrm{p}(\hat{y} \neq y|y = -1, s = 0) = \mathrm{p}(\hat{y} \neq y|y = -1, s = 1)$$

False Negative Rate (FNR):

$$\mathrm{p}(\hat{y} \neq y|y = 1, s = 0) = \mathrm{p}(\hat{y} \neq y|y = 1, s = 1)$$

		Predicted Label $\hat{y} = 1$	Predicted Label $\hat{y} = -1$	
True Label	$y = 1$	True Positive	False Negative	$P(\hat{y} \neq y \mid y = 1)$ False Negative Rate
	$y = -1$	False Positive	True Negative	$P(\hat{y} \neq y \mid y = -1)$ False Positive Rate

Table 2.1: Table of misclassification rates based on predicted and true labels adapted from Zafar *et al.* [2].

2.2.3 Equal Opportunity

The notion of equal opportunity is represented by the conditional independence between a chosen sensitive attribute and the class predictor given the true positive outcome. When this independence holds then the notion is satisfied or in other words the predictor satisfies equalized odds with respect to the sensitive attribute and true positive outcome. In a loan application setting for example this would mean that people who successfully applied for a loan had an equal opportunity of getting the loan in the first place while those who didn't get the loan don't need to satisfy any non-discriminatory requirement [22]. Hardt *et al.* provide the following equation for equal opportunity:

$$\mathrm{p}(\hat{y} = 1 | s = 0, y = 1) = \mathrm{p}\ (\hat{y} = 1 | s = 1, y = 1)$$

2.2.4 Equalized Odds

Equalized odds is a stricter version of equal opportunity in that it requires non-discrimination within both output classes. If the class predictor and a chosen sensitive attribute are conditionally independent from the true outcome then according to Hardt *et al.* [22] the predictor satisfies equalized odds with respect to the sensitive attribute and true outcome. Essentially, when the true outcome is known, knowing the sensitive attribute does not provide any additional information when predicting the class predictor. Using a binary sensitive attribute and class variable the formulation of equalized odds is:

$$\mathrm{p}(\hat{y} = 1 | s = 0, y) = \mathrm{p}(\hat{y} = 1 | s = 1, y), \quad y \in \{-1, 1\}$$

Also, under equalized odds both groups of the binary sensitive attribute must have equal True Positive Rates when $y = 1$ and equal False Positive rates when $y = -1$ in

order to enforce equal accuracy and equal bias in both groups and punish models which perform well only on the larger group [22].

2.2.5 Individual Fairness

Individual Fairness differs from the previous notions because it's emphasis lies on treating similar individuals the same way instead of just groups with same specific attribute values [23]. This notion imposes fair treatment to each pair of individuals instead of large groups and as a result is a more fine-grained fairness notion. The problem with having such a fine-grained notion though is that it is difficult to measure similarities and differences between individuals [1].

2.2.6 Fairness in Generative Adversarial Networks

Fairness can be introduced into supervised Machine Learning pipelines using three main classes of intervention [24]. These are:

1. Preprocessing ([25], [23]): modifying the dataset used for training.
2. In-processing ([26], [27]): modifying the algorithms themselves.
3. Post-processing ([22]): modifying the predictions of the algorithms.

The Fair TGAN model proposed in this project used the first two classes but in a reverse order. This model had an added fairness component working in parallel with the discriminator which was an in-process intervention. The input of this component was the generator output of generated independent attribute values excluding the generated sensitive attribute values. The reason for removing the sensitive attribute from the fairness component input was to allow this component to learn to predict the wrong sensitive attribute value from the rest of the independent attributes. The purpose of this was to render these attributes invariant to the sensitive attribute. With the introduction of this invariance the Fair TGAN model tried to alleviate **disparate impact** and **disparate mistreatment**. Finally, the output of the model was a fair synthetic tabular dataset which was used in supervised Machine Learning pipelines something that constituted a preprocessing fairness modification.

Chapter 3

Tabular Generative Adversarial Network Models

This chapter introduces the Tabular GAN (TGAN) and the Fair TGAN which is the model that was built during the project. TGAN will be referred to as the Base TGAN model when compared to the Fair TGAN model.

3.1 Base TGAN Model

The TGAN model by Xu and Veeramachaneni [17] works by learning the marginal distribution of each feature in the original dataset by minimizing the Kullback-Leibler (KL) divergence. A repository containing the fully developed TGAN model was available.[1]

3.1.1 Motivation

The reasoning behind choosing to work with the TGAN model is the fact that the authors show that the synthetic data it generates perform much better than those of similar systems ([28], [29]) when used to train algorithms such as Decision Trees, Linear Support Vector Machines, Random Forests, AdaBoost and Multi-Layer Perceptrons. Also, using their repository a TGAN model which was built on TensorFlow was successfully installed, trained and able to generate samples of datasets efficiently.

[1] https://github.com/DAI-Lab/TGAN

3.1.2 Function

The TGAN is able to generate high quality, synthetic tabular data based on a given dataset. Data generation happens by using Long-Short Term Memory (LSTM) networks, which are a special kind of recurring neural network capable of learning longtrem dependencies. These networks have attention with the purpose of generating the synthetic table column by column [17].

The input datasets for the TGAN can contain both continuous and discrete variables. These variables follow an unknown joint distribution and the table entries are independent samples of that distribution. The generative model to be learned should represent a joint distribution of all the table variables in order to allow generated samples from it to form a new synthetic table. This new synthetic table should satisfy two requirements to be acceptable according to the authors:

1. Machine Learning models trained with the real training table and the synthetic table should achieve similar accuracy on the real test table.

2. Mutual information between arbitrary pairs of variables should be similar for both real and synthetic tables.

3.1.3 Data Transformations

A series of reversible data transformation have to be applied on both continuous and discrete variables in order to allow the neural networks inside the TGAN to effectively learn the model.

3.1.3.1 Continuous Variables

Continuous variables receive mode-specific normalization because these variables could follow either unimodal or multimodal distributions. The process for normalizing continuous variables includes normalizing them to the [-1,1] range, using a `tanh` activation for feature generation and clustering their values using a Gaussian Mixture Model (GMM) with five components to model a distribution with a weighted sum of five Gaussian distributions with means $\eta_i^{(1)}, ..., \eta_i^{(5)}$ and standard deviations $\sigma_i^{(1)}, ..., \sigma_i^{(5)}$. This allows effective multimodal distribution sampling. This clustering also works for unimodal distributions as the GMM will assign very high probability to only one component. Then the probability of a continuous attribute value $c_{i,j}$ (i is attribute

and j the row) sampled from each of the five distributions is a vector of five normalized probability distributions $u_{i,j}^{(1)},..,u_{i,j}^{(5)}$ over the five Gaussian distributions. Then $c_{i,j}$ is normalized based on the mean and variance of the maximum $u_{i,j}^{(k)}$ component as $v_{i,j} = \left(c_{i,j} - \eta_i^{(k)}\right)/2\sigma_i^{(k)}$ and restricted to a $[-0.99, 0.99]$ range.

3.1.3.2 Discrete Variables

Discrete variables (d_i) are converted to one-hot-encoding representation, then noise from a uniform distribution with $\gamma = 0.2$ is added to each dimension and finally they are renormalized (d_i).

3.1.4 Model and Data Generation

The TGAN generates samples as $v_{i:n_c,j}$, $u_{i:n_c,j}$ and $\mathrm{d}_{i:n_d,j}$ (n_c and n_d represent the number of continuous and discrete variables respectively). These samples go through a simple post-processing operation which formulates $c_{i,j}$ from $u_{i,j}$ and $v_{i,j}$ ($c_{i,j} = 2v_{i,j}\sigma_i^{(k)} + \eta_i^{(k)}$, where $k = \arg_k \max u_{i,j}^{(k)}$) and picks the most probable category for discrete features $d_{i,j} \leftarrow \arg\max_k \mathrm{d}_{i,j}^{(k)}$

3.1.4.1 Generator

An LSTM network is used as a generator. A continuous value is generated by firstly generating a scalar value v_i and then generating the cluster vector u_i. A discrete value is generated as a probability distribution over all labels of that variable.

The LSTM input in each t step is the random variable z with n_z dimensions sampled from $\mathcal{N}(0,1)$, the previous hidden vector f_{t-1} or an embedding vector f'_{t-1} and the weighted context vector a_{t-1} which is a weighted average over all previous LSTM outputs. The output is h_t with n_h dimensions (same as the hidden state size) and it is projected to a hidden vector $f_t = \tanh(W_h h_t)$ with n_f dimensions. W_h is a parameter learned in the network. f_t is converted to an output variable that can be in four states:

1. Value part of a continuous variable: $v_i = \tanh(W_h f_t)$ with hidden vector f_t for step $t+1$.

2. Cluster part of a continuous variable: $u_i = \mathrm{softmax}(W_h f_t)$ with hidden vector f_t for step $t+1$.

3. Discrete variable: $\mathrm{d}_i = \mathrm{softmax}(W_t f_t)$ with hidden vector $f'_t = E_i\left[\arg_k \max \mathrm{d}_i\right]$ for step $t+1$ ($E \in \mathbb{R}^{|D_i| \times n_f}$ is an embedding matrix for discrete variable D_i)

4. f_0 is a special vector <GO> and it is learned during training.

3.1.4.2 Discriminator

The discriminator is an l-layer fully connected neural network that takes as input the concatenation $v_{i:n_c,j}$, $u_{i:n_c,j}$ and $\mathrm{d}_{i:n_d,j}$. The output of the discriminator is a scalar computed as: $W^{(D)}(f_l{}^{(D)} \oplus \text{diversity}(f_l{}^{(D)}))$. Internal layers are computed as follows:

$$f_1^{(D)} = \text{LeakyReL U}\left(\text{BN}\left(W_1^{(D)}\left(v_{1:n_c} \oplus u_{1:n_c} \oplus \mathrm{d}_{1:n_d}\right)\right)\right)$$

$$f_i^{(D)} = \text{LeakyReLU}\left(\text{BN}\left(W_i^{(D)}\left(f_{i-1}^{(D)} \oplus \text{diversity}\left(f_{i-1}^{(D)}\right)\right)\right)\right), i = 2:l$$

- $\oplus$: concatenation,
- diversity($\cdot$): mini-batch discrimination vector with each element being the total distance between one sample and all other samples in the mini-batch,
- BN($\cdot$): batch normalization [30],
- LeakyReLU($\cdot$): leaky reflect linear activation function [31].

The function of the generator and discriminator is presented in figure 3.1 from Xu and Veeramachaneni [17].

3.1.4.3 Loss Functions

The model is trained using the Adam optimizer [32]. KL divergence [33] of the cluster vector and discrete variables are jointly optimized and added to the generator loss function to warm up the model efficiently and make it more stable. Generator is optimized as such:

$$\mathcal{L}_G = -\mathbb{E}_{z\sim\mathcal{N}(0,1)} \log D(G(z)) + \sum_{i=1}^{n_c} \mathrm{KL}\left(u_i', u_i\right) + \sum_{i=1}^{n_d} \mathrm{KL}\left(\mathrm{d}_i', \mathrm{d}_i\right)$$

The real data are u_i and d_i and the generated data are u_i' and d_i'. Discriminator is optimized with a cross-entropy loss:

$$\mathcal{L}_D = -\mathbb{E}_{v_{1:n_c},u_{1:n_c},\mathrm{d}_{1:n_d}\sim\mathbb{P}(\mathbf{T})} \log D\left(v_{1:n_c}, u_{1:n_c}, \mathrm{d}_{1:n_d}\right) + \mathbb{E}_{z\sim\mathcal{N}(0,1)} \log D(G(z))$$

3.2 Fair TGAN

This project's model has been developed on top of the Base TGAN model described in the previous section after downloading and deploying its repository and ensuring that it was working properly.

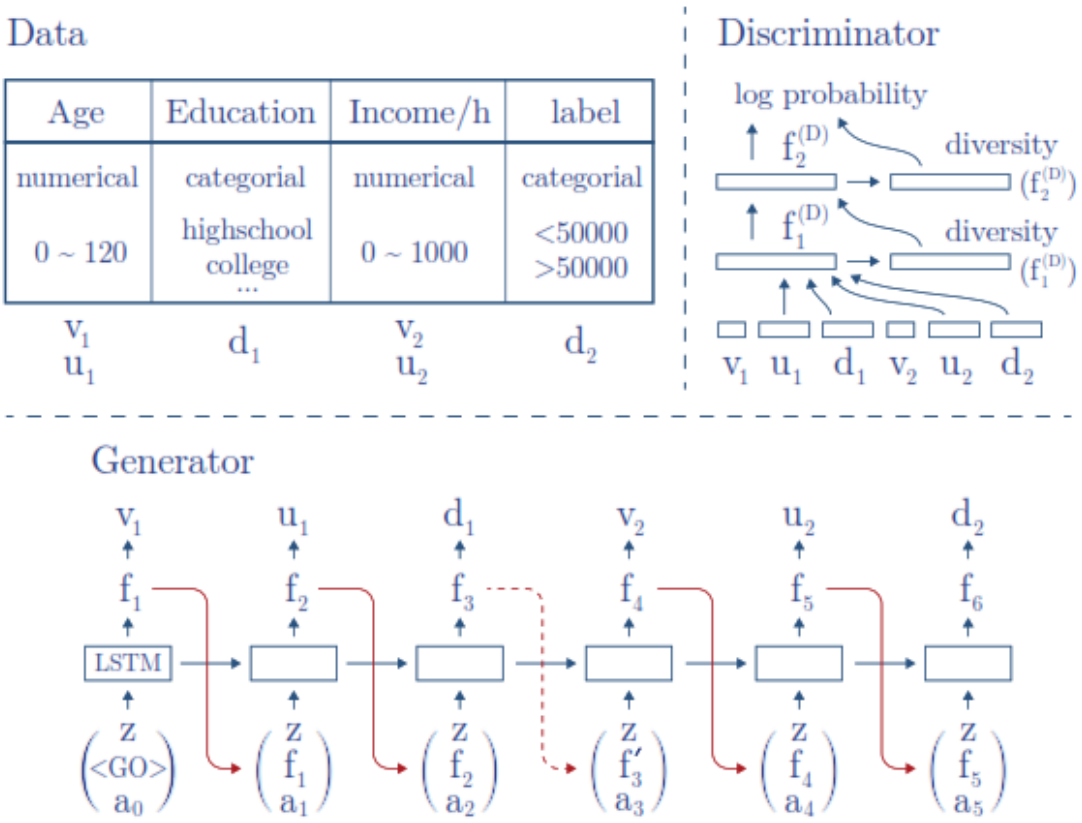


Figure 3.1: Generator and Discriminator architecture of TGAN, copied from Xu and Veeramachaneni [17].

3.2.1 Approach

In order to create the Fair TGAN an additional component working along the discriminator was added with the purpose to drive the model to generate fair data. This was the fairness component. Under this approach the intention was to simultaneously maximize accuracy and fairness with the discriminator and fairness components respectively. While the discriminator tries to distinguish between real and fake data the generator is trained to produce fake data that trick the discriminator and the fairness component drives the generator to generate data that are invariant to the sensitive attribute by predicting the wrong attribute value from the remaining attributes that are generated. The result of this approach was a synthetic dataset with independent attribute values that were maximally invariant to the sensitive attribute and it's value changes which also maintained the classifier training utility of the original dataset.

A similar approach involving a classification and fairness component has been im-

plement by Adel *et al.* [21] in their Fair Adversarial Discriminative model (FAD). In FAD a generator generates values for all independent attributes excluding the sensitive attributes. These attributes are the inputs of a classification and fairness component which work in parallel and they serve the purpose of the discriminator from a standard GAN. The classification and fairness components are respectively involved in the simultaneous optimization for accuracy and fairness. Based on this approach, a third component with the purpose of enforcing classification accuracy was added in some Fair TGAN models for experimentation purposes.

Data in the Fair TGAN model are considered as $Data = \{(x, y, s)\}$ where x are the independent attributes excluding the sensitive attribute, y is the label and s is the sensitive attribute. Respectively, the generated data are represented as $Data' = \{(x', y', s')\}$. The goal of the Fair TGAN model was to ensure that:

1. $\hat{y}$, the prediction of y is not dependent on s without removing s
2. the generated attribute values $Data' = \{(x', y', s')\}$ are fair with regards to disparate impact and disparate mistreatment
3. the classifier training utility of the generated data is maintained

The overall goal of creating such a model was to find out if it was possible to create a fair synthetic dataset out of an unfair one which could also retain the training utility of the original with the possibility of making fairer predictions on the original test set.

3.2.2 Fairness Component

Discarding s was naive as there could still be correlations between x attributes and s that could have influenced the $\hat{y}$ prediction. The purpose of this component in the Fair TGAN model was to predict the value of the sensitive attribute based on the remaining attributes. Essentially, this component was a sensitive attribute classifier $F(x')$ that predicts $\hat{s}$. Fairness was achieved through adversarial construction by defining such a classifier that was trained to predict the wrong sensitive attribute value from the given synthetic independent attribute values in order to ensure that the final generated dataset was not dependent on the sensitive attribute.

3.2.2.1 Architecture

The fairness component created an l-layer fully connected dense neural network using a `for` loop that was iterating over the given number of hidden layers specified during

the setup of the model. The input of the fairness component was a concatenation of a subset of the generator output. Since the output of the generator were tensors representing all the columns of the input dataset, including the label, and the fairness component only needed the independent variables, the sensitive variable and label had to be removed.

3.2.2.2 Loss function

To allow the model to aim for increasing the loss function of the fairness component, it was multiplied by a minus sign. The loss function used was the sigmoid cross entropy for logit inputs which is used in discrete classification tasks to measure probability error. The following equation represents the fairness loss of the Fair TGAN model, adapted from the discriminator loss function from subsection 3.1.4.3:

$$\mathcal{L}_F = -\left[-\mathbb{E}_{x' \sim \mathbb{P}(\mathbf{T})} \log F\left(x'\right) + \mathbb{E}_{z \sim \mathcal{N}(0,1)} \log\left(s\right)\right]$$

3.2.3 Classifier Component

In some trained Fair TGAN models an additional label classifier $C(x')$ was added in order to examine if it would have a positive effect on classification accuracy without affecting the quality of the fairness measures [21]. This classifier predicted the output label $\hat{y}$. The classifier neural network architecture is the same as the fairness network described previously. Its input is also the same which is the generated independent attributes x'. The difference is in the loss function. The classification loss that this component minimizes in the Fair TGAN model is:

$$\mathcal{L}_C = -\mathbb{E}_{x' \sim \mathbb{P}(\mathbf{T})} \log C\left(x'\right) + \mathbb{E}_{z \sim \mathcal{N}(0,1)} \log\left(y\right)$$

A simplified architecture of the Fair TGAN model is outlined in figure 3.2, adapted from Adel *et al.* [21]. Classifier is shown with a dashed arrow to indicate that the component was not part of the final model but only used in certain cases for experimentation purposes.

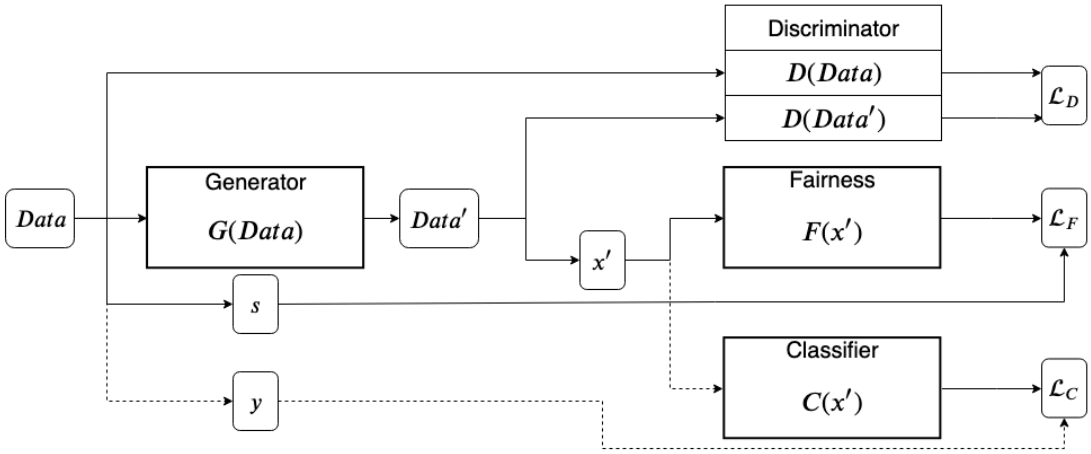


Figure 3.2: Fair TGAN model architecture, adapted from Adel *et al.* [21].

Chapter 4

Data and Model Training Setup

This section presents the training and evaluation procedures of Fair TGAN models followed during the project. The chosen dataset for this project was the Adult dataset and it is described in detail in the next section. The generated datasets were compared with the original dataset as well as datasets generated from the Base TGAN model. Comparisons involved their structure, performances of Machine Learning classifiers trained on them and fairness tests performed on them. These procedures are described in section 4.2.

4.1 Dataset

The **Adult dataset** from the UCI Machine Learning Repository [34] is a multivariate dataset with 48,842 instances and 14 discrete and continuous attributes and a binary class attribute. This dataset has been chosen as it is frequently used in Machine Learning research involving fairness ([21], [24], [35], [36], [37]) due to its size and choices of sensitive variables it provides.

For the experiments of this project the chosen sensitive attribute was `sex`. It is binary with values `Male` and `Female`.

4.1.1 Attributes

The attributes of the dataset and their types are listed below as reported in the UCI Machine Learning Repository website[1]:

1. age: continuous

[1] https://archive.ics.uci.edu/ml/datasets/adult

2. workclass: discrete
3. fnlwgt: continuous
4. education: discrete
5. education-num: continuous
6. marital-status: discrete
7. occupation: discrete
8. relationship: discrete
9. race: White, discrete
10. **sex: discrete - [Female, Male]**
11. capital-gain: continuous
12. capital-loss: continuous
13. hours-per-week: continuous
14. native-country: discrete

The class attribute is binary and it states if the instance's income exceeds \$50K per year or not as `>50K` or `<=50K` respectively.

4.1.2 Preprocessing

In order to prepare the dataset to be imported in the Fair TGAN model, a simple preprocessing operation was performed on it as the dataset file contained each sample in the form of a single string per line. During this process each sample was split in its different attribute values and all continuous values were changed to integers. 3620 samples containing missing values represented as `?` were removed because the Base TGAN model cannot generate valid data if missing values can be found in the input dataset. While removing missing values is not a suggested solution in Machine Learning it was decided that in this case they could be removed since the datasets were going to be considered in comparison to each other and not individually and the number of samples with missing values was low and thus their removal would not hinder training.

The final dataset was extracted to be used as input in the Base and Fair TGAN models and the experiments.

For further Machine Learning experiments the dimensionality of discrete attributes `native-country` and `education` was reduced because they were considered very sparse. Specifically, `native-country` values were reduced to either `US` or `Non-US` and `education` values were reduced to `Bachelors`, `Some-college`, `HS-grad`, `Prof-school`, `Assoc-acdm`, `Assoc-voc`, `Masters`, `Doctorate`, `High-school` and `Primary-Middle-school`. Finally, continuous feature `fnlwgt` was dropped from the datasets as it was highly predictive of the class.

4.2 Model Training and Evaluation

Different Fair TGAN models have been trained for the purposes of this project. These models differed in regards to their architecture and component configuration and in some cases in the learning rate used for optimization. Specific Fair TGAN models were selected for presentation in the thesis based on their resulting datasets. These models and their results are presented in Chapter 5. The training procedure followed was the same for all Fair TGAN models and is described in the following subsection. Then generated datasets from those models were compared to the original dataset in terms of structure (attribute distributions and correlations). Machine Learning algorithms were trained using those datasets as well as the original dataset and their performances were compared. Finally, the fairness tests described in subsection 4.2.3 were executed to measure disparate impact and disparate mistreatment.

4.2.1 Fair TGAN Model Training

The training procedure was identical for all Fair TGAN models despite any differences in structure. The only hyperparameters that varied between different models were the learning rates of the generator/discriminator and the fairness component while the values of all others were kept the same for all models. The reason for changing only the learning rates was because they directly affected the data generation and fairness component loss functions and this could show how strongly different learning rates changed the structure of the generated datasets with focus on the distribution of the sensitive variable and their fairness metrics. Since the focus of the project was to create a model that generated fair datasets, optimization of the remaining hyperparameters

was left for further research. The available hyperparameters of the Fair TGAN model and their values used for the experiments in this project are presented below:

- `max_epoch`: Number of epochs used during training: 5
- `steps_per_epoch`: Number of steps run on each epoch: 10000
- `batch_size`: Size of the batch that fed the model at each step: 200
- `z_dim`: Number of dimensions in the noise input for the generator: 200
- `noise`: Upper bound to the Gaussian noise added to categorical columns: 0.2
- `l2norm`: L2 regularization coefficient when computing losses: 0.00001
- `discrim_learning_rate`: Learning rate for the optimizer of the generator and discriminator: Variable
- `fair_learning_rate`: Learning rate for the optimizer of the fairness component: Variable
- `num_gen_rnn`: Number of units in rnn cell in generator: 500
- `num_gen_feature`: Number of units in fully connected layer in generator: 100
- `num_dis_layers`: Number of layers in discriminator: 4
- `num_dis_hidden`: Number of units per layer in discriminator: 100

The Base TGAN model contained three training methods. All methods were based on the `TowerTrainer` class from the `train` package of TensorPack and they are presented below:

- `GANTrainer`: training iteration of components is explicitly defined.
- `SeparateGANTrainer`: component optimization operations are executed based on a given ratio.
- `MultiGPUTrainer`: `GANTrainer` replacement with multi GPU support.

Since the Fair TGAN model involved changes in the architecture with the addition of fairness and classifier components, it was decided that `GANTrainer` was the most proper trainer in this scenario as it allowed explicit definition of the component

training iteration. Specifically for the Base TGAN model the generator loss function was optimized first and after that optimization was complete the discriminator loss function was optimized. For the Fair TGAN model the fairness loss function was optimized along with the discriminator loss function. They way this order was set was by defining the generator optimization procedure as a control dependency for the other two optimizations. This ensured that the generator loss function would be evaluated before proceeding to the other loss functions. Finally, the `AdamOptimizer` was used for fitting.

4.2.2 Generated Datasets vs Original Dataset

Generated datasets of 40000 samples from selected Fair TGAN models were compared with the original training dataset of 30162 samples and a dataset (also of 40000 samples) generated by the Base TGAN model in terms of structure. This involved explorations of the datasets using the `pandas` library as a first step. It was used to verify dataset sizes and examine statistical information about attributes including mean, standard deviation, minimum and maximum values and distribution of values of the sensitive variable and output label both individually and in combination. In addition, Normalized Mutual Information (NMI) measurements (normalized version of standard MI from `scikit-learn.metrics.normalized_mutual_info_score`) were used to identify how much the sensitive attribute could tell us about the output label in both original and generated datasets and pairwise correlation matrices were compared.

4.2.3 Machine Learning Classification and Fairness tests

Further Machine Learning experiments involved the use of `scikit-learn`, a Python software Machine Learning library. Specifically, the following classifiers were used:

- Multinomial Naive Bayes: `sklearn.naive_bayes.MultinomialNB`
- Logistic Regression: `sklearn.linear_model.LogisticRegression`
- Decision Trees: `sklearn.tree.DecisionTreeClassifier`
- Random Forests: `sklearn.ensemble.RandomForestClassifier`
- Multi-layer Perceptron Classifier: `sklearn.neural_network.MLPClassifier`

Training and test set features were standardized for Logistic Regression. Decision Tree and Random Forest classifiers maximum depth was set to 20 while Random Forests had 500 trees and quality of split (`criterion`) was measured using entropy for information gain. Multi-layer Perceptrons had a ReLU activation function, the Adam solver [38] and hidden layer size of (100,).

Training these classifiers involved generated training datasets from Base and Fair TGAN models. Those TGAN models were trained with the original Adult training dataset while the original test set was set aside. Then the accuracy of those classifiers on the test set was compared to the accuracy of the respective classifiers trained on the original training set in order to show how close the training utility of the generated datasets was to the original's.

Fairness tests involved measuring **disparate impact** and **disparate mistreatment** as defined in section 2.2. To measure these fairness notions a distinction between disadvantaged and advantaged value for the sensitive attribute `sex` was required. Since the original dataset was biased towards women based on the following measures it was decided that the disadvantaged sensitive attribute value would be `Female` and the advantaged value would be `Male`.

For measuring disparate impact the $p\%$-rule [39] was used. This value is calculated as the ratio of the percentage of disadvantaged samples in the positive class (`Income`: `>50K`) over the percentage of advantaged samples also in the positive class. This $p\%$-rule was calculated for the original input training data, the generated training data, the original test set as well as the final predictions of classifiers were applicable. The formula for calculating the $p\%$-rule is:

$$p\%-\text{rule} = \frac{\frac{\text{disadvantaged positives}}{\text{total disadvantaged}}}{\frac{\text{advantaged positives}}{\text{total advantaged}}}$$

To check for disparate mistreatment the following rates were calculated using classifier predictions for disadvantaged and advantaged samples separately:

- *False Positive Rate*: how many samples were incorrectly classified in the positive class from all true negative samples.
- *False Negative Rate*: how many samples were incorrectly classified in the negative class from all true positive samples.

In order for disparate mistreatment to be avoided then the above rates calculated separately for the disadvantaged and advantaged sensitive attribute values should be equal. These rates will be collectively referred to as disparate mistreatment rates.

Chapter 5

Results and Analysis

This chapter presents and explains the findings of the project through selected trained Fair TGAN models and their generated datasets. Initially some information on the original Adult dataset is provided along with a trained Base TGAN model and a comparison of a generated dataset from it with the original. Following that, selected Fair TGAN models are presented in a sequence that indicates the timeline in which they were built and trained. This sequence intends to display how each one lead to the changes made to the next and how those changes affected the performance of their respective generated datasets leading to the final best performing model. Included with each model is their evaluation based on datasets they generated as described in section 4.2.

All models were trained on a desktop computer equipped with an AMD™ Ryzen™ 2700x CPU and an NVIDIA™ GeForce GTX 1080Ti™ GPU. TensorFlow GPU[1] was used to run all the Fair TGAN models.

5.1 Original Adult Dataset

A summary of the whole dataset, its attributes and the preprocessing operations that were executed to prepare them for input in the TGAN models as well as for the subsequent Machine Learning experiments is explained in section 4.1.

The dataset is split into training and test sets found in its UCI Machine Learning Repository.[2] The sample counts in each set along with the amount of samples containing missing values are presented in table 5.1.

[1] https://www.tensorflow.org/install/gpu

[2] https://archive.ics.uci.edu/ml/machine-learning-databases/adult/

	Training	Test	Total
Original	32561	16281	48824
Missing	2399	1221	3620
Remaining	30162	15060	**45222**

Table 5.1: Total sample counts and counts of samples with missing values for the original Adult training and test sets.

The remaining training samples were used for training the TGAN Models while the remaining test samples were used for the Machine Learning classification and fairness tests.

Since the focus of this project was on the specific attribute of `sex`, the purpose of the dataset exploration operations revolved around that. Initially, the counts of `Female` and `Male` values in the training and test sets were found. These are presented in table 5.2:

	Female	Male	Total
Training	9782	20380	30162
Test	4913	10147	15060
Total	14695	30527	**45222**

Table 5.2: `sex` attribute sample counts in the original training and test sets.

Females represent 32% of the training set which was an adequate amount to train TGAN models that generated datasets which had good distributions of the sensitive variable to allow fairness analysis. The distribution is similar for the test set as well at 33% which helped with the classification tasks performed later. Also the Normalized Mutual Information (NMI) score between the attribute `sex` and the output label is 0.0436 for the training set and 0.0425 for the test set. These scores will be relevant when compared with those of generated datasets later.

Accuracy results on the test set from Machine Learning classifiers trained with the training set are presented in table 5.3. These results will serve as the accuracy basis for comparison between the original and generated datasets.

The most important reason for choosing the Adult dataset was that it is inherently biased towards women. Specifically, the $p\%$-rule for both the training and test set is

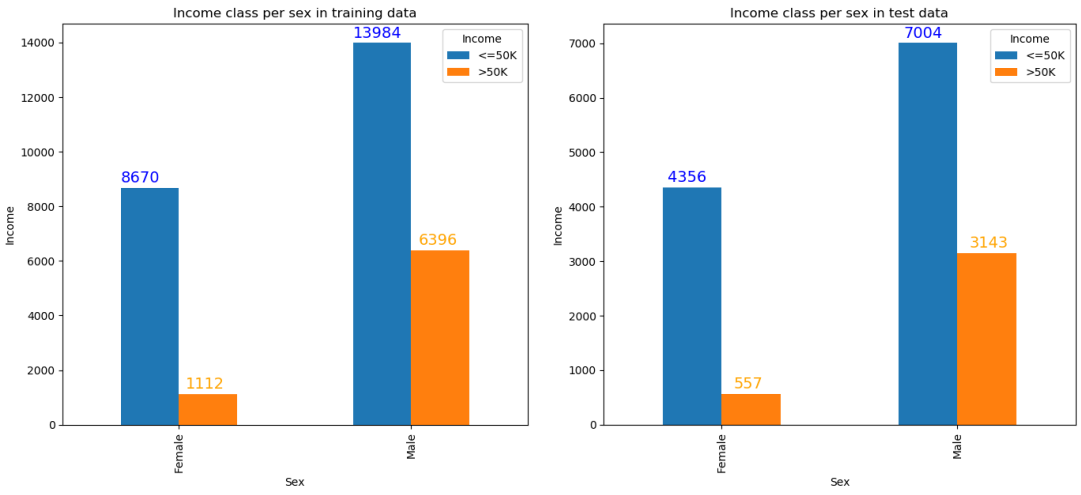
Multinomial NB	0.775
Logistic Regression	0.847
Decision Trees	0.819
Random Forests	0.857
Multi-Layer Perceptron	0.829

Table 5.3: Test set prediction accuracies of different classifiers trained with the original training set.

below 80%. The 80% threshold was set by and U.S. Equal Employment Opportunity Commission as the: *"80%-rule"* [37] or more generally p%-rule whose measure is defined in subsection 4.2.3. The p%-rule for the original training and tests sets is:

- Training: 36.22%
- Test: 36.60%

These values indicate that the percentage of females that have income more than $50K is almost three times smaller than the same percentage for males. As a result, any classification model trained with the Adult dataset will suffer from disparate impact. These values are also visualised in figure 5.1.



(a) Training set cross tabulation.

(b) Test set cross tabulation.

Figure 5.1: Cross tabulation of sensitive attribute `sex` and output label `Income`.

The next step was to look at the fairness metrics for the test set predictions. For the original dataset and all subsequent TGAN models these measures were checked for the

Logistic Regression, Random Forest and Multi-Layer Perceptron classifiers but since all exhibited similar values only the Logistic Regression results are presented.

To check if classifier predictions suffered from disparate mistreatment the two rates explained in subsection 4.2.3 are presented in table 5.4. Both rates are unequal which indicates the presence of disparate mistreatment in the dataset. Unfairness is evident by the increased False Positive Rate (FPR) which indicates higher positive misclassification for males and increased False Negative Rate (FNR) which indicates higher negative misclassification for females.

	Female	Male
FPR	2%	10%
FNR	48%	39%

Table 5.4: Disparate mistreatment rates on Logistic Regression predictions trained with the original training set.

As a final point it is important to note that the $p\%$-rule of the test set is low. As a result any trained classifier will have to make unfair predictions in order to be accurate on that test set, no matter how fair the training dataset is, thus generating a trade-off between accuracy and fairness.

5.2 Base TGAN Model

This section will present the results of the Base TGAN model trained on the Adult dataset. These results will serve for later comparisons with datasets generated by subsequent Fair TGAN models.

5.2.1 Setup

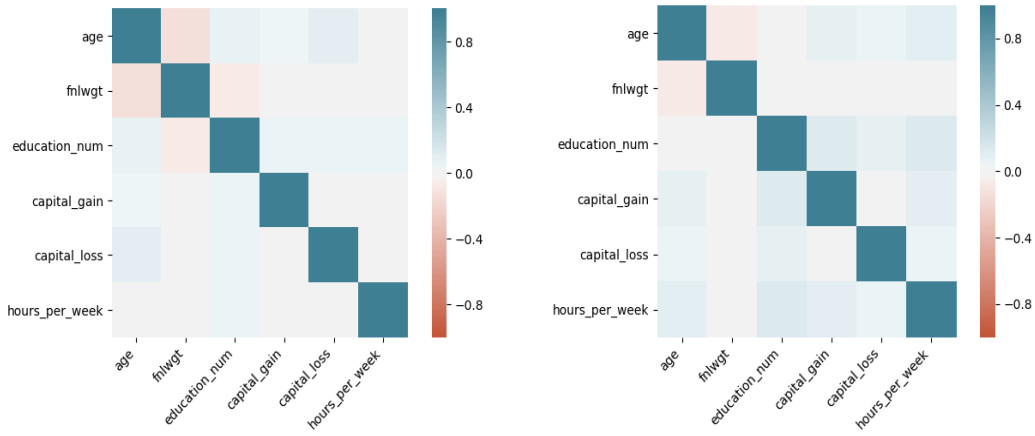
Training hyperparameter values were those set in subsection 4.2.1 for Fair TGAN models. These values along with the single learning rate with value 0.0002 were suggested by the authors of the repository. The Base TGAN model required 43 minutes and 45 seconds to train for 5 epochs and 10000 steps per epoch with an average of 19.046 iterations per second. This showed that the training performance of GANs for tabular data is very efficient in comparison to training for images or video data.

5.2.2 Generated Dataset Structure

A training dataset of 40000 samples was generated from the Base TGAN model. The structure of this dataset in regards to the sensitive attribute and output label is presented in table 5.5. The distributions of these variables indicate that the generated dataset is very close the original in regards to them. Also, two pairwise correlation matrices of the Base TGAN generated and original training dataset are presented in figures 5.2a and 5.2b respectively. These matrices do not show any strong positive or negative correlations for either dataset but they are very close between them. The NMI score between `sex` and `Income` is 0.0537 for the Base TGAN dataset and 0.0436 for the original training set.

	Sex		Income	
	Female	Male	>50K	<=50K
Original	32%	68%	25%	75%
Base TGAN	**35%**	**65%**	**24%**	**76%**

Table 5.5: Sensitive attribute and output label distribution comparisons between Base TGAN and original training datasets.



(a) Correlation matrix of Base TGAN generated dataset.

(b) Correlation matrix of original Adult training dataset.

Figure 5.2: Base TGAN and original training dataset pairwise correlation matrices.

5.2.3 Results

The p%-rule for the Base TGAN training dataset is 31.95% which is lower than the original's 36.22%. This decrease is a result of lower percentage of positively classified females (10.20% vs 11.37%) and higher percentage of positively classified males (31.94% vs 31.38%). As a result, the Base TGAN model generates unfair data.

Table 5.6 presents the performance of classifiers trained with the Base TGAN training set and the original. Apart from the Multinomial NB classifier which performs identically for both datasets, all other classifiers seem to slightly underperform.

	Base TGAN	Original
Multinomial NB	0.775	0.775
Logistic Regression	0.827	0.847
Decision Trees	0.784	0.819
Random Forests	0.825	0.857
Multi-Layer Perceptron	0.817	0.829

Table 5.6: Prediction accuracies of different classifiers trained with the Base TGAN and original training sets.

The Logistic Regression classifier test set prediction p%-rules and disparate mistreatment rates are presented on table 5.7. There seems to be a deterioration in both types of measures. The p%-rule is less than half of the original while the gap between disadvantaged and advantaged sensitive attribute values has grown especially for FNRs. This a direct result of the Logistic Regression classifier trained on the Base TGAN dataset misclassifying more females in the negative class (75% vs 48%) and more males in the positive class (14% vs 10%).

	Base TGAN		Original	
p%-rule	12.66%		30.01%	
Disparate Mistreatment	Female	Male	Female	Male
FPR	1%	14%	2%	10%
FNR	75%	37%	48%	39%

Table 5.7: Fairness metrics on Logistic Regression test set predictions.

5.3 Fair TGAN Models

This section presents a specific selection of trained Fair TGAN models that differ either in their component setup and data flow between them or the fairness component learning rate. Each model's results served as the basis for the changes in the next model until the final Fair TGAN model which produced the best results of all the models trained during the project.

It has to be noted that for the first five models presented below the learning rate of all components was identical and it was represented as a single hyperparameter called `learning_rate`. The divided learning rates as shown in subsection 4.2.1 were only utilized in the final Fair TGAN model.

5.3.1 Fair TGAN Model 1

5.3.1.1 Setup

The first Fair TGAN model built contained the addition of just the fairness component on the Base TGAN model. During training, priority was given to the generator as it was set as the control dependency for the other two components. The fairness and discriminator components were not given any other dependencies between them as the intention was for them to be trained in parallel.

This initial model had a difference in data flow from the general architecture presented in figure 3.2. The difference was that the original s and generated s' sensitive attributes were removed from their respective discriminator inputs, $Data$ and $Data'$ and from the discriminator loss function as well. The intention behind this setup was to see how the generator would be trained in regards to the sensitive attribute when the discriminator wasn't aware about the existence of that attribute. Hyperparameter value `learning_rate` was set to 0.0002.

5.3.1.2 Generated Dataset Structure

The resulting generated dataset contained only female samples. All datasets generated using this model had only female samples and this rendered the model unusable for any further experiments. These results were attributed to the fact that since the discriminator didn't receive the sensitive attribute as input then the generator learned that it didn't matter what value it assigned to it.

5.3.2 Fair TGAN Model 2

5.3.2.1 Setup

The second model was based on the previous one with the addition of the classifier component as shown in figure 3.2. Again, all the components after the generator did not receive the sensitive attribute as input.

The classifier component was added to check if it could fix the problem with the sensitive attribute value generation of the previous model. Since the purpose of this component was to explicitly control the accuracy of the generated data this could have forced the generator to not produce only the value `Female` for the sensitive attribute. Hyperparameter value `learning_rate` was kept at 0.0002.

5.3.2.2 Generated Dataset Structure

With the added classifier component the result hadn't changed since the second Fair TGAN model also produced data with only the value `Female`. The reason that the value `Female` was generated instead of `Male` was attributed possibly to the fact that since the majority of samples were male and the fairness component's intention was to increase the prediction loss of the sensitive attribute it resulted in the generator producing only `Female` values.

5.3.3 Fair TGAN Model 3

The fact that the first two models produced datasets with only female samples was attributed to the sensitive attribute not being passed in the discriminator component so all subsequent models would include this attribute in their discriminator inputs. As a result, usage and data flow of the generator and discriminator in this and all following models were identical to the Base TGAN model.

5.3.3.1 Setup

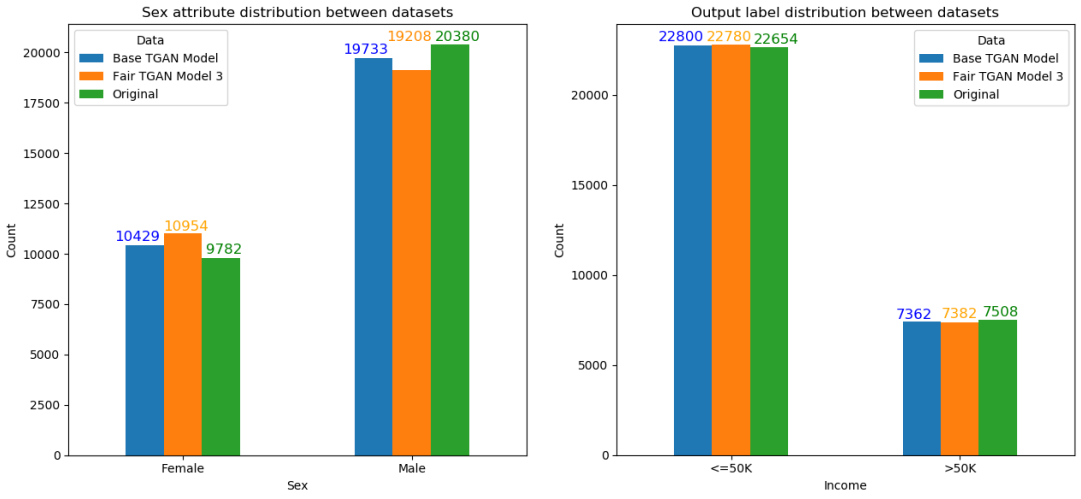
With the addition of the sensitive attribute in the discriminator input the classifier component was kept in the model as the intention was to try and make only singular changes between models to allow for better understanding and attribution of the results. With this setup the third model is represented by figure 3.2 exactly.

Hyperparameter value `learning_rate` was kept at 0.0002. Total training time for this model was 1 hour, 11 minutes and 20 seconds with an average of 11.676 iterations

per second. The increase in training time from the Base TGAN model is due to the additional fairness and classifier components.

5.3.3.2 Generated Dataset Structure

Initially, sensitive attribute and output label distributions between the original and Base TGAN generated datasets were compared. These distributions are presented in figures 5.3a and 5.3b respectively. To allow for visual comparison through those figures, 30162 samples were randomly selected from the Base and Fair TGAN model datasets. While these distributions are close, the pairwise correlation matrix of the continuous attributes deviates from the original while only the value 0 was generated for attribute `capital_loss`. Especially attributes `fnlwgt` and `education_num` seem to be more negatively correlated to `age` and `hours_per_week`. A comparison with the original correlation matrix is shown in figure 5.4. Finally, the NMI score between `sex` and `Income` is 0.108, higher than the original's score of 0.0436 indicating that this set could be even more unfair.



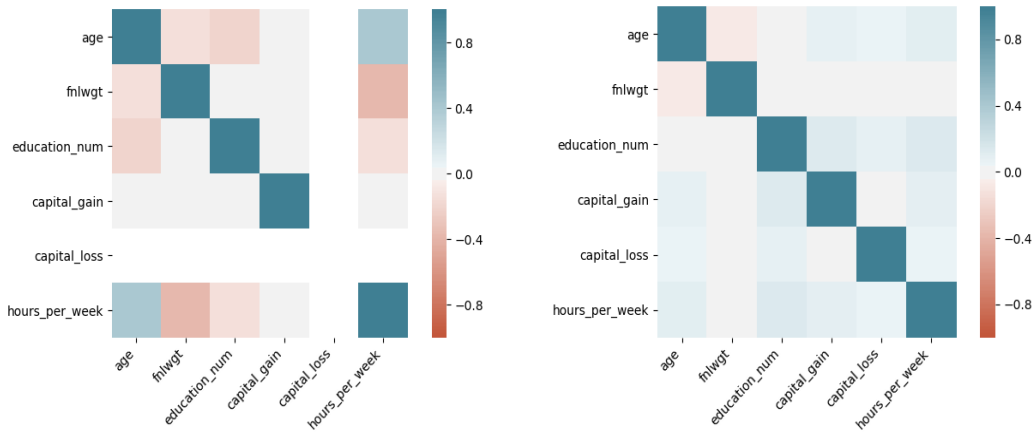
(a) Sensitive attribute distributions.

(b) Output label distributions.

Figure 5.3: Cross tables of sensitive attribute and output label distributions for Fair TGAN Model 3.

5.3.3.3 Results

Unlike the Base TGAN model this model's generated training dataset achieved only a $p\%$-rule of 15.82% which is much lower than the original's 36.22%. This decrease was



(a) Correlation matrix of Fair TGAN Model 3 generated dataset.

(b) Correlation matrix of original Adult training dataset.

Figure 5.4: Fair TGAN Model 3 and original training dataset correlation matrices.

a result of a very low percentage of positively classified females (5.57% vs 11.37%) and higher percentage of positively classified males (35.23% vs 31.38%).

Classifier performances with regard to this training dataset are shown in table 5.8. Apart from the Decision Tree classifier which performs better than the original and the Multinomial NB classifier which performs the same, all other classifiers seem to underperform.

	Base TGAN	Fair TGAN 3	Original
Multinomial NB	0.775	0.775	0.775
Logistic Regression	0.827	0.73	0.847
Decision Trees	0.784	0.88	0.818
Random Forests	0.803	0.771	0.84
Multi-Layer Perceptron	0.814	0.773	0.839

Table 5.8: Prediction accuracies of different classifiers trained with the Fair TGAN Model 3, Base TGAN and original training sets.

Despite the fact that Fair TGAN Model 3's training dataset was even more unfair than the Base TGAN set, it trained a Logistic Regression classifier that could make fairer predictions very close to the original even if it's accuracy was lower. These results are shown in table 5.9. This was due to both higher overall True Positive Rate (TPR) of 0.91 and overall FPR of 0.83 which indicate that Fair TGAN Model 3's

training set trained the classifier to almost always classify samples in the positive class.

	Base TGAN		Fair TGAN 3		Original	
p%-rule	12.66%		29.32%		30.01%	
Disparate Mistreatment	Female	Male	Female	Male	Female	Male
FPR	0.01%	0.14%	0.04%	0.12%	0.02%	0.10%
FNR	0.75%	0.37%	0.94%	0.81%	0.48%	0.39%

Table 5.9: Fairness metrics on Logistic Regression test set predictions for Fair TGAN Model 3.

5.3.4 Fair TGAN Model 4

This model was the first to produce a fair dataset with no disparate impact.

5.3.4.1 Setup

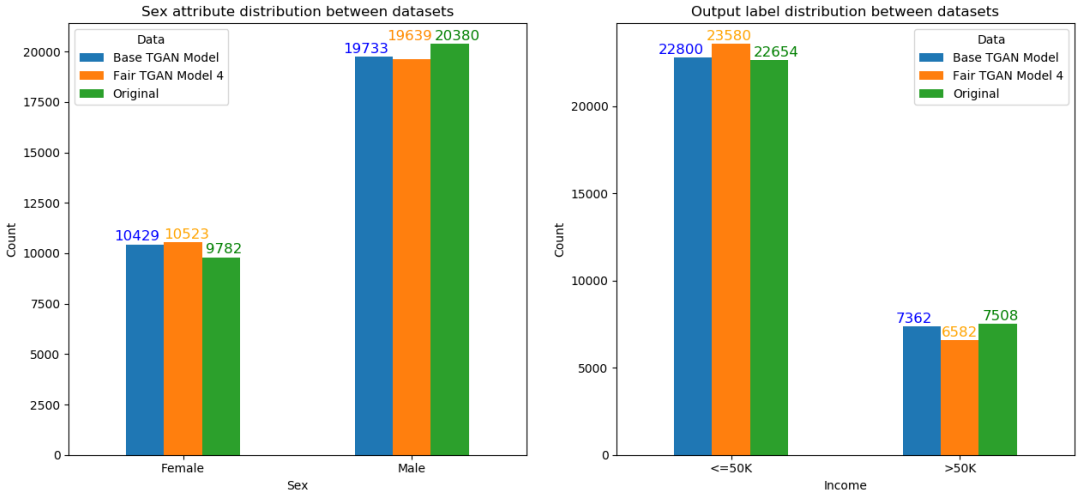
After the previous models failed to generate fair datasets it was decided to drop the classifier component in order to observe only the effect of the fairness component on the model. This was the only change made on the previous model while data flow and hyperparameter values were kept the same.

Hyperparameter value `learning_rate` was kept at 0.0002. Total training time for this model was 1 hour, 11 minutes and 20 seconds with an average of 11.742 iterations per second.

5.3.4.2 Generated Dataset Structure

Sensitive attribute and output label distributions are similar to the original and Base TGAN generated datasets with this Fair TGAN model generating more `Female` and negative class samples and less `Male` and positive class samples. These distributions are presented in figures 5.5a and 5.5b respectively. The pairwise correlation matrix of the continuous attributes seems to follow the same pattern as the original and Base TGAN but it indicates more positively correlated attributes except `education_num` and `fnlwgt`. This cannot be definitively attributed to the fairness component. The correlation matrices for this model's dataset and the original's are shown in figure 5.6. The interesting measure for this dataset is the NMI score between `sex` and `Income`

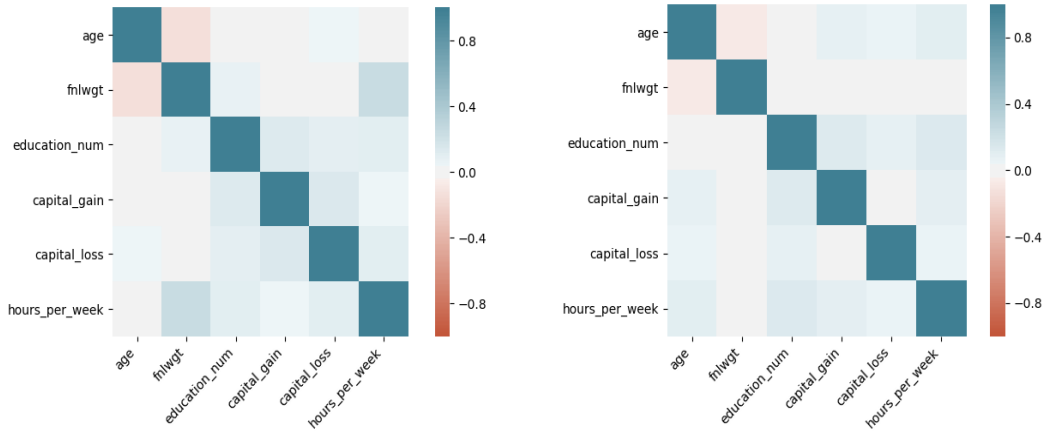
which is 0.0026, much lower that the original's score of 0.0436. This is a strong indication of achieved invariance of the sensitive attribute.



(a) Sensitive attribute distributions.

(b) Output label distributions.

Figure 5.5: Cross tables of sensitive attribute and output label distributions for Fair TGAN Model 4.



(a) Correlation matrix of Fair TGAN Model 4 generated dataset.

(b) Correlation matrix of original Adult training dataset.

Figure 5.6: Fair TGAN Model 4 and original training dataset correlation matrices.

5.3.4.3 Results

This was the first model that produced data that were fair in regards to disparate impact and the $p\%$-rule. Specifically this model achieved a $p\%$-rule of 79.74% which almost

clears the 80% threshold explained in section 5.1. This percentage was achieved due to both an increased percentage of positively classified females (18.75% vs 11.37%) and lower percentage of positively classified males (23.51% vs 31.38%).

While the generated training set was fair the classifiers it trained underperformed in comparison to the original, except the Decision Trees. Classifier accuracies are shown in table 5.10. Since the dataset's sensitive attribute and output label distributions as well as correlation matrix are very similar to the original, the increased amount of female samples and decreased amount of male samples in the positive class were the most probable cause of this underperformance. This could be an indication of a trade-off between accuracy and fairness.

	Base TGAN	Fair TGAN 4	Original
Multinomial NB	0.775	0.775	0.775
Logistic Regression	0.827	0.736	0.847
Decision Trees	0.784	0.814	0.818
Random Forests	0.803	0.744	0.84
Multi-Layer Perceptron	0.814	0.728	0.839

Table 5.10: Prediction accuracies of different classifiers trained with the Fair TGAN Model 4, Base TGAN and original training sets.

Unlike the previous model, the Logistic Regression classifier trained with the Fair TGAN Model 4's fair training dataset produced very unfair predictions. Fairness metrics on those predictions are shown in table 5.11. Overall TPR and FPR were 88% and 70% respectively and compared with the previous model, positive classification deteriorated while negative classification improved. This could possibly be a result of increased negative class samples and decreased positive class samples despite the fact that those changes were not large compared to the original training dataset.

5.3.5 Fair TGAN Model 5

At this point, every model trained had changes only in regards to component structure or data flow between these components. Since no other model configuration trained was able to improve on Fair TGAN Model 4's fair dataset generation it was decided that the `learning_rate` hyperparameter of that model would be changed to see how it would affect data generation.

	Base TGAN		Fair TGAN 4		Original	
p%-rule	12.66%		3.33%		30.01%	
Disparate Mistreatment	Female	Male	Female	Male	Female	Male
FPR	1%	14%	1%	19%	2%	10%
FNR	75%	37%	99%	65%	48%	39%

Table 5.11: Fairness metrics on Logistic Regression test set predictions for Fair TGAN Model 4.

The learning rate hyperparameter is a value that controls by how much the model weights are changed in regards with losses and the value of 0.0002 was deemed to low as a higher amount of steps was necessary for a change to be made. Specifically, significant change seemed to start only after Epoch 4 at 30000 steps for the generator and discriminator losses for all previous models and as a result it was decided to increase the learning rate parameter in order to examine how the Adam optimizer's control over how much the model changes would affect component losses and the end result of data generation.

The first model trained with a higher learning rate value failed to produce usable data. For this model the learning rate was set to 0.2 which seemed to be too high in the end. Specifically, this model was always generating the same sample which was a female classified in the negative class.

5.3.6 Fair TGAN Model 6

The result of the previous model led to the decision of splitting the learning rate between generator and discriminator, and fairness for this final model of the project. With this change this final model was able to generate fair data that were able to retain most of the training utility of the original dataset and predict slightly fairer classification on the original test set.

5.3.6.1 Setup

The results of the previous model deemed the increased learning rate too high for the generator and discriminator components as the model produced a singular sample. For this model it was decided to divide the learning rate hyperparameter to two new variables, the `discrim_learning_rate` for the generator and discriminator and the

`fair_learning_rate` for the fairness component. The former hyperparameter's value would be reverted to its initial value since that seemed to work well for data generation. Instead, the fairness learning rate would be changed since it would only affect the new component added to the model and with this change it could be possible to accurately examine how it could control fairness in the generated data while all other parameters were kept constant.

Hyperparameter value `discrim_learning_rate` was reverted to 0.0002 and `fair_learning_rate` was set to 0.2. Total training time for this model was 1 hour, 3 minutes and 24 seconds with an average of 13.138 iterations per second.

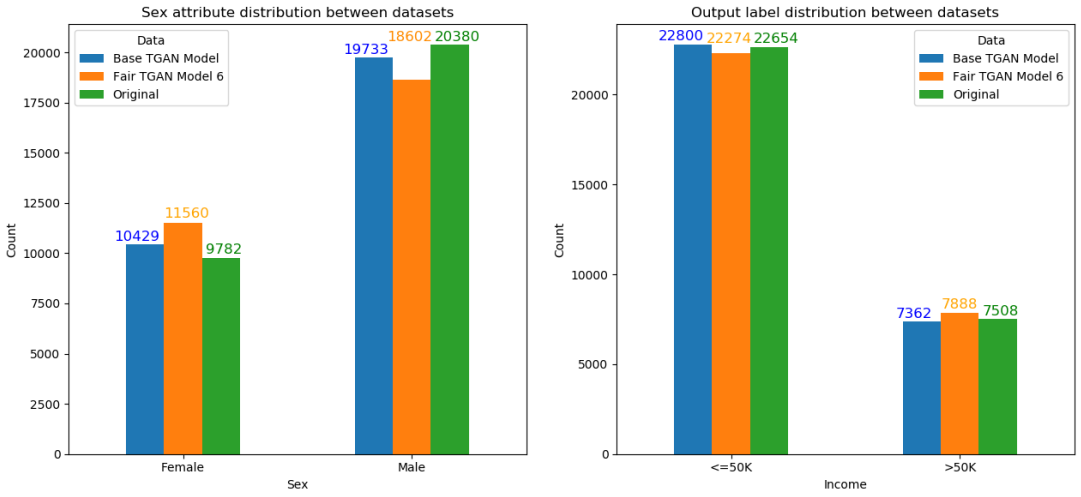
5.3.6.2 Generated Dataset Structure

The generated dataset from this model has the biggest differences in sensitive attribute and output label distributions from the original dataset than any other model. Still, these distributions do not deviate in an alarming way from the original dataset. As shown in figure 5.7a this model generated more `Female` samples and less `Male` samples in comparison to the original just as Fair TGAN Model 4 did which was the first to produce fair data. Unlike model 4 though this model generated more positive samples and less negative samples than the original as seen in figure 5.7b. These distributions are a strong indication that this model is able to produce fair data. The correlation matrices from figure 5.8 indicate that correlations between continuous variable differ from the original. Specifically, `hours_per_week` seems to be more positively correlated with `age` and more negatively correlated with `capital_gain` and `fnlwgt` while `fnlwgt` and `age` seem to be slightly positively correlated instead of negatively as the original. Finally, the NMI score between `sex` and `Income` was 0.0014 which was the lowest achieved by any model, showing strong invariance of the sensitive attribute.

5.3.6.3 Results

Fair TGAN Model 6 produced the best dataset from all models regarding all examined aspects in the project's experiments. The dataset that this model generated is fair in regards to the *80%-rule*, its training utility on the classifiers used was the closest to original's from all other fair datasets that were generated and the predictions of those classifiers were in fact fairer than the original's.

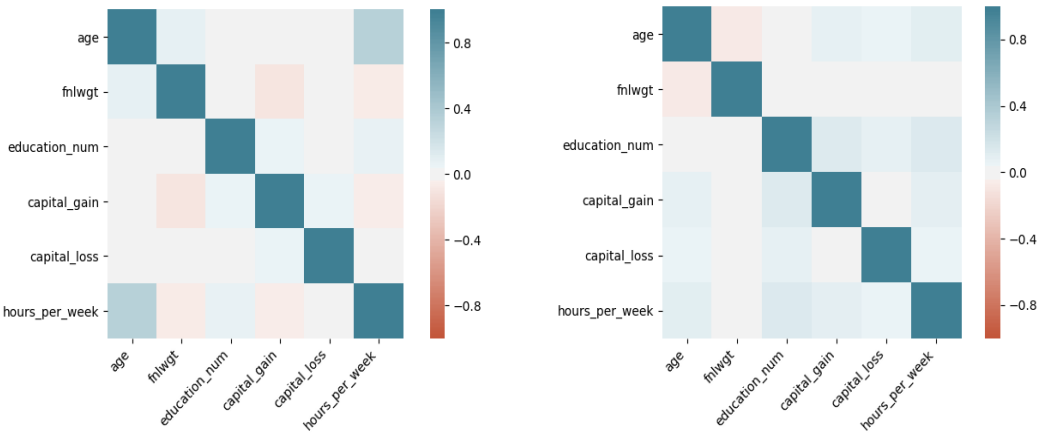
This model's dataset achieved the highest p%-rule from all datasets generated by previous models with a value of 86.42% and it cleared the 80% threshold indicating



(a) Sensitive attribute distributions.

(b) Output label distributions.

Figure 5.7: Cross tables of sensitive attribute and output label distributions for Fair TGAN Model 6.



(a) Correlation matrix of Fair TGAN Model 6 generated dataset.

(b) Correlation matrix of original Adult training dataset.

Figure 5.8: Fair TGAN Model 6 and original training dataset correlation matrices.

that disparate impact was removed. Specifically, 3650 female samples and 6798 male samples were in the positive class with their respective p%-rule percentages being 23.81% (original: 11.37%) and 27.55% (original: 31.38%). These percentages led to the final 86.42%.

The classifiers trained with this dataset performed better than those from Fair TGAN Model 4, the other model that produced fair data. These performances were close to the original and they were the best achieved by any fair dataset produced by

any other model so far. They are shown in table 5.12. These performances indicate that the increased invariance of the sensitive variable did not hinder classifier performance significantly. In fact, despite that the generated fair training set had a percentage of female samples in the positive class of 9.125% and the original test set only 3.7%, the classifiers trained with it were still accurate on that unfair test set.

The Logistic Regression classifier trained with the Fair TGAN Model 6's fair training dataset made fairer predictions than the original as the p%-rule of those predictions is 32.6% while for the original is 30.01%. This percentage is a great improvement over the Base TGAN as well. Disparate mistreatment is not removed from the predictions but the gaps between female and male FPR and FNR percentages show improvement from the Base TGAN. These values are presented in table 5.13. Overall TPR and FPR for Logistic Regression was 0.9 and 0.49 respectively which is a significant improvement over the one from Fair TGAN Model 4 which tried to classify most samples in the positive class.

	Base TGAN	Fair TGAN 6	Original
Multinomial NB	0.775	0.775	0.775
Logistic Regression	0.827	0.808	0.847
Decision Trees	0.784	0.852	0.818
Random Forests	0.803	0.795	0.84
Multi-Layer Perceptron	0.814	0.764	0.839

Table 5.12: Prediction accuracies of different classifiers trained with the Fair TGAN Model 6, Base TGAN and original training sets.

	Base TGAN		Fair TGAN 6		Original	
p%-rule	12.66%		32.6%		30.01%	
Disparate Mistreatment	Female	Male	Female	Male	Female	Male
FPR	1%	14%	4%	13%	2%	10%
FNR	75%	37%	61%	47%	48%	39%

Table 5.13: Fairness metrics on Logistic Regression test set predictions for Fair TGAN Model 6.

Chapter 6

Conclusions

The overall goal of this project was to create a Generative Adversarial Network model that could generate fair synthetic tabular datasets which could retain the training utility of the original tabular dataset given as input. In terms of disparate impact this goal has been achieved on the chosen Adult income dataset. The final model trained with an unfair dataset was able to generate a fair dataset with a $p\%$-rule over 80% that was used to train classifiers that could perform with similar accuracy to those trained with the original training set while also improving on the $p\%$-rule of the predictions on the original test set.

The process followed during the project was to setup up a Fair TGAN model based on component selection, component architecture, data flow or learning rate, train such a model with the original training dataset, generate a synthetic training set, train Machine Learning classifiers and test them on the original test set, calculate fairness metrics and compare them with those of the original. Each subsequent model configuration was based on the results of the previous one. After training several models the results of the final Fair TGAN model as presented in subsection 5.3.6 were considered sufficient to finalize the experimentation.

While a fair training set without disparate impact was generated and the test set predictions were fairer from the original it should be noted that they were still well below the 80% threshold and still suffered from disparate mistreatment. Ideally, true fairness would be achieved if those predictions were fair but since the original test set was unfair this wouldn't allow a classifier trained with the fair training set to be both accurate and fair on that test set since they are mutually exclusive. This issue paves the road for future work on controlling accuracy and fairness.

The main focus of the project was to build a model that generates fair tabular

datasets and with such a model the ground is laid for several follow up research directions to achieve fairness in our predictions. As mentioned previously, an important issue is to control accuracy and fairness especially when our test sets are unfair. Regarding the Fair TGAN model this could be applied by using periodic training of each component to prioritize either accuracy or fairness. Another approach for achieving fairness can be the use of constrained optimization in the fairness component through the TensorFlow Constrained Optimization Library[1] which supports proxy constraints which will be differentiable approximations of the non-convex $p\%$-rule [36]. This approach though raises the issue that the creation of synthetic fair datasets could be deemed unnecessary if constrained optimization is required since this optimization can applied straight to the classification task. Finally, further research on generation of fair tabular data should involve multiclass, continuous or multiple sensitive attributes.

In conclusion, fairness in Machine Learning is a very complicated topic due to all the different approaches that introduce it in supervised classification and others that measure it but in the end important decisions should not be left exclusively on machines, no matter how fair they are deemed to be.

[1] https://github.com/google-research/tensorflow_constrained_optimization

Bibliography

[1] Ziyuan Zhong. A tutorial on fairness in machine learning, https://towardsdatascience.com/a-tutorial-on-fairness-in-machine-learning-3ff8ba1040cb. 2018.

[2] Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez Rodriguez, and Krishna P Gummadi. Fairness beyond disparate treatment & disparate impact: Learning classification without disparate mistreatment. In *Proceedings of the 26th International Conference on World Wide Web*, pages 1171–1180. International World Wide Web Conferences Steering Committee, 2017.

[3] Kaggle. The state of data science machine learning 2017, https://www.kaggle.com/surveys/2017. 2017.

[4] The Economist. The world's most valuable resource is no longer oil, but data, https://www.economist.com/leaders/2017/05/06/the-worlds-most-valuable-resource-is-no-longer-oil-but-data. 2017.

[5] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[6] Ian Goodfellow. Nips 2016 tutorial: Generative adversarial networks. *arXiv preprint arXiv:1701.00160*, 2016.

[7] L. J. Ratliff, S. A. Burden, and S. S. Sastry. Characterization and computation of local nash equilibria in continuous games. In *2013 51st Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 917–924, Oct 2013.

[8] Kamyar Nazeri, Eric Ng, and Mehran Ebrahimi. Image colorization using generative adversarial networks. In *International Conference on Articulated Motion and Deformable Objects*, pages 85–94. Springer, 2018.

[9] Simone Meyer, Victor Cornillère, Abdelaziz Djelouah, Christopher Schroers, and Markus H. Gross. Deep video color propagation. *CoRR*, abs/1808.03232, 2018.

[10] Bill Howe, Julia Stoyanovich, Haoyue Ping, Bernease Herman, and Matt Gee. Synthetic data for social good. *CoRR*, abs/1710.08874, 2017.

[11] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*, pages 265–284. Springer, 2006.

[12] Jun Zhang, Graham Cormode, Cecilia M Procopiuc, Divesh Srivastava, and Xiaokui Xiao. Privbayes: Private data release via bayesian networks. *ACM Transactions on Database Systems (TODS)*, 42(4):25, 2017.

[13] Qi Dong, Michael R Elliott, and Trivellore E Raghunathan. A nonparametric method to generate synthetic populations to adjust for complex sampling design features. *Survey methodology*, 40(1):29, 2014.

[14] Cristóbal Esteban, Stephanie L Hyland, and Gunnar Rätsch. Real-valued (medical) time series generation with recurrent conditional gans. *arXiv preprint arXiv:1706.02633*, 2017.

[15] Shreyas Patel, Ashutosh Kakadiya, Maitrey Mehta, Raj Derasari, Rahul Patel, and Ratnik Gandhi. Correlated discrete data generation using adversarial training. *arXiv preprint arXiv:1804.00925*, 2018.

[16] Noseong Park, Mahmoud Mohammadi, Kshitij Gorde, Sushil Jajodia, Hongkyu Park, and Youngmin Kim. Data synthesis based on generative adversarial networks. *CoRR*, abs/1806.03384, 2018.

[17] Lei Xu and Kalyan Veeramachaneni. Synthesizing tabular data using generative adversarial networks. *CoRR*, abs/1811.11264, 2018.

[18] Richard Primus. The future of disparate impact. *Michigan Law Review*, pages 1341–1387, 2010.

[19] Michael Feldman, Sorelle A Friedler, John Moeller, Carlos Scheidegger, and Suresh Venkatasubramanian. Certifying and removing disparate impact. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 259–268. ACM, 2015.

[20] Solon Barocas and Andrew D Selbst. Big data's disparate impact. *Calif. L. Rev.*, 104:671, 2016.

[21] Tameem Adel, Isabel Valera, Zoubin Ghahramani, and Adrian Weller. One-network adversarial fairness. 2019.

[22] Moritz Hardt, Eric Price, and Nathan Srebro. Equality of opportunity in supervised learning. *CoRR*, abs/1610.02413, 2016.

[23] Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. Fairness through awareness. In *Proceedings of the 3rd innovations in theoretical computer science conference*, pages 214–226. ACM, 2012.

[24] Prasanna Sattigeri, Samuel C Hoffman, Vijil Chenthamarakshan, and Kush R Varshney. Fairness gan. *arXiv preprint arXiv:1805.09910*, 2018.

[25] Michael Feldman, Sorelle A Friedler, John Moeller, Carlos Scheidegger, and Suresh Venkatasubramanian. Certifying and removing disparate impact. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 259–268. ACM, 2015.

[26] Gabriel Goh, Andrew Cotter, Maya Gupta, and Michael P Friedlander. Satisfying real-world goals with dataset constraints. In *Advances in Neural Information Processing Systems*, pages 2415–2423, 2016.

[27] Toshihiro Kamishima, Shotaro Akaho, Hideki Asoh, and Jun Sakuma. Fairness-aware classifier with prejudice remover regularizer. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 35–50. Springer, 2012.

[28] N. Patki, R. Wedge, and K. Veeramachaneni. The synthetic data vault. In *2016 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pages 399–410, Oct 2016.

[29] Haoyue Ping, Julia Stoyanovich, and Bill Howe. Datasynthesizer: Privacy-preserving synthetic datasets. In *Proceedings of the 29th International Conference on Scientific and Statistical Database Management*, SSDBM '17, pages 42:1–42:5, New York, NY, USA, 2017. ACM.

[30] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[31] Andrew L Maas, Awni Y Hannun, and Andrew Y Ng. Rectifier nonlinearities improve neural network acoustic models. In *Proc. icml*, volume 30, page 3, 2013.

[32] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[33] S. Kullback and R. A. Leibler. On information and sufficiency. *Ann. Math. Statist.*, 22(1):79–86, 03 1951.

[34] Dheeru Dua and Casey Graff. UCI machine learning repository, 2017.

[35] Alekh Agarwal, Alina Beygelzimer, Miroslav Dudík, John Langford, and Hanna M. Wallach. A reductions approach to fair classification. *CoRR*, abs/1803.02453, 2018.

[36] Andrew Cotter, Heinrich Jiang, and Karthik Sridharan. Two-player games for efficient non-convex constrained optimization. *CoRR*, abs/1804.06500, 2018.

[37] Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez Rodriguez, and Krishna P Gummadi. Fairness constraints: Mechanisms for fair classification. *arXiv preprint arXiv:1507.05259*, 2015.

[38] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[39] Dan Biddle. *Adverse impact and test validation: A practitioner's guide to valid and defensible employment testing*. Gower Publishing, Ltd., 2006.