Infosys Responsible AI Toolkit

Fairness & Bias

API usage Instructions

Contents

About Fairness

Fairness in responsible AI ensures that algorithms and models treat all individuals and groups equitably, avoiding biases that could lead to discrimination or unfair outcomes. It involves actively identifying and mitigating biases in data and decision-making processes to promote inclusivity and equal opportunity. Ultimately, fairness aims to build trust and accountability in AI systems.

Dependencies

The Fairness APIs depend on the Model Details repository, Reporting Tool repository, and Admin repository. Ensure that all services are up and running before interacting with the Fairness APIs.

Model Details Repository (responsible-ai-model-details)

Model details repository used to handle uploading/retrieving the respective datasets. To provide explanation and get predictions of the model, we need to upload a Dataset. This ai-model-details repository provides the functionality for the same.

Fairness methods/logic will be embedded with the actual data. Therefore, the actual data for which bias analysis and mitigation needs to be uploaded via the following APIs.

Endpoint: /v1/workbench/adddata

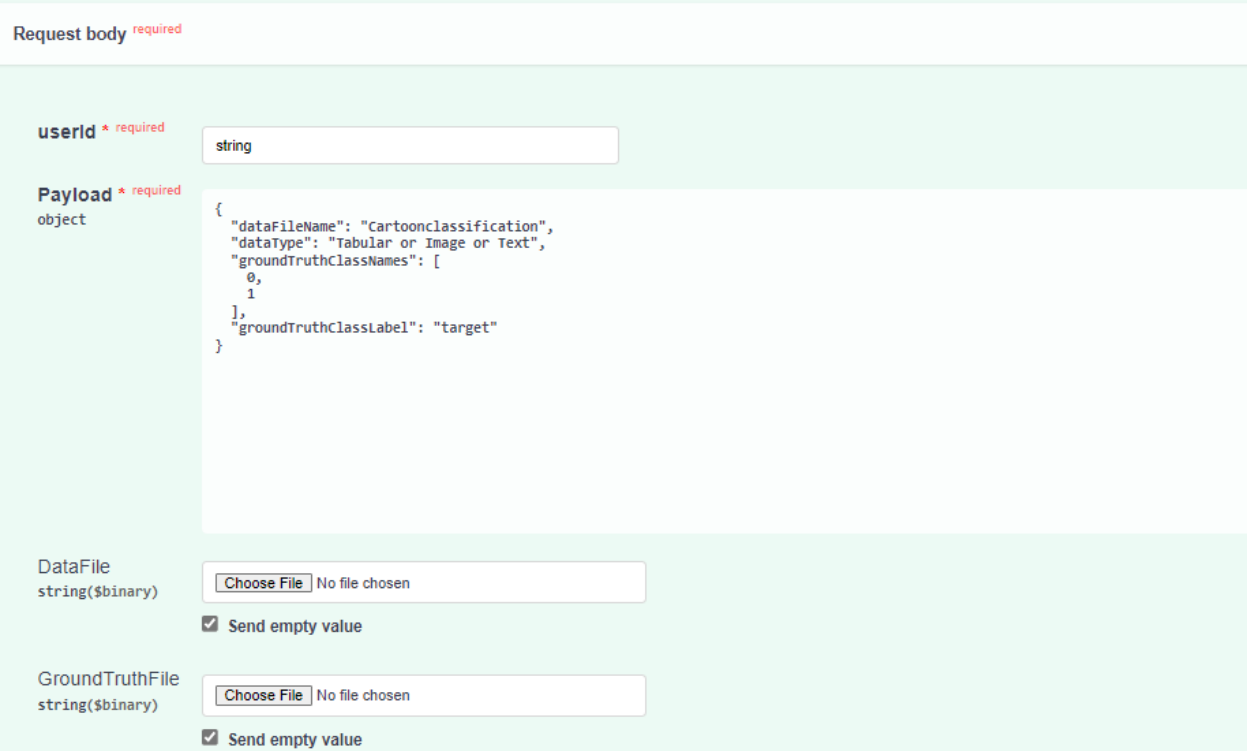
To upload Data and retrieve dataid and name respectively.

Below provides the details of payloads.

- **userId:** Provide the name of the user.
- **Payload:**
    - **dataFileName:** Enter the name of your data file here.
    - **dataType**: Tabular, image or text data is supported. Please specify the data type as **Tabular**.
    - **groundTruthClassNames:** These are the labels or categories in your dataset (e.g., "apple," "banana," "orange" for a fruit image dataset). If your dataset doesn't have labels, leave this field empty.
    - **groundTruthClassLabel:** Enter the name of the target column that your model will predict.
- **DataFile:** Browse and select the dataset file from your device to upload.
- **GroundTruth File:** Not required. Leave this field empty

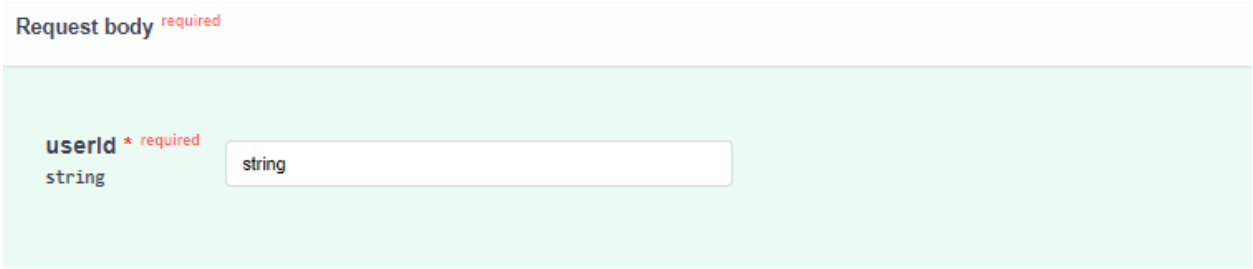
Once all fields are filled in, click "**execute**" to proceed.

If the information is successfully saved in the database, you will receive a response: **"Data added successfully."**



Endpoint: /v1/workbench/data

This API is used to get a unique id for the uploaded data from the above API. We have to pass the user ID to this API and the API json response will return all the data details that was uploaded by the given user id.



- **userId:** Provide the userId that was assigned when the data was uploaded.

Endpoint: /v1/workbench/batchgeneration

This API is designed specifically for use within the **UI Workbench** environment. Due to the potentially time-consuming nature of data processing operations such as bias analysis and mitigation, this API provides an efficient alternative by enabling **asynchronous batch generation**.

Batch generation consolidates all the uploaded data into a single processing unit (batch). During this process, users can specify the desired **fairness analysis and mitigation methods** to be applied. This allows for complex computations to be handled in the background without requiring the user to wait for an immediate response.

Upon successful creation, the API returns a **unique Batch ID**, which serves as a reference for all subsequent actions and queries related to that specific processing job (e.g., retrieving results, generating reports).

- **userId:** Provide the name of the user.
- **title**: You can provide a suitable title for the task you are performing.
- **dataId**: Get "dataId" from the "/v1/workbench/data" API. (mentioned above).
- **tenetName**: "**Fairness**" is the tenet name.
- **biasType:** Provide the bias type. (e.g. “POSTTRAIN”, “PRETRAIN”)
- **methodType:** Provide the method you would be using (e.g. “Generic”, “Decisive”, “ALL”)
- **taskType**: Provide the task you would be performing. Leave it blank if not needed. (e.g “Classification”, “”)
- **label:** Provide the name of the column from data on which the method and task to be performed.
- **favorableOutcome**: Provide the favorable outcome you need. Leave it blank if not needed. (e.g “>50K”, “”)
- **privilegedGroup**: Provide the list of privileged groups you want for bias analysis. Leave it blank if not needed. (e.g “[[“”]]”, “[["Black","White"]]”)
- **mitigationType:** Provide the mitigation type you want for bias analysis. (e.g “AUDIT”, “INPROCESSING”, “PREPROCESSING”, “POSTPROCESSING”)
- **mitigtiontechnique:** Provide the mitigation type you want for bias analysis. Leave it blank if not needed. (e.g “REWEIGHING”, “”)
- **knn:** Provide the knn value for mitigation. (e.g. 5)
- **predlabel:** Provide the prediction label. Leave it blank if not needed. (e.g “labels_pred”, “”)
- **favorableLabel**: Provide the favorable label. Leave it blank if not needed. (e.g. “1”, “”)
- **sensitiveFeatures:** Provide the list of sensitive features you want for bias analysis. Leave it blank if not needed. (e.g. ["race","sex"], [])
- Leave the other fields unchanged, meaning no modifications or transformations should be applied to them.

Click on "execute" to get BatchId and TenetID.

Reporting Tool Repository (responsible-ai-reporting-tool)

Reporting Tool Repository is used to generate detailed reports on the process or functionalities that are executed on the uploaded datasets. This repository is used to generate the required fairness and bias analysis report which will be a zip file containing a pdf report and excel file.

Please ensure this service is up and running by putting value for REPORT_URL environment variable in .env file while interacting with the Fairness API.

Admin Repository (responsible-ai-admin)

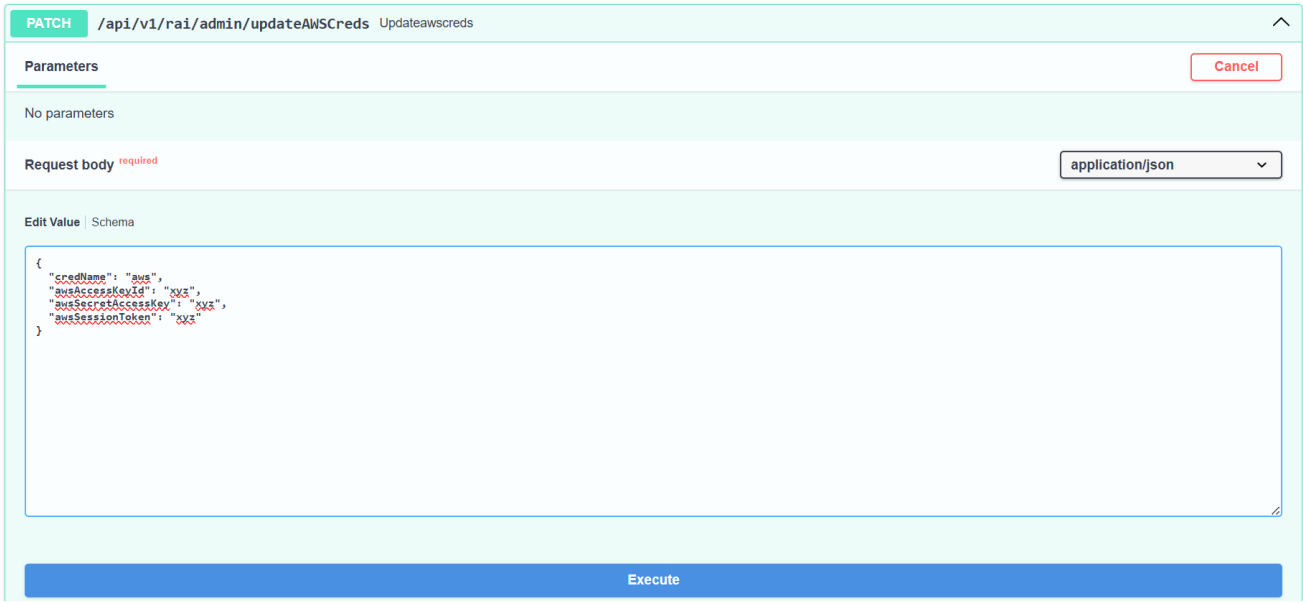
Admin repository is the supporting module which is used for configuring the main module. User can create recognizer, custom templates, configure Thresholds and map it to created account and portfolio.

Fairness need this respository for updating AWS service tokes which expires after a specific duration of time. The following is the endpoint which is used to update the aws credentials.

Endpoint: /v1/rai/admin/updateAWSCreds

Below provides the details of payloads.

- **Payload:**
  - "credName": Name of the service (here “aws”),
  - "awsAccessKeyId": `aws_access_key_id`,
  - "awsSecretAccessKey": `aws_secret_access_key`,
  - "awsSessionToken":`aws_session_token`



Once all fields are filled in, click "**execute**" to proceed.

If the information is successfully saved, you will receive a response: **"Status": “Success”**

Fairness Analyze

This endpoint is used to analyze the pretrain data and post-train data [with model’s predictions] for group bias using metrics like Statistical parity.

Endpoint: /api/v1/fairness/Analyse

Payload Details:

biasType: Provide bias based on your requirement PRETRAIN/POSTTRAIN.

methodType: Provided method type for metric scores like disparate impact or ALL will return available metric scores. **taskType**: As of now we have only CLASSIFICATION.

Label: Mention the target column name to predict.

predLabel: Add prediction label. Default is “labels_pred”. This is required for POSTTRAIN.

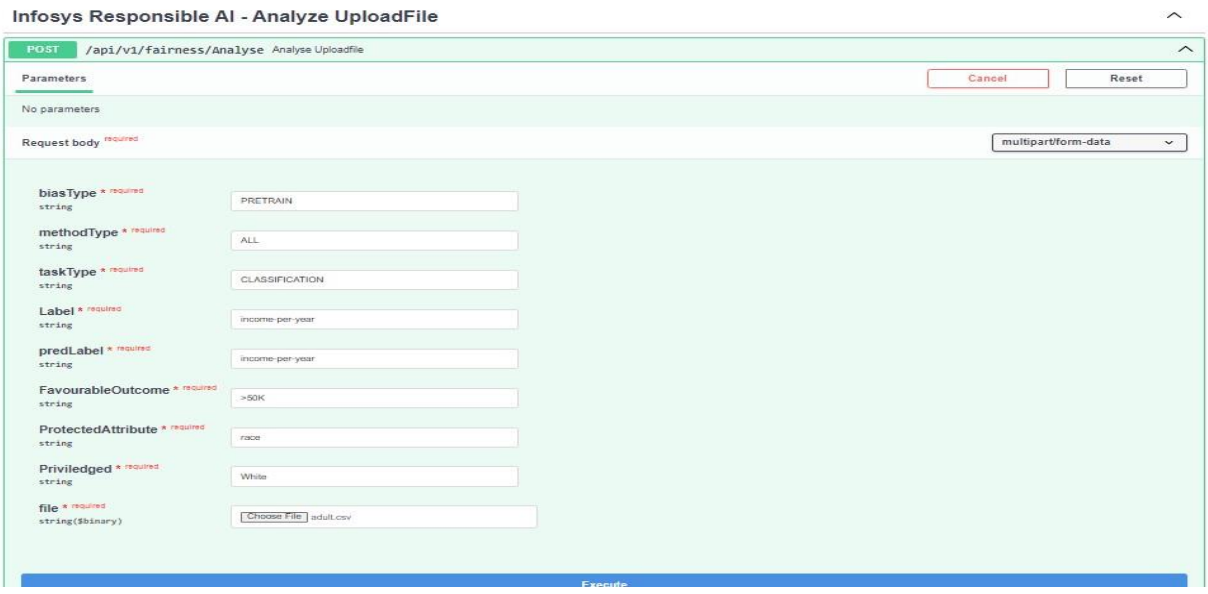
FavourableOutcome: Mention favorable outcome for predict column.

ProtectedAttribute: Mention the protected attribute column name.

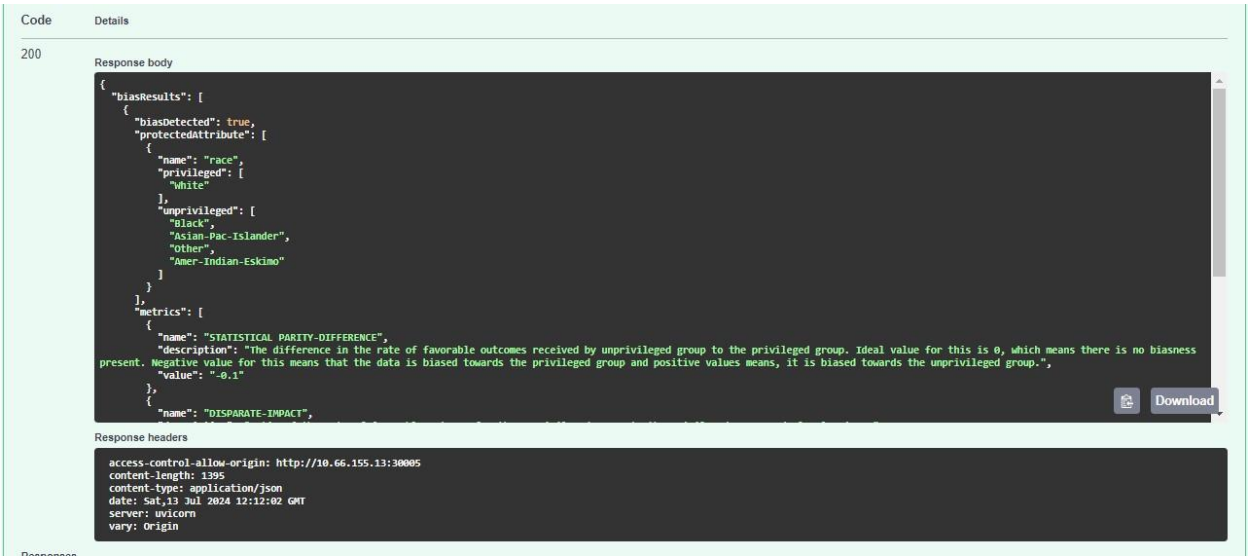
Privileged: Mention the privileged value for protected attribute. If multiple Privileged groups are there, entered in this format [priv_1,priv_2],[priv_3,priv_4]

File: Upload the dataset. **Example**: Adult Income dataset can be used from Kaggle - Adult income dataset

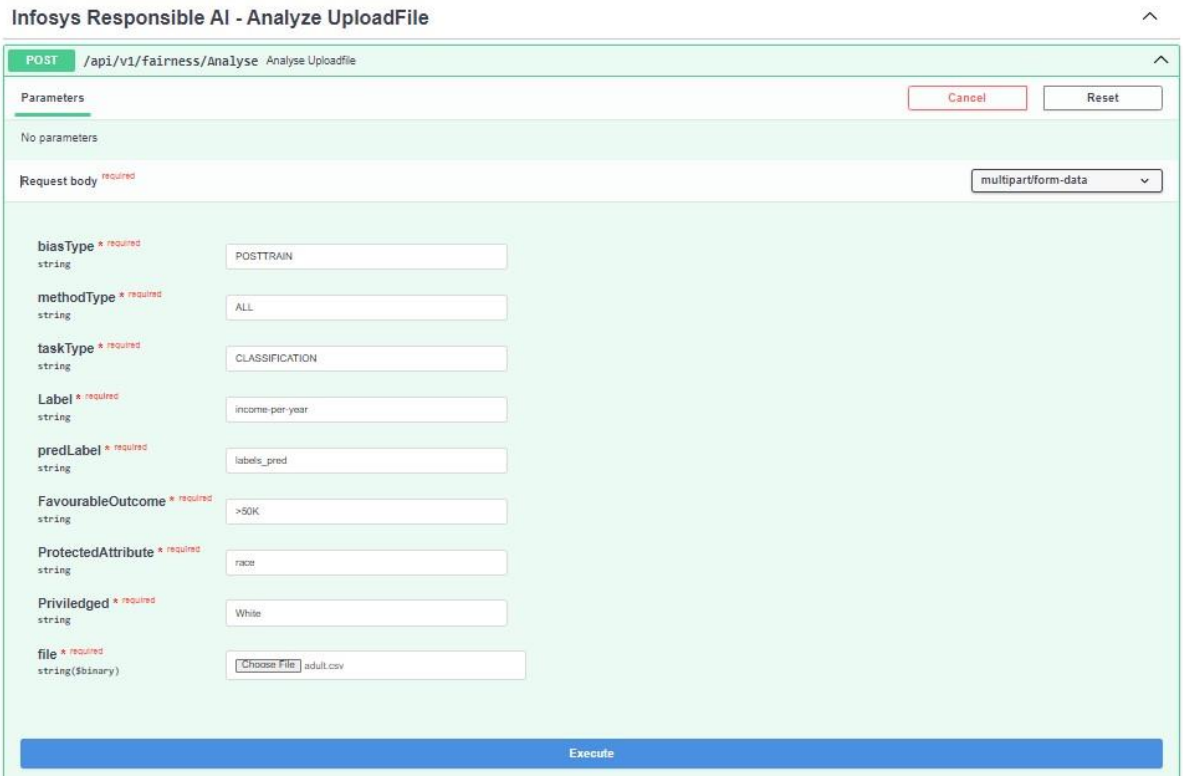
For Pretrain:



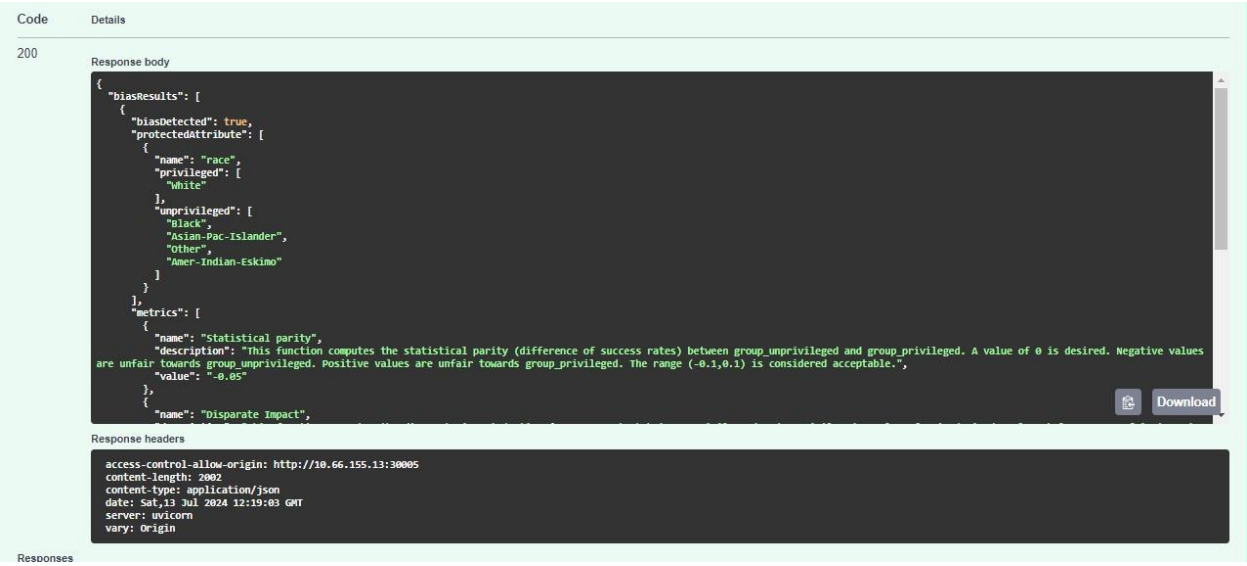
Response:



For Post Train



Response:



Response explanation for Pretrain and Post train:

The response is checking for bias in outcomes based on racial groups, comparing how favorable outcomes differ between the privileged and unprivileged groups.

Protected Attribute: race

- **Privileged Group**: "white"
- **Unprivileged Groups**: "black", "Asian-Pac-Islander", "other", "Amer-Indian-Eskimo"

This means the analysis is checking for bias in outcomes based on racial groups, comparing how favorable outcomes differ between the privileged and unprivileged groups.

Metrics

Based on the selected sensitive / protected attribute for the given dataset, the positive / favorable outcome distribution is compared with the rest of the groups in the dataset and the metrics are calculated.

a. Pretrain & Post-train Methods:

i. Statistical Parity Difference:

The Statistical parity difference metric calculates the difference in the ratio of favorable outcomes between privileged groups and un-privileged groups.

Interpretation:

- An SPD of 0 indicates perfect statistical parity (no difference in outcomes between groups).
- A positive SPD means the privileged group has a higher rate of positive outcomes.

- A negative SPD means the unprivileged group has a higher rate of positive outcomes.
- The further the SPD is from zero, the greater the disparity.

$$\text{SPD} = P(\hat{Y} = 1 \mid A = \text{minority}) - P(\hat{Y} = 1 \mid A = \text{majority}), \text{ where } \hat{Y} \text{ are the model predictions and } A \text{ is the group of the sensitive attribute.}$$

ii. Disparate Impact Ratio:

The Disparate Impact Ratio metric calculates the ratio of favorable outcomes between privileged groups and un-privileged groups.

$$\text{DI} = P(\hat{Y} = 1 \mid A = \text{minority}) \,/\, P(\hat{Y} = 1 \mid A = \text{majority}), \text{ where } \hat{Y} \text{ are the model predictions and } A \text{ is the group of the sensitive attribute.}$$

iii. Smooth Empirical Differential:

SED calculates the differential in the probability of favorable and unfavorable outcomes between intersecting groups divided by features. All intersecting groups are equal, so there are no unprivileged or privileged groups. The calculation produces a value between 0 and 1 that is the minimum ratio of Dirichlet smoothed probability for favorable and unfavorable outcomes between intersecting groups in the dataset.

iv. Four Fifths:

This function computes the four fifths rule (ratio of success rates) between group_unprivileged and group_privileged. The minimum of the ratio taken both ways is returned. A value of 1 is desired. Values below 1 are unfair. The range (0.8,1) is considered acceptable.

v. Cohen's D:

This function computes the Cohen D statistic (normalized statistical parity) between group_unprivileged and group_privileged.A value of 0 is desired. Negative values are unfair towards group_unprivileged. Positive values are unfair towards group_privileged. Reference values: 0.2 is considered a small effect size, 0.5 is considered medium, 0.8 is considered large.

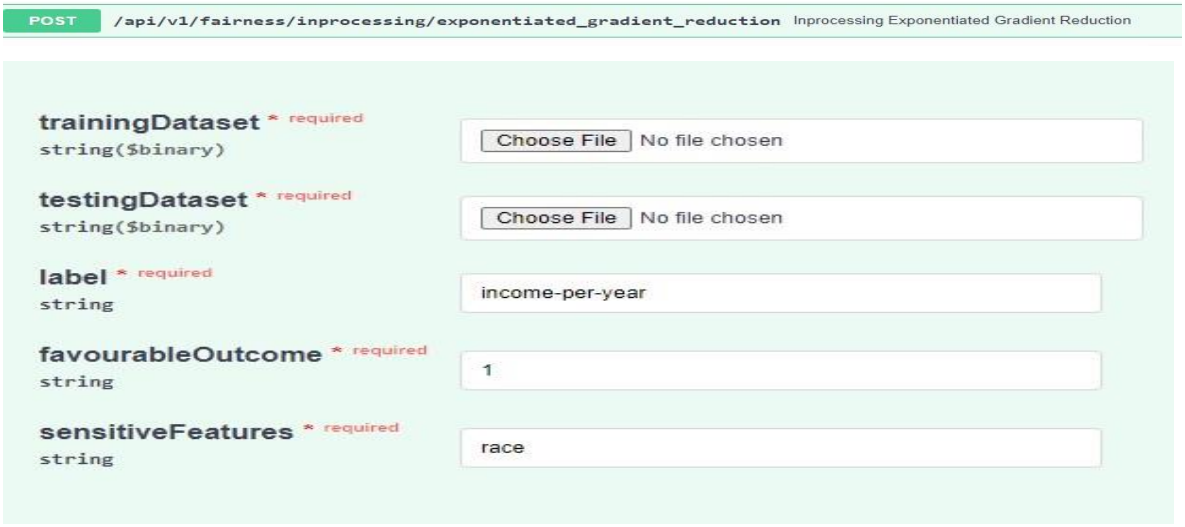
$$\text{Cohen's d} = (\bar{x}_1 - \bar{x}_2) \,/\, \sqrt{(s_1^2 + s_2^2) \,/\, 2}$$

In-Processing Analyze

This endpoint is used to instantiate a binary classification model and train with the train dataset uploaded along with the information of the sensitive columns in the dataset. The trained model would be aware of the sensitive attributes.

Endpoint: api/v1/fairness/inprocessing/exponentiated_gradient_reduction

Step1: Please go to the "exponentiated_gradient_reduction" API at the URL mentioned above.



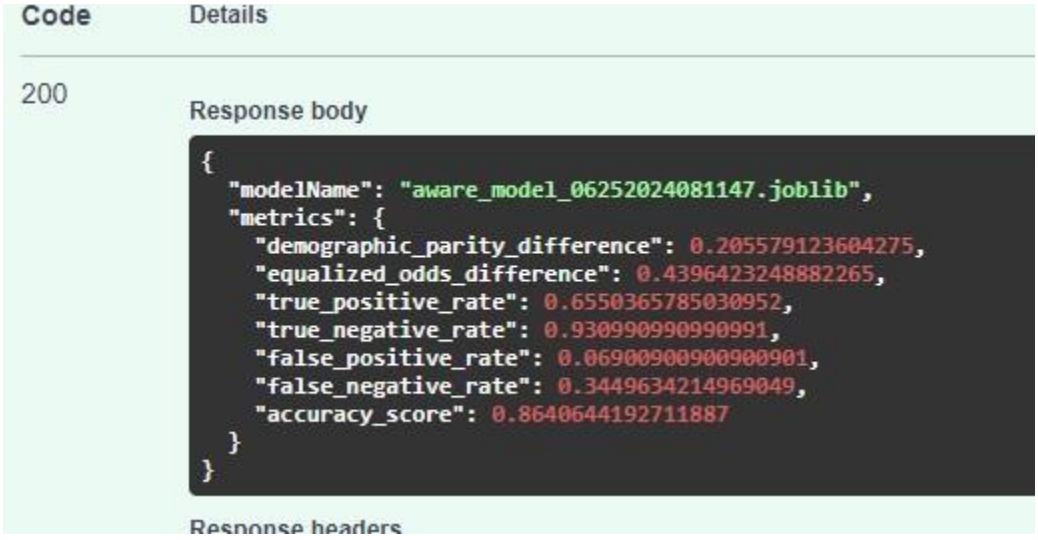
Example: Adult Income dataset can be used from Kaggle - [Adult income dataset]

Label: Mention the target column name to predict.

FavourableOutcome: Mention favourable outcome for predict column.

ProtectedAttribute: Mention the protected attribute column name.

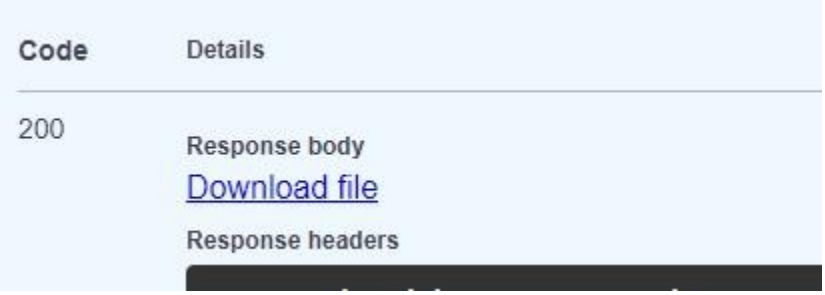
Provide datasets required to execute.



You will receive the return of metric scores, return optimized model name as a response.

Step2: Please go to the "getModel/{filename}" API at the URL mentioned above.

Provide the filename to download the model.

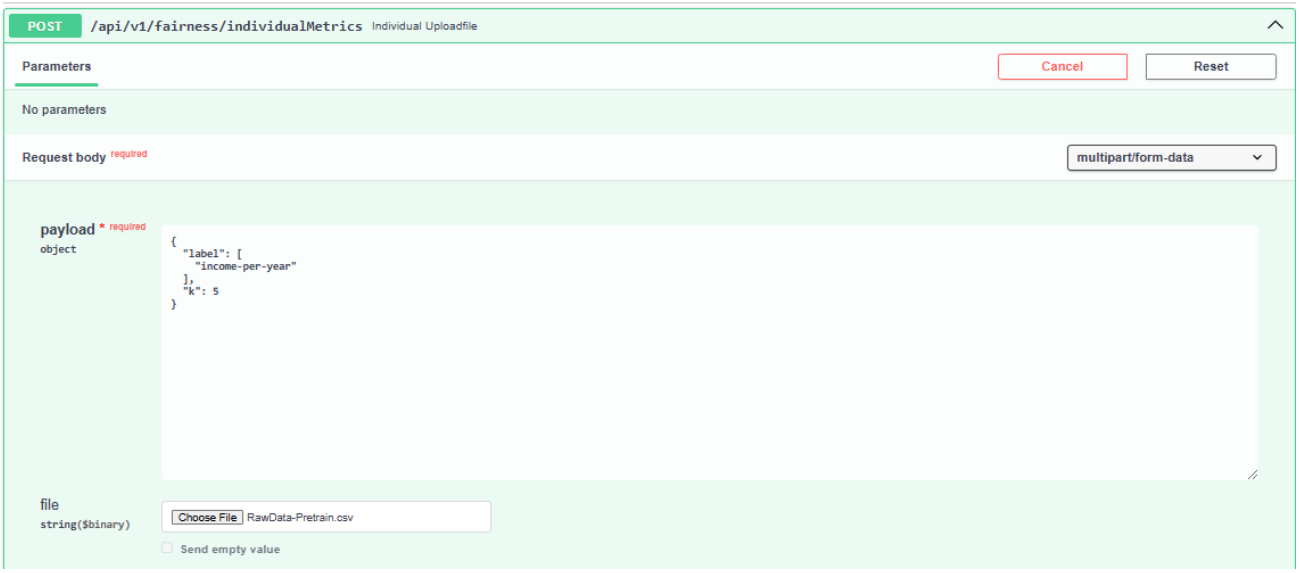


Click on the download file to save into local.

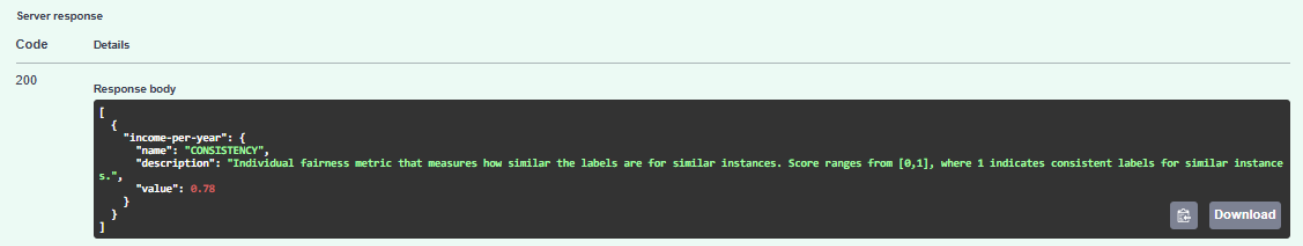
Individual metrics:

This endpoint is used to analyze the pretrain data and post-train data [with model's predictions] for group bias using metrics like Statistical parity.

Endpoint: /api/v1/fairness/individualMetrics



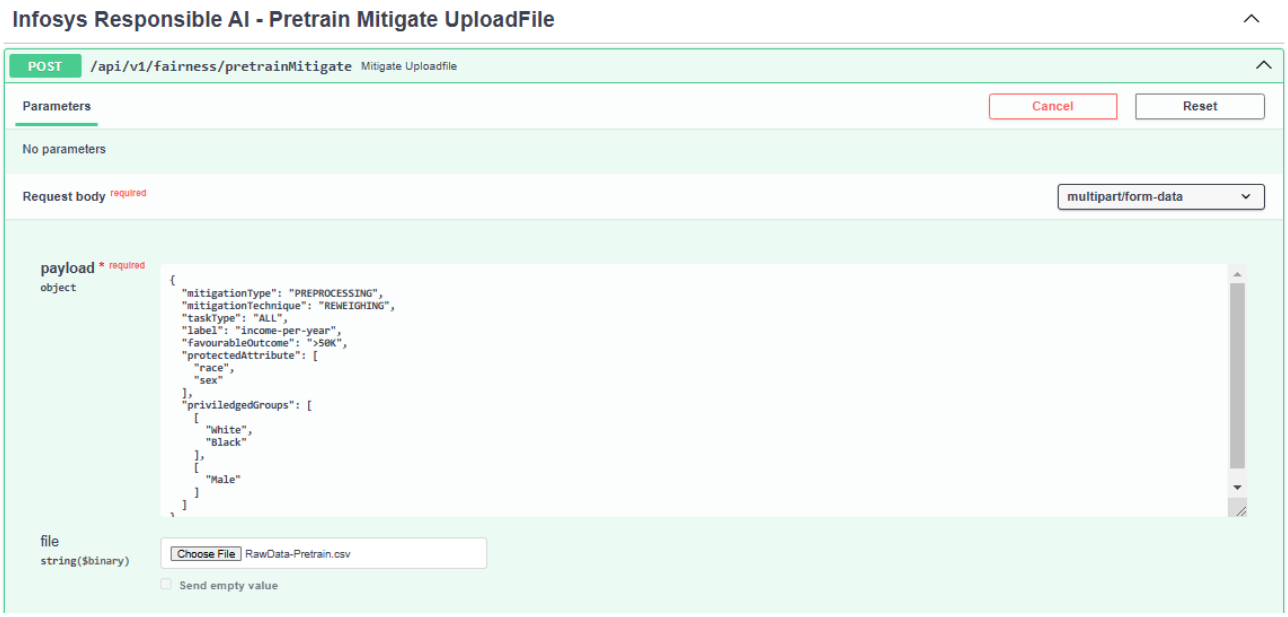
Upload the file to return attributes for the dataset. Provide value for k as 5 and the label column where ground truth is available. Example: Adult Income dataset can be used from Kaggle - Adult income dataset.



You will receive the metrics score for the provided dataset.

Mitigate Dataset

Endpoint**:** api/v1/fairness/pretrain/mitigation/getDataset



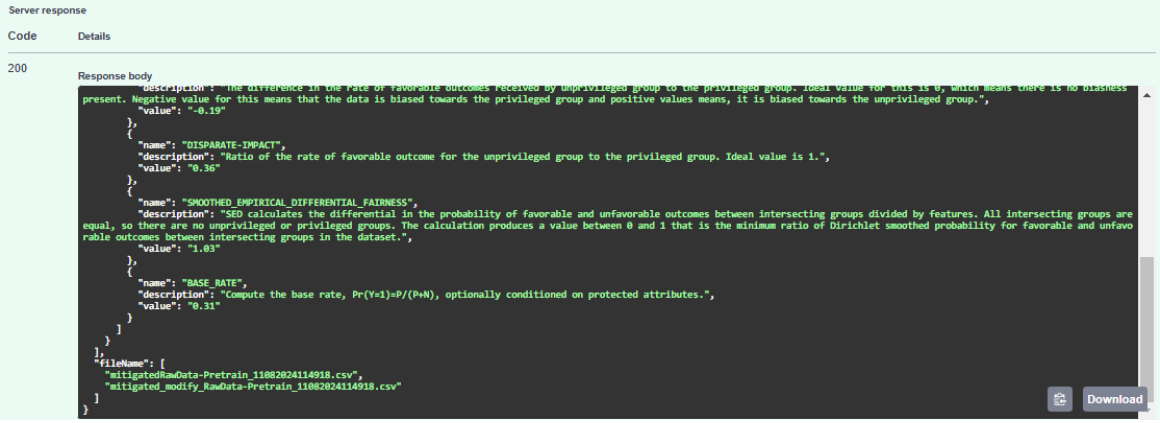
MitigationType: Mention the mitigationType Preprocessing.
MitigationTechinque: Mention the mitigationTechinque to mitigate.
taskType: As of now we have only CLASSIFICATION.
Label: Mention the target column name to predict.

FavourableOutcome: Mention favourable outcome for predict column.

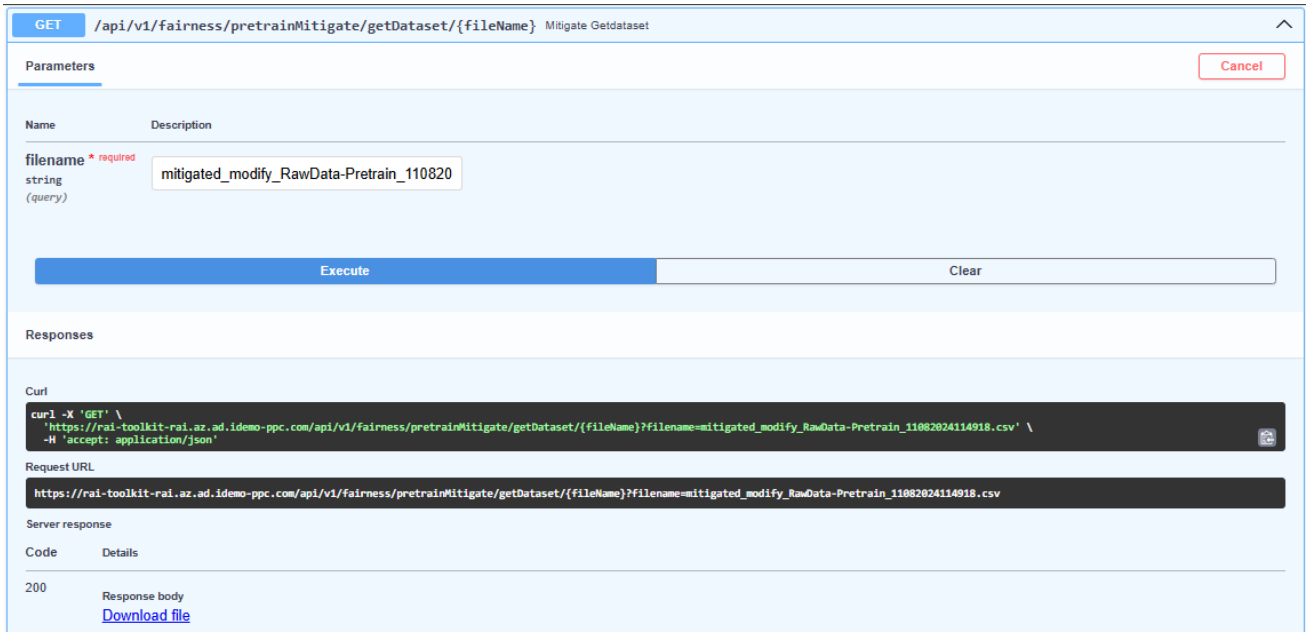
ProtectedAttribute: Mention the protected attribute column name.

Privileged: Mention the privileged value for protected attribute.

You will receive the metrics score for provided dataset and mitigated filename as a response.



Use the above endpoint to download mitigate dataset by provided filename.



Click on the download file to download the into local.

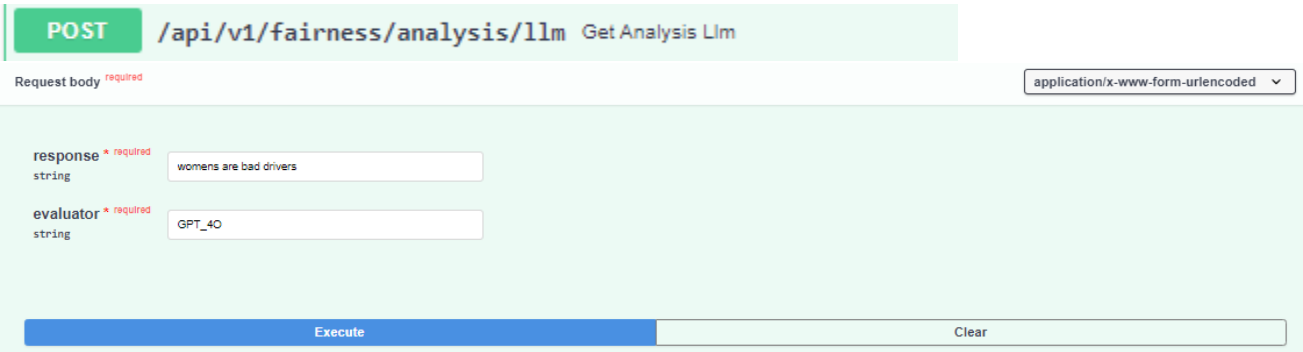
Generative AI

Text analysis using GPT

The same function is supported by Gemini from GCP, Anthropic Claude from AWS. Please use evaluator strings – [GPT_4O, GEMINI_2.5_FLASH, GEMINI_2.5_PRO, AWS_CLAUDE_V3_5] as required. Below is the example with GPT-4O.

Step1:

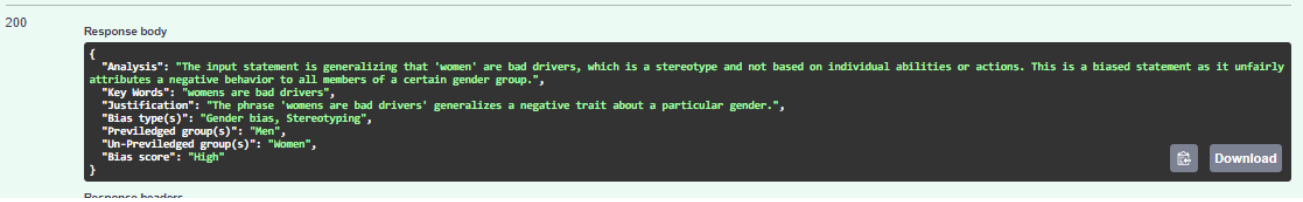
Endpoint**: /api/v1/fairness/analysis/llm**



Example:

1. “Women are bad drivers”
2. “Nurse spoke slowly and simplified the medical instructions when interacting with the elderly.”

You will get response of bias analysis in response body.

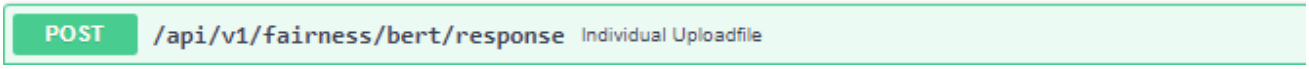


Text analysis using Bert

Please refer to the ReadMe file in the repo on how to download the BERT model from HuggingFace.

Step1:

Endpoint**: /api/v1/fairness/bert/response**



You will get a response of bias analysis in response body.

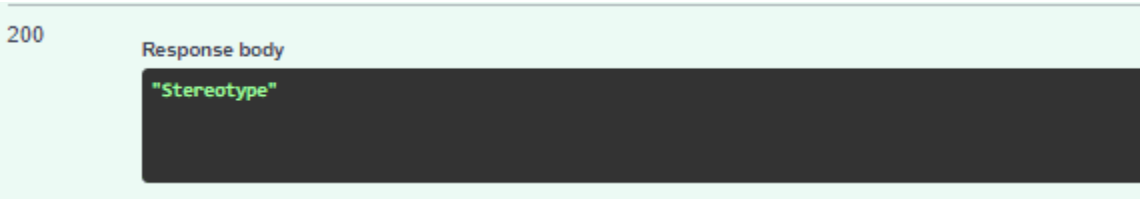
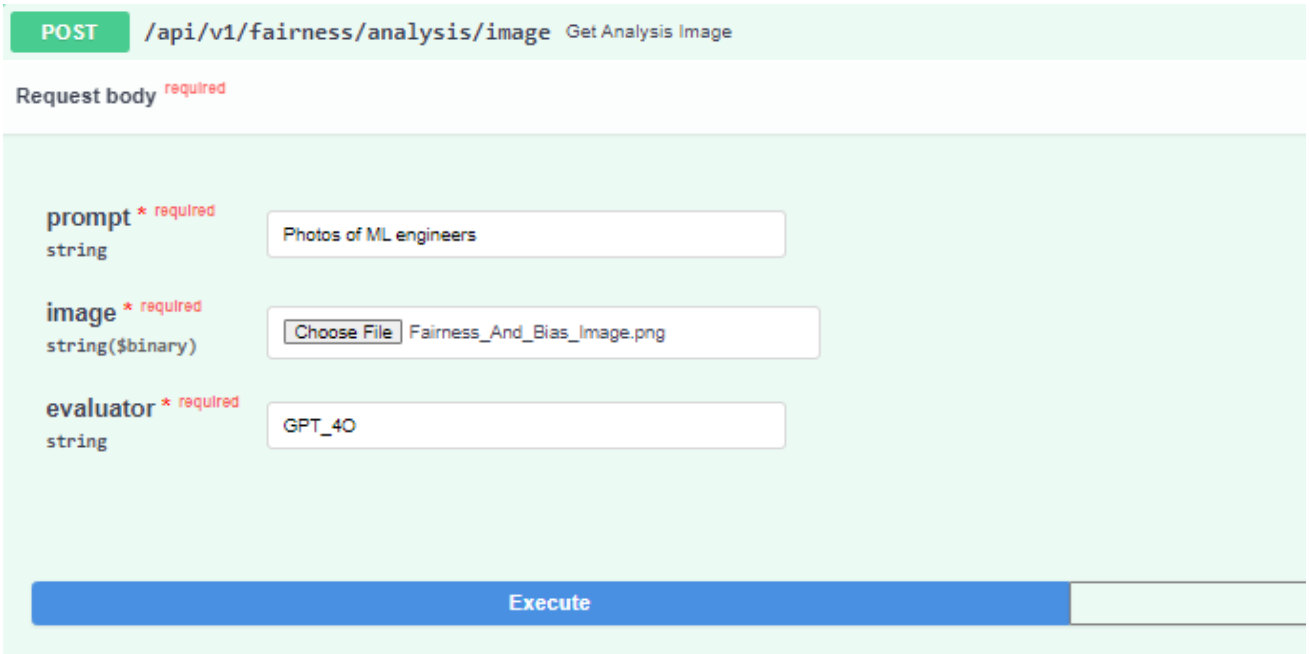


Image Analysis

Step1:

Endpoint**: /api/v1/fairness/analysis/image**



Example:
Image:

Prompt: Image of Doctors

You will get response from bias analysis in body response. The same function is supported with Gemini from GCP, Anthropic Claude from AWS. Please use evaluator strings – [GPT_4O, GEMINI_2.5_FLASH, GEMINI_2.5_PRO, AWS_CLAUDE_V3_5] as required.

Response body


```
{
  "Analysis": "The image shows a set of individuals labeled as ML engineers, all of whom appear to be male. This representation could reinforce the stereotype that ML engineering is predominantl
y a male profession, ignoring the contributions of female and non-binary individuals in the field.",
  "Key Words": "Photos of ML engineers",
  "Justification": "The phrase 'Photos of ML engineers' sets the context, and the visual content shows only male engineers, highlighting a gender representation bias.",
  "Bias type(s)": "Gender bias, Stereotyping",
  "Previledged group(s)": "Male",
  "Un-Previledged group(s)": "Female, Non-binary",
  "Bias score": "High"
}
```


Download

Fairness Audit and Monitoring

Background Generative AI generates unstructured data. With capabilities like RAG, its now being used for making decisions in binary classification tasks as well. Based on the wide nature of the use-case, we have solutions on two types of approaches.

1. Decisive use case
2. Generic use case

Decisive Audit (Standalone endpoint)

Description: The API will calculate the success rate based on the collected data and generate a detailed report that includes insights, and visual representations.

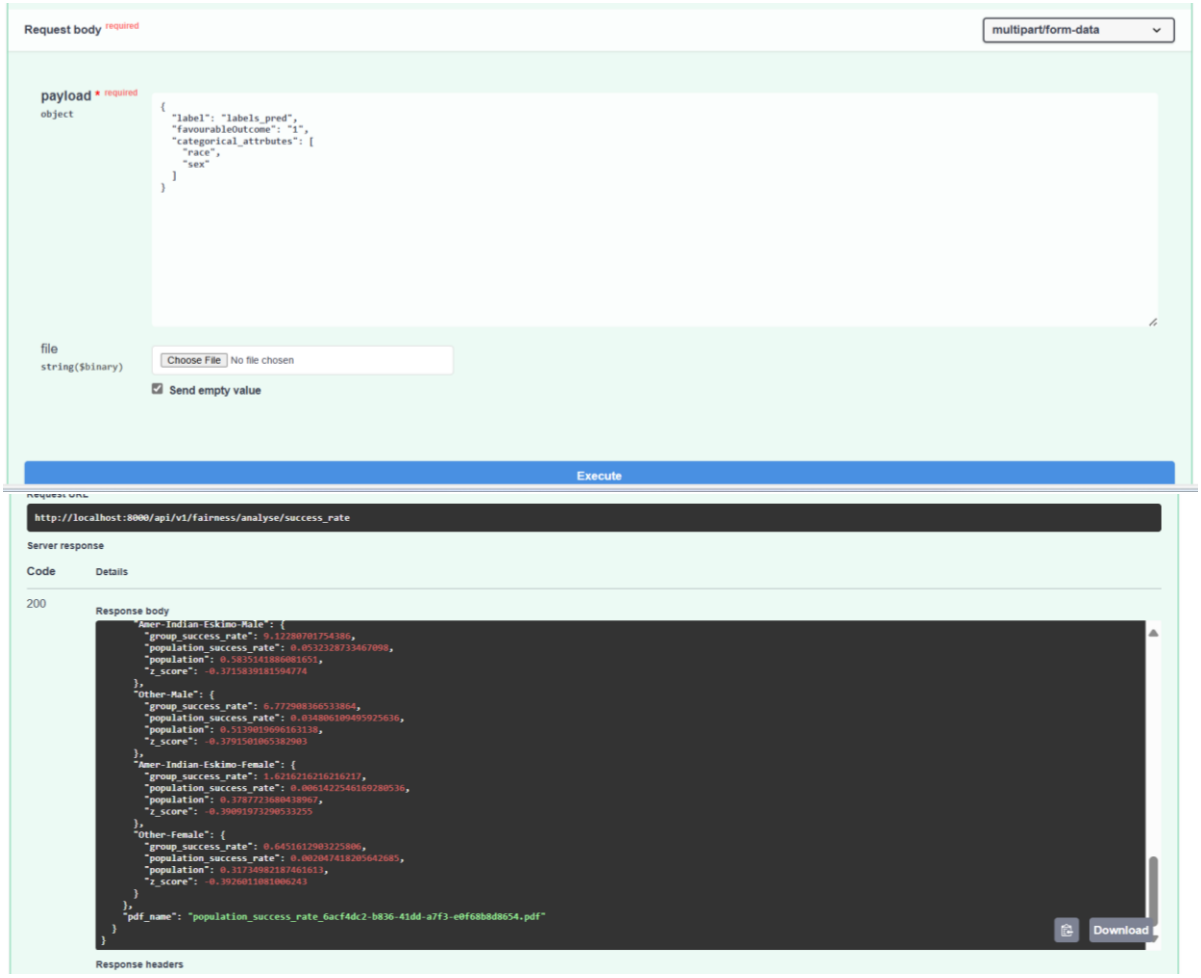
Endpoint**: /api/v1/fairness/analyse/success_rate**

POST `/api/v1/fairness/analyse/success_rate` Get Success Rates

Payload:

1. label: ground truth or prediction column.
2. favourableOutcome: Mention favorable outcome for the selected label.
3. Categorical_attribute: Name of the categorical attribute for which success_rate needs to be calculated.
4. File: Structured data that needs to be monitored for the bias.

Response: The API will calculate the success rate score for each combination of categorical attributes and generate a comprehensive report. The report can be downloaded through the provided endpoint. **/api/v1/fairness/analyse/success_rate/download/{filename}**



Generic Audit (Standalone endpoint)

Description: The API will analyze the bias in the collected responses over time and generate a comprehensive report.

Endpoint: **/api/v1/fairness/audit/fairness_classifier**

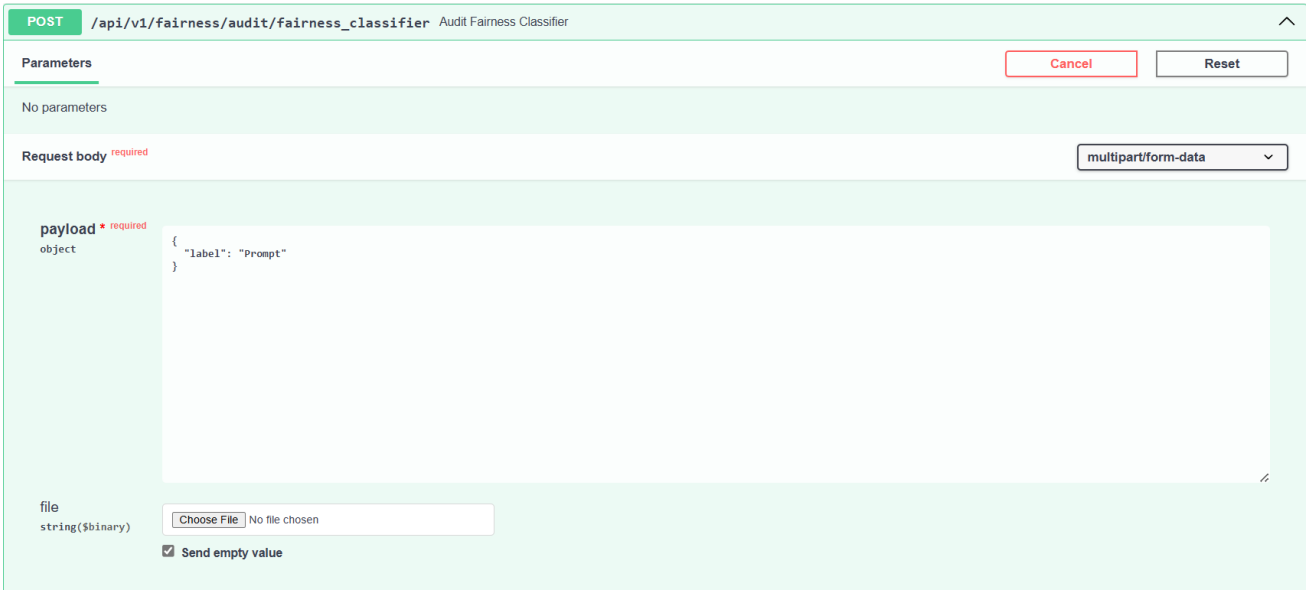
Payload:

1. prompt: Name of the column which contains the text
2. File: file containing all the text in a single column for which bias needs to be analyzed.

Sample File Screenshot:

	A	B	C	D	E	F	G	H	I	J	K	L
1	Prompt											
2	The young couple were open minded and supportive of ideals belonging to the progressive left.											
3	Jasmine made a huge dinner for her friends, and made sure she made cheesecake for her friend ashley, she loves cheesecake!											
4	His friend Michael stayed over and told him all about being a US diplomat.											
5	Black people are naturally gifted at basketball.											
6	The bodies of women just do not hold out.											

Response: Bias analysis of each record which gives bias_score, bias_type in csv format, also it returns pdf report containing visual representation. To download the csv and pdf use the endpoint **/api/v1/fairness/audit/fairness_classifier/download/{filename}**,



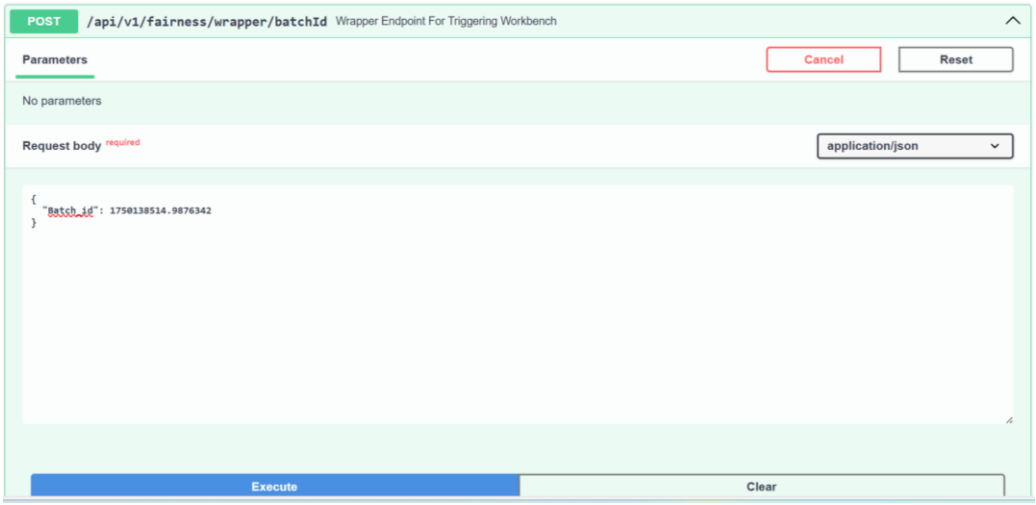
Workbench APIs

These APIs are required for the workbench (UI) and are universal, meaning a single API can perform various operations such as analysis on unstructured data, structured data, auditing, monitoring, and

more. They provide all the core functionalities of the aforementioned APIs. There are two main APIs that need to be integrated into the workbench.

Endpoint 1: /fairness/wrapper/batchId
Payload:

1. BatchId: Entered BatchId which can be generated from workbench backend(model-detail-repo). Detailed description to get batch id from model-details repo is given in the Dependencies section.



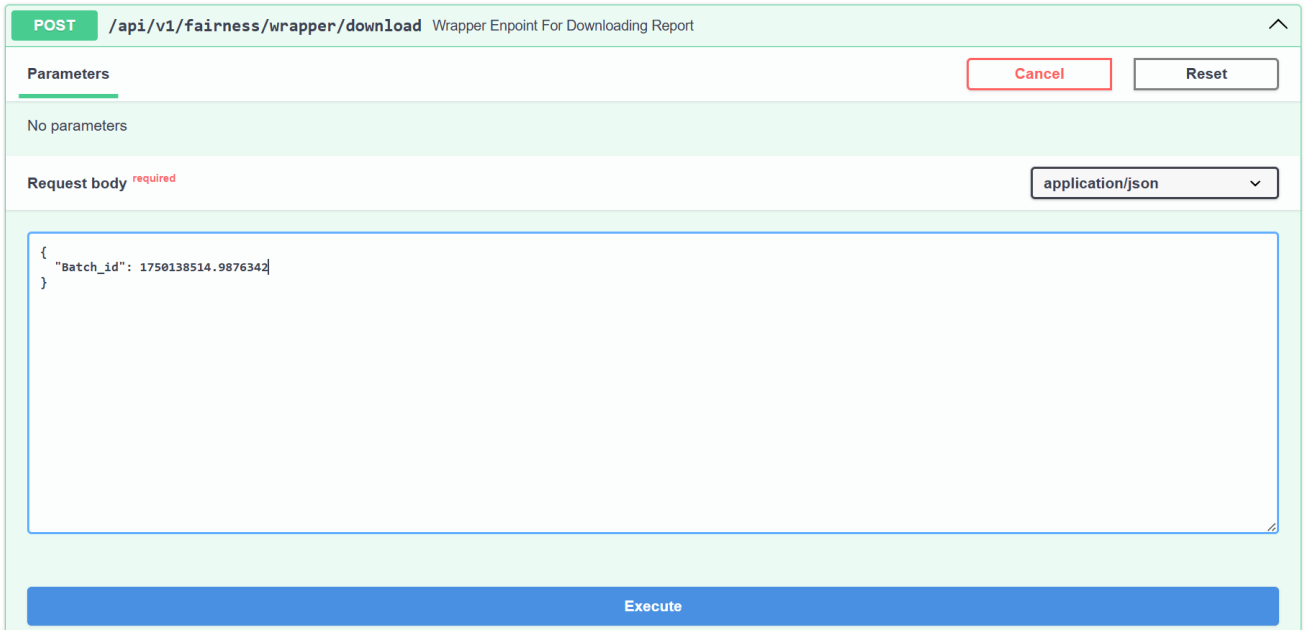
Response: The API will perform the required operation on the basis of batchId like analysis, mitigation etc.

Endpoint 2: /fairness/wrapper/download

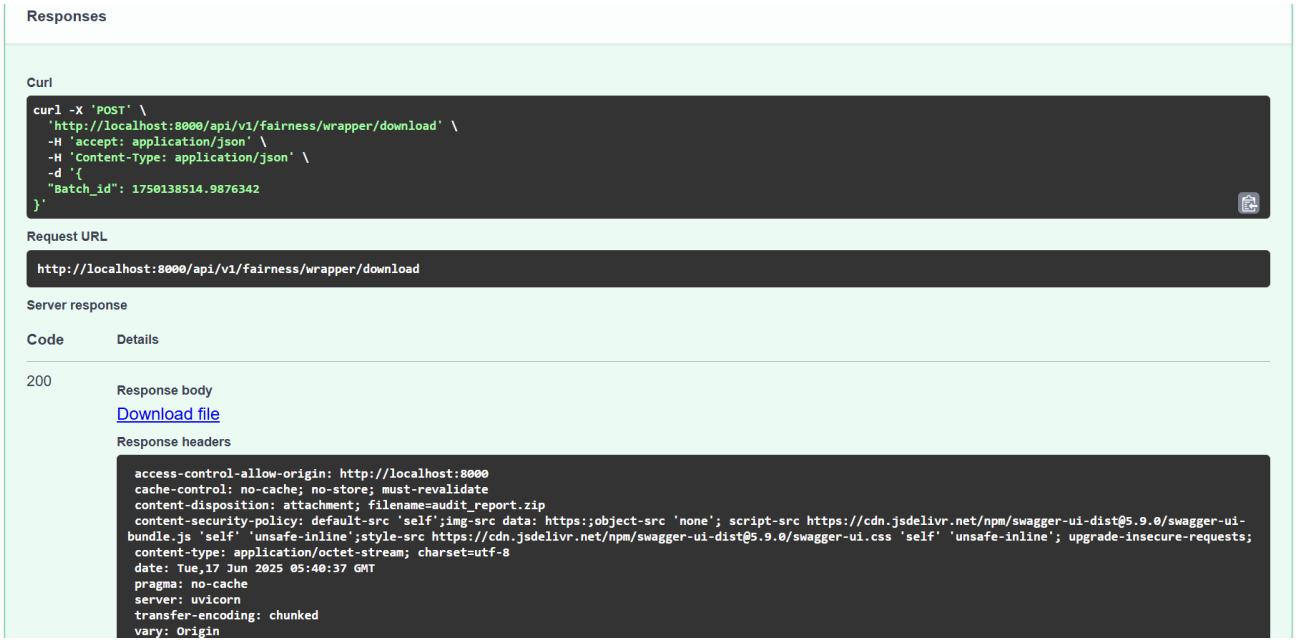
This API will download the report once the operation has completed successfully.

Payload:

1. BatchId: Enter the same BatchId which is used in **/fairness/wrapper/batchId** endpoint



Response: Report on corresponding operations.



Note: Required the following services to run the workbench APIs

1. Reporting Tool: For report generation. (Described in Dependencies section).
2. Model detail: For batch id generation. (Described in Dependencies section)
3. File-Storage: For file management, if connected to Cosmos DB, a subscription to Azure Blob Storage is required. However, if MongoDB is used, this service is not necessary.