Bias in Multiclass Classification Cheat Sheet

Part 1 - Introduction

Recall that in **Binary Classification**, we allow our output to be in one of two classes usually denoted by 0 or 1. In some cases we need more output classes, e.g., Bicycle, Car, Plane, Motorcycle. In that case, we are in the setting of **Multiclass Classification**. We will usually name the output classes by $1, 2, \ldots, N$. But, importantly, we do not assume any ordering on them!

We will also allow for a multiclass **protected attribute** $\mathcal{P}$, e.g., ethnicity with groups White, Black, Hispanic, Asian. We will usually name these groups with $1, 2, \ldots, M$.

The **challenge** of fairness in Multiclass Classification is to ensure the **performance** of our classifier while also ensuring similar behaviour/performance for all **protected groups**.

Part 2 - Notions of Fairness

Taxonomy

- Equality of Outcome
- Equality of Opportunity

Equal Outcome: we want the algorithm to behave similarly for different groups

Most important notion for equal outcome fairness is the **Frequency Matrix**. It is a $M \times N$ matrix computed as

$$FM_{gi} = P(y_{pred} = i | \mathcal{P} = g)$$

If all M rows of the above matrix are precisely equal, we say that we have **Multiclass Statistical Parity**!

Equal Opportunity: We want the algorithm to perform similarly for different groups

Most important notion for equal opportunity fairness is the **Conditional Confusion Matrices** (we call this the **Confusion Tensor**). It is a family of M Confusion Matrices (for each group g) computed as

$$CM_{ij}^{g} = P(y_{pred} = i | y_{true} = j, \mathcal{P} = g)$$

If all M conditional confusion matrices are precisely equal, we say that we have **Term by Term Equality of Odds**!

Other fairness notions are introduces in the slides.

Part 3 - Measuring Bias

Equal Outcome: all equal outcome metrics start from the **Frequency Matrix**

Multiclass Statistical Parity is a generalisation of the 1d notion of Statistical Parity.

We start by comparing the rows of the Frequency Matrix in pairs with the total variation distance (https://en.wikipedia.org/wiki/Total_variation_distance_of_probability_measures).

$$d(g, h) = \frac{1}{2} \sum_{i} |FM_{gi} - FM_{hi}|$$

We then average out all the computed distances, or take a maximum to get an idea of the worst case bias!

Equal Opportunity: All equal opportunity metrics start from the **Confusion Tensor**

Multiclass Equality of Odds is a generalisation of the 1d notion of Equality of Odds.

We start by comparing the conditional confusion matrices in pairs with an extension of the total variation distance.

$$d(g, h) = \frac{1}{2N} \sum_{ij} |CM_{ij}^{g} - CM_{ij}^{h}|$$

We then average out all the computed distances, or take a maximum to get an idea of the worst case bias!

The **holisticai library** can help in measuring bias. To install the library in a notebook, run the following line of code.

In [ ]:
```
%pip install holisticai
```

Once the library is installed, the documentation can be found here (https://holisticai.readthedocs.io/en/latest/). To import all the Frequency matrix and Confusion Tensor, use the line below

In [ ]:
```
from holisticai.bias.metrics import frequency_matrix, confusion_ten
```

Part 4 - Mitigating Bias

Taxonomy

- Pre-processing
- In-processing
- Post-processing

Pre-processing refers to techniques where mitigation is applied to the data.

In-processing refers to techniques where the mitigation is included in the model and it's training.

Post-processing refers to techniques where the mitigation happens after training, directly on the outcomes.

Part 4.1 - Pre-processing

One possible Pre-processing method, that would work for any AI task, is the **Correlation Remover**. This method works by removing (or reducing) correlations between the sensitive attributes and all attributes used in training. Note: removing correlations does not mean removing all dependence! For instance sensitive attributes could still be recovered using pairs of training attributes.

In equations, this can be formulated as follows. Suppose $\mathbf{s}$ is a sensitive attribute, and $\mathbf{x_1}, \ldots, \mathbf{x_n}$ are the training feature vectors. We find the new vectors $\mathbf{z_1}, \ldots, \mathbf{z_n}$ by solving the following optimization problem

$$\min_{\mathbf{z_1},\ldots,\mathbf{z_n}} ||\mathbf{z_i} - \mathbf{x_i}||^2$$

subject to

$$\sum_{i=1}^{n} \mathbf{z_i}(\mathbf{s_i} - \bar{s})^T = 0$$

Part 4.2 - In-processing

Reference: Julien Rouzot, Julien Ferry, Marie-José Huguet. Learning Optimal Fair Scoring Systems for Multi- Class Classification. ICTAI 2022 - The 34th IEEE International Conference on Tools with Artificial Intelligence, Oct 2022, Virtual, United States.

The method is based upon Mixed Integer Linear Programming methods. A One VS All model is trained for each class, and the result is then obtained choosing the class associated with the model with highest probability. The method can optimize for accuracy, or balanced accuracy. The method allows to add one, or a number of different fairness constraints (e.g., statistical parity).

Part 4.3 - Post-processing

Reference: Alabdulmohsin, Ibrahim M., and Mario Lucic. "A near-optimal algorithm for debiasing trained machine learning models." Advances in Neural Information Processing Systems 34 (2021)

This method works by altering the outputs of a ML system, according to a probabilistic model, which they prove to be near-optimal for bounding the excess Bayes-risk.

Part 4.4 - Holisticai library

The **holisticai library** can help in mitigating bias. To install the library in a notebook, run the following line of code.

In [ ]:
```
%pip install holisticai
```

Once the library is installed, the documentation can be found here (https://holisticai.readthedocs.io/en/latest/). To import all bias mitigation strategies, one can run the following line.

In [ ]:
```
from holisticai.bias.mitigation import *
```