Independent Project: Explainable AI Dashboard

Hexiao Zhang
20932780
hzhangea@connect.ust.hk
Supervised by Prof. Raymond Wong

1 Project Overview

In this project, I worked with staff from Glassbox AI Ltd on the explainable AI dashboard. The dashboard is a web application that aims to provide interpretation services for machine learning problems. I participated in the development of the explainer dashboard and fairness dashboard. Specifically, my contributions include the following two main areas:

- Implemented four kinds of charts visualizing the explanation for machine learning models with structure data.
- Implemented a module to calculate metrics about the fairness of models.

The minutes of four meetings with the supervisor are attached to the report.

2 Introduction

Our PaaS (Platform-as-a-Service) web application is designed to empower users to submit their trained machine learning models and datasets and receive insights on their model performance, outputs, and potential biases in an Explainable AI (XAI) framework. With the goal of enabling users to better understand their models and make more informed decisions based on their outputs, our platform provides an intuitive and user-friendly interface for visualizing and analyzing machine learning models.

Consider a scenario where a financial institution has developed a machine learning model for assessing credit risk. The institution submits the model and dataset to our PaaS platform, and our XAI framework provides a detailed analysis of the model's decision-making process, highlighting the most important factors that influenced the model's output. Armed with this information, the institution can more confidently use the model to assess credit risk, and better understand any potential biases that may exist in the data. Our XAI framework empowers users to develop machine learning models that are not only accurate but also transparent and trustworthy, improving the reliability and accountability of AI systems in various domains.

3 Software Architecture

The dashboard consists of several parts. Among them, the front end uses the React framework, while the back-end server employs the Express framework. The Express server is connected to a MySQL database, which provides persistent storage for users and historical records. The Python algorithm module periodically sends requests to the Express server for tasks using AJAX polling and returns the results.

The back-end services are provided by two modules, Express and Python. Typically, the services of the Python module should also be provided by server-side frameworks like Django or Flask. However, our team did not have access to a GPU server that could be exposed to

the public network during the rapid development phase. Using the current AJAX polling method to provide algorithm services is an effective choice.

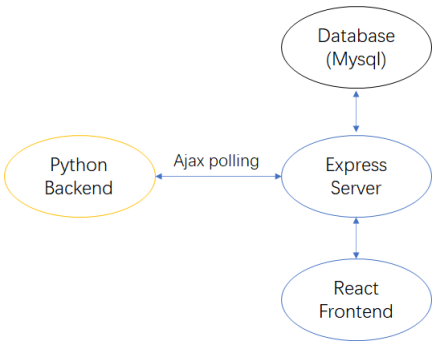

Figure 1: The Design of Software

4 Explainer Dashboard: Interactive Charts

4.1 Preliminary: SHAP

Our dashboard uses **shap**, an open-source Python library, to implement the functionality of model explanation. Inspired by cooperative game theory, **SHAP (SHapley Additive exPlanations)** can explain the output of any machine learning model. SHAP constructs an additive explanation model, where all features are considered as "contributors". For each predicted sample, the model generates a predicted value, and a SHAP value is assigned to each feature of the sample.

Suppose the i-th sample is x_i, the j-th feature of the x_i sample is x_{ij}, the model's predicted value for this sample is y_i, and the baseline of the entire model (usually the mean of the predicted variable for all samples) is y_{base}, then the SHAP value follows the equation below:

$$y_i = y_{base} + \sum_j f(x_{ij}) \tag{1}$$

Where $f(x_{ij})$ is the SHAP value of x_{ij}. Intuitively, $f(x_{i1})$ is the contribution of the first feature in the i-th sample to the final predicted value y_i. When $f(x_{i1}) > 0$, it indicates that this feature has increased the predicted value and has a positive effect; otherwise, it means that the feature has reduced the predicted value and has a negative effect.

Traditional feature importance only tells us which feature is important, but we do not know how that feature affects the prediction results. The biggest advantage of the SHAP method is that it can reflect the positive or negative impact of features in each sample.

4.2 Visualization with SHAP

The shap library provides a variety of visualization methods to interpret the output of models. We want to adapt these charts in our dashboard. However, most of the plots in SHAP are implemented with matplotlib, which generates essentially static images, neither interactive nor easy to modify.

In this project, I reimplemented four kinds of shap's visualization methods with Plotly, in which way model interpretation diagrams can be embedded in web pages as interactive, customizable components. Plotly is an open-source graphing library that enables users to create interactive, web-based visualizations with simple programming interfaces. It supports multiple programming languages, both Python and JavaScript.

The workflow is shown below. In the back end, we use Plotly.py to create preliminary charts based on SHAP values and send them to the front end as JSON objects. Upon receiving the Plotly charts, the front end makes detailed adjustments according to the UI style.

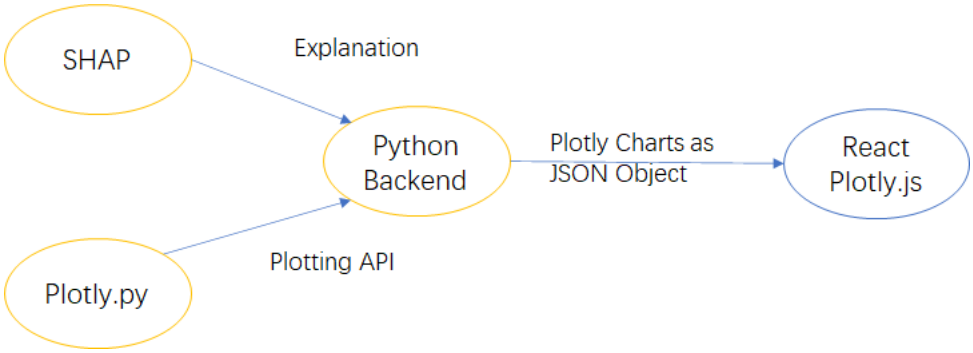

Figure 2: The Workflow of Visualization

In the following sections, I will introduce these four plots one by one.

4.3 Waterfall Plot

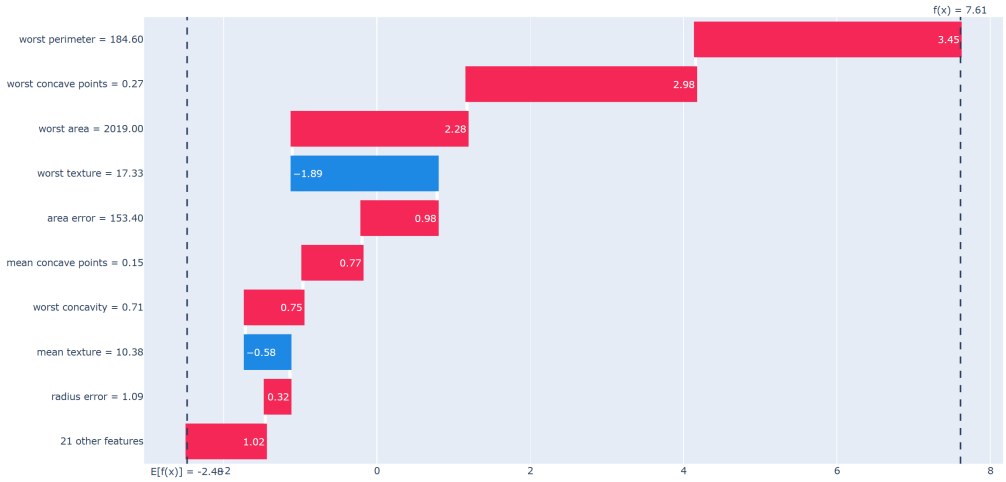

Figure 3: An example of Waterfall Plot

For a given sample, the waterfall plot illustrates how each attribute affects the final output of the model based on the mean prediction. As you can see in the example, the mean prediction is $E[f(x)] = -2.48$ and the model's output is $f(x) = 7.61$. f(x) is positive, meaning that this sample is classified as positive.

The Y-axis ticks are the attributes of this sample, sorted by the absolute value of their SHAP values. For example, the "worst perimeter" is the one that has the greatest impact on the model output, with a value of 184.6. It increases the model output by 3.45 on top of the impact of the other features.

Some features make $f(x)$ lower, i.e. bring this sample closer to the negative class, indicated by blue bars in the figure. For those features with relatively low impact, the sum of their shap values is represented in the bottom row, which is "21 other features" in this case.

4.4 Summary Plot

The summary plot shows for each attribute the distribution of the value and corresponding SHAP value across the dataset. The Y-axis ticks are sorted by the sum of the absolute value of SHAP values for a given attribute.

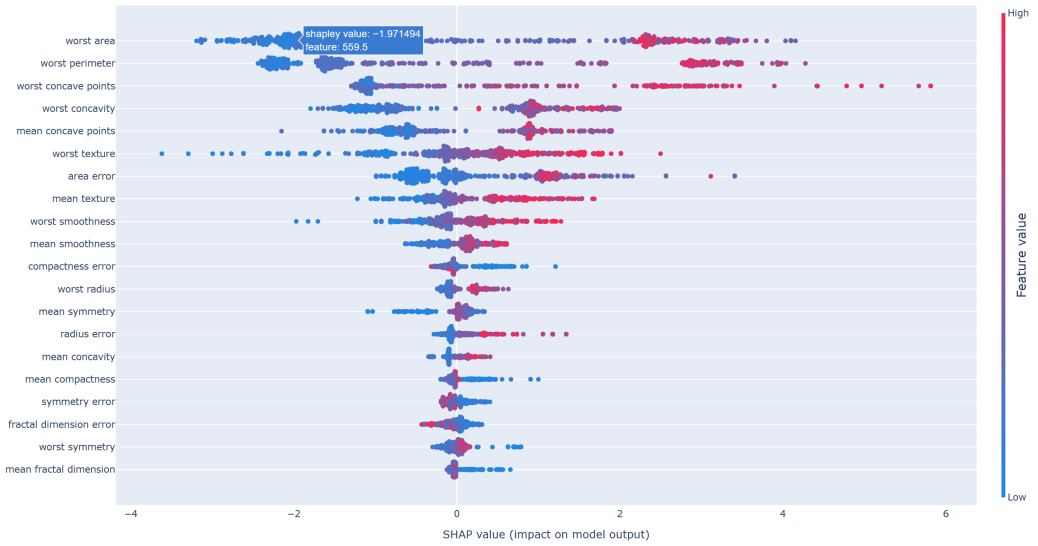

Figure 4: An example of Summary Plot

For each feature, we can visualize the distribution of its SHAP values from the distribution of points. The color of the dot further demonstrates the relationship between the feature value and its value. For example, for the attribute "worst area", when its SHAP value is positive, the color of the points is largely red; when its SHAP value is negative, the color of the points is blue. Since red indicates a larger feature value, we can infer that a larger "worst area" will lead to a more likely positive sample.

4.5 Decision Plot

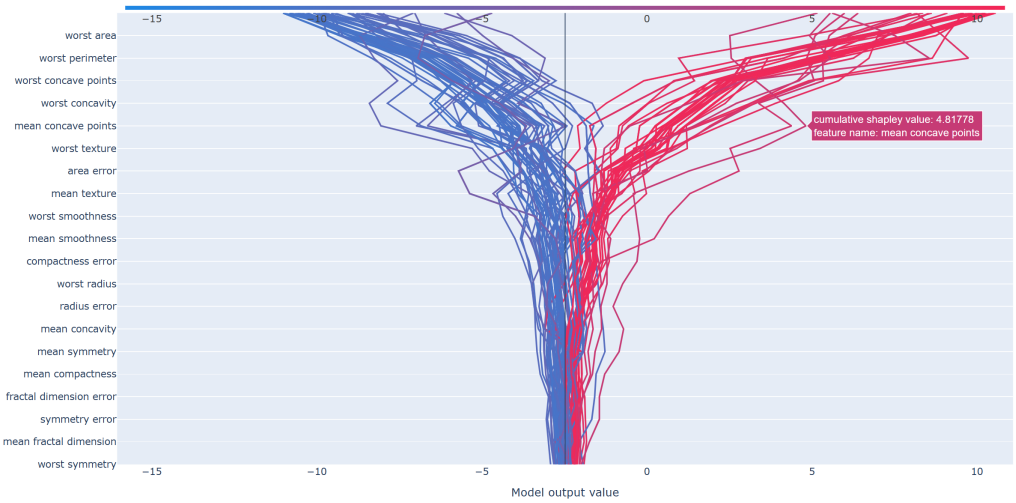

Figure 5: An example of Decision Plot

The decision plot is the globalized and line graph version of the waterfall plot. Each line represents a sample, while the black vertical line indicates the mean output $E[f(x)]$ of the model. The changes in the line graph show how each attribute affects the output of the model. The color of the line indicates the output of the model $f(x)$ for this sample. The Y-axis ticks are sorted by the sum of the absolute value of SHAP values for a given attribute.

We cannot include all samples in the decision plot and summary plot. Since each point in the chart is interactive, having too many elements can negatively impact front-end performance. Therefore, we only extract a portion of samples from the dataset for plotting. This is already sufficient to reflect the overall trend of the entire dataset.

4.6 Local Dependence Plot

The local dependence plot shows the effect of the value of an attribute on the model output. The red bar chart shows the distribution of values of this feature over the entire dataset. The blue curve indicates the change of the model's mean prediction when the feature is a given value (i.e., the x-axis coordinate).

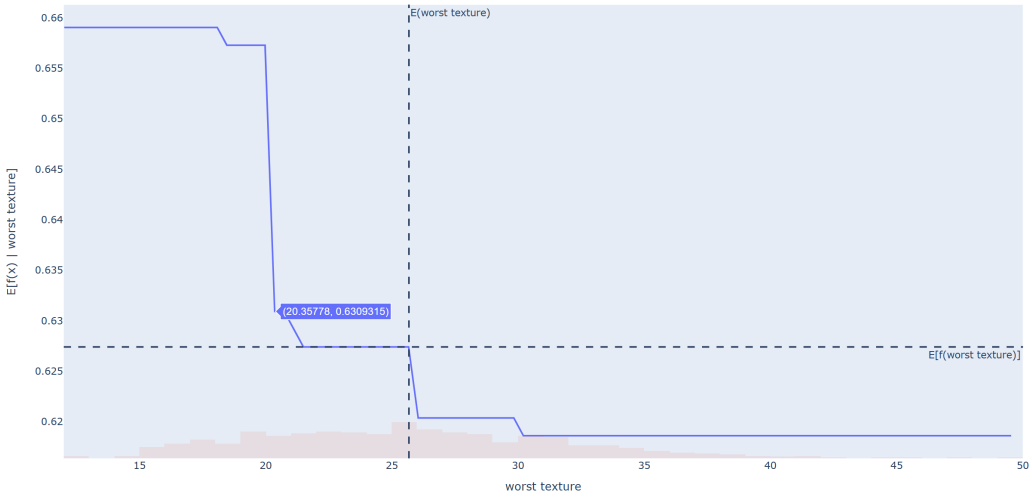

Figure 6: An example of Local Dependence Plot

5 Fairness Dashboard: Bias Detection

If we want our AI system to be free of bias and discrimination, the first step to take is to test whether there is a bias. Taking existing software and libraries as references, I summarized their workflows below.

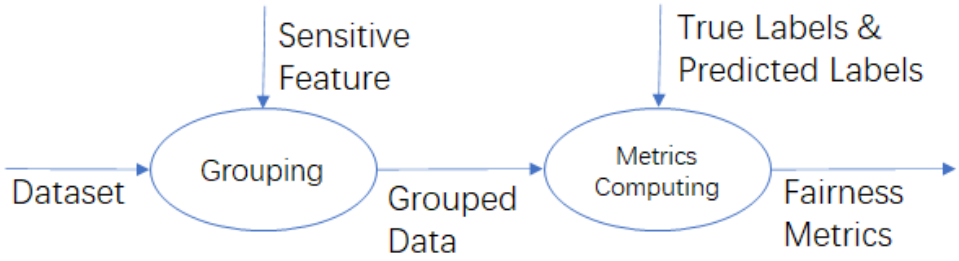

Figure 7: The Workflow of Bias Detection

The workflow contains two steps. First, the user should select a certain column of the dataset, which is called the sensitive feature. The dataset will be separated into groups based on the sensitive feature. If the selected column has too many unique values, we split them into several bins. In the second step, we calculate some metrics about the fairness of the model within and between those groups.

Suppose a company has developed a machine learning model to predict employee salaries based on their experiences. The company has collected a dataset of employee information

that includes gender, job role, education background and so on. The company wants to ensure that the model is free from bias and treats all employees fairly, regardless of their gender.

Firstly, the dataset is grouped based on gender to allow for a comparison of performance metrics between male and female subgroups. Secondly, the performance metrics are calculated based on the true labels and predicted labels of the model. By comparing these metrics between male and female subgroups, any potential biases in the model can be identified and addressed.

The fairness measurement[1] we use is listed in the table.

Table 1: Performance Metrics for each Group

Metric	**Definition**
Count	Number of samples
False Negative Rate	Proportion of samples predicted as negative but are actually positive
False Positive Rate	Proportion of samples predicted as positive but are actually negative
True Negative Rate	Proportion of samples predicted as negative and are actually negative
True Positive Rate	Proportion of samples predicted as positive and are actually positive
Accuracy	Proportion of correct predictions
Mean Prediction	Average predicted value
Selection Rate	Proportion of samples classified as positive.

Table 2: Performance Metrics between Groups

Metric	**Definition**
Demographic Parity Difference	The maximum difference in selection rate between any two groups.
Demographic Parity Ratio	The maximum ratio in selection rate between any two groups.
Equalized Odds Difference	The maximum difference between true positive rates/false positive rates of any two groups, whichever is greater.
Equalized Odds Ratio	The maximum ratio between true positive rates/false positive rates of any two groups, whichever is greater.
Equal Opportunity	$\frac{\text{true positive rate}}{\text{true positive rate} + \text{false negative rate}}$
Group Benefit	$\frac{\text{true positive rate} + \text{false positive rate}}{\text{true positive rate} + \text{false negative rate}}$

Most metrics only work for binary classification. For multi-classification, a simple solution is asking the user to classify multiple labels into privileged classes and non-privileged classes, in which way we can convert multiple classes into two classes.

The prototype of the front end is as the figure 8. Its interaction logic follows the workflow: first, the user clicks on a feature, then different metrics will be shown in the dashboard. Of course, more work needs to be done to enable users to use it more effectively.

6 Summary

This project has enhanced my engineering and teamwork skills. For the front-end, I implemented four kinds of reactive charts; For the back-end, I implemented the corresponding data collection part and the module for calculating fairness metrics. I have become more familiar with React programming through practice, which I knew very little about at the beginning of

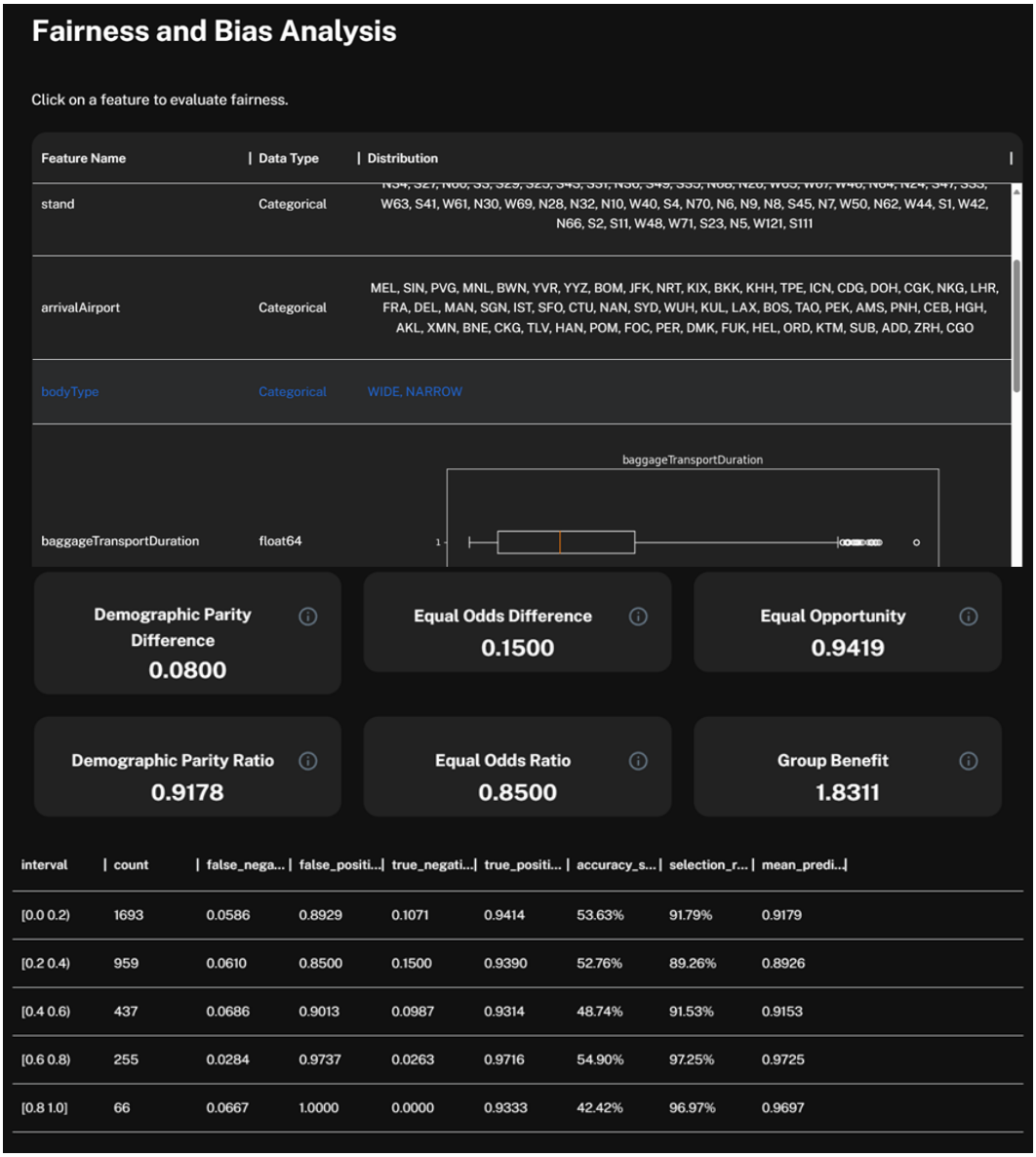

Figure 8: The Prototype of Bias Detection's Frontend

the project. I have worked very well with company staff and we built up precious friendships. In general, I am satisfied with the workload and the results of the project. Practice does bring more rewards than taking classes and exams.

7 Appendix A

7.1 Minutes of the 1st Project Meeting

Date 13 Feb (Mon)

Time 2:00 pm to 2:45 pm

Place Zoom Meeting

Present Zhang Hexiao, Prof. Raymond Wong, Hao Liu, Tim Lo, Nong Dingzhan

Note-taker Zhang Hexiao

The professor introduced administration issues of the independent project and the arrangement of four meetings. The CEO of GlassBox AI Ltd., Tim Lo, introduced the company and the background of the project. Nong Dingzhan, a software engineer from the company, introduced the technical details and workflow of the dashboard. We discussed the task of plotting interactive chats.

7.2 Minutes of the 2nd Project Meeting

Date 6 March (Mon)

Time 2:00 pm to 2:30 pm

Place Zoom Meeting

Present Zhang Hexiao, Prof. Raymond Wong, Tim Lo, Nong Dingzhan, Aaliya Raza

Note-taker Zhang Hexiao

Hexiao introduced the work done in this phase. During this stage, he set up the development environment, clarified the project's code structure, and laid a solid foundation for subsequent development. He implemented the waterfall plot using Ant Design Charts, but the degree of customization was limited, so he needs to look for a more suitable plotting library.

The professor asked about the specific directions of the optimized drawing implementation. The company's engineer Aaliya participated in the discussion.

7.3 Minutes of the 3rd Project Meeting

Date 27 March (Mon)

Time 3:00 pm to 3:55 pm

Place Zoom Meeting

Present Zhang Hexiao, Prof. Raymond Wong, Tim Lo, Nong Dingzhan, Aaliya Raza

Note-taker Zhang Hexiao

Hexiao presented the progress of the work. We implemented all four types of interactive charts needed for the dashboard. With the multi-language interface provided by Plotly, the plotting task is done in stages by the front-end and back-end, greatly improving the efficiency of development and collaboration.

The professor asked the student to explain the details and interpretation of the summary plot and decision plot. Aaliya helped Hexiao with further explanation.

7.4 Minutes of the 4th Project Meeting

Date 24 April (Mon)

Time 2:00 pm to 2:50 pm

Place Zoom Meeting

Present Zhang Hexiao, Prof. Raymond Wong, Tim Lo, Nong Dingzhan, Aaliya Raza

Note-taker Zhang Hexiao

Hexiao presented the progress of the work. Over the past weeks, we did some research on the existing applications and libraries for bias detection. Then taking them as references, Hexiao implemented a module to calculate fairness metrics in the backend. Specifically speaking, this includes grouping the input dataset by a certain sensitive feature and calculating fairness metrics within and between groups.

The professor asked the student to interpret the new dashboard. Participants discussed the direction of further improvements.

References

[1] Ghosh, A., L. Genuit, M. Reagan. Characterizing intersectional group fairness with worst-case comparisons. *CoRR*, abs/2101.01673, 2021.