Infosys Responsible AI Toolkit

Explainability (Traditional ML Models) API

usage Instructions

Contents

Introduction

Artificial Intelligence (AI) offers numerous benefits across various domains, It is reshaping our world and driving innovation. It is necessary for an AI system to provide clear and understandable reasons for its decisions or predictions. Essentially, it means that the system should be able to justify its actions in a way that humans can comprehend. Hence this explainability is crucial for building trust, adoption of AI and ensuring accountability in high-risk AI applications.

Once API swagger page is populated as per instructions given in the github repository Readme file, click on ‘try it out’ to use required endpoints. Details of endpoints associated with Explainability tenet are outlined below.

Dependencies for Explainability

The Explainability APIs depend on both the Model Details repository and the Reporting Tool repository. Please follow the setup instructions in the README files of both repositories to configure them. Ensure that both services are up and running before interacting with the Explainability APIs.

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

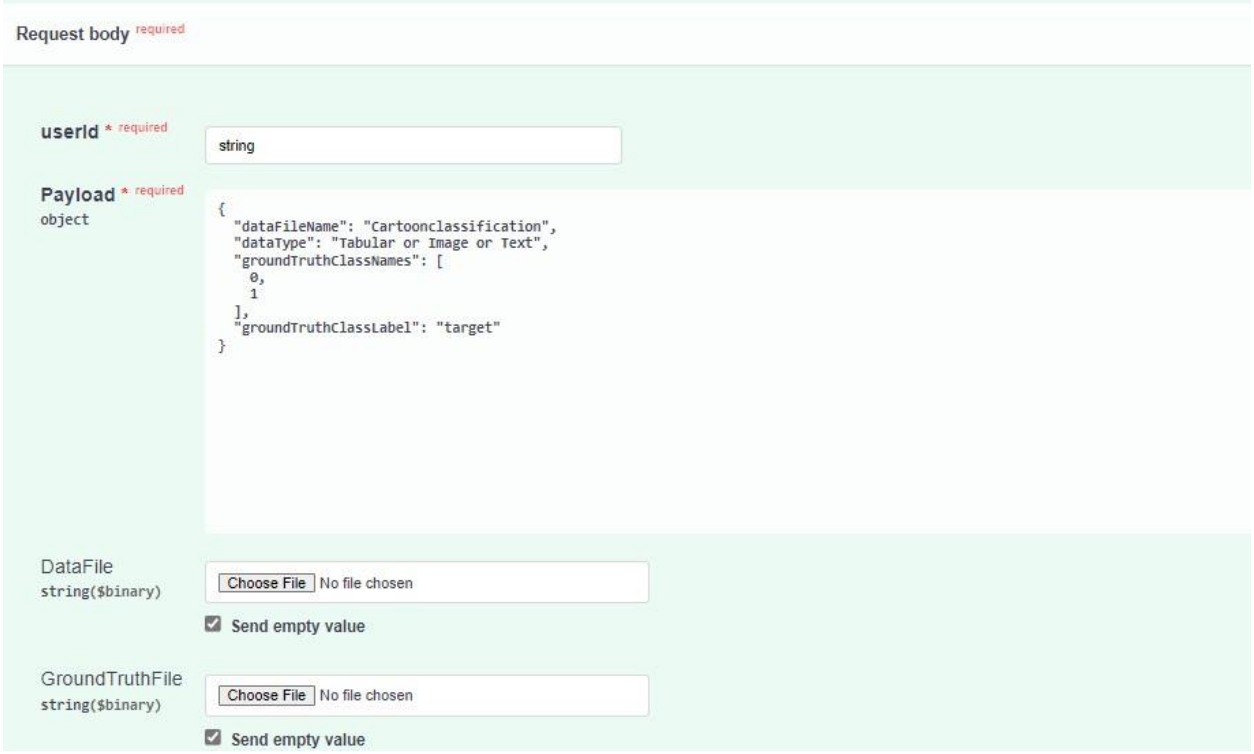
Explainer methods/logic will be embedded with the actual model and data. Therefore the actual model and the data for which prediction and explanation required needs to be uploaded via the following APIs.

3.1: /v1/workbench/adddata

Provided below the details of payloads.

- **userId:** Provide the name of the user.
- **Payload:**
    - **dataFileName:** Enter the name of your data file here.
    - **dataType**: Currently, only Tabular 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."**



3.2: /v1/workbench/data –

This API is used get the 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.

3.3: /v1/workbench/addmodel –

The required payloads need to be filled along with the actual predictive model.

- **userId:** Provide the name of the user
- **Payload:**

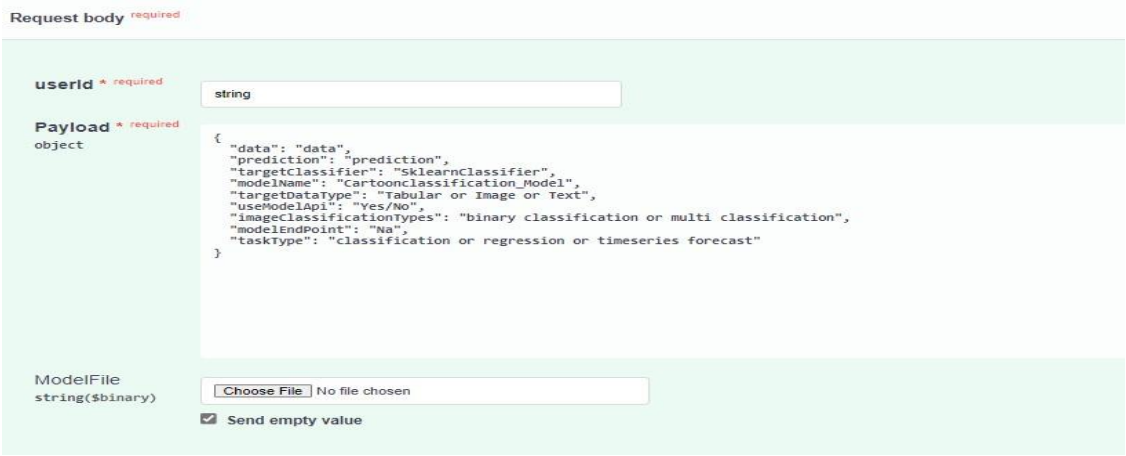
- **modelName:** Enter the name you want to assign to your model.
- **targetDataType:** Currently, only Tabular data is supported. Please specify the data type as **Tabular**.
- **taskType:** Specify the type of task for your model. Use "**CLASSIFICATION**" for classification tasks, "**REGRESSION**" for regression tasks, or "**TIMESERIESFORECAST**" for time series forecasting tasks.
- **targetClassifier:** This field can be removed or ignored.
- **useModelApi:** If you are uploading the model, enter "No." If you are providing the model via an endpoint, enter "Yes." If useModelApi is set to "Yes":
- **modelEndpoint:** If accessing the model via an endpoint, provide the endpoint URL in this field.
- **data:** This field should contain the input parameters for the endpoint that bind input data.
- **prediction:** This field should contain the output parameter of the endpoint that returns the prediction data.
- **ModelFile: Please ignore this field.**

  If useModelApi is set to "No":

- **You can either remove or leave the following fields as they are:**
    - **modelEndpoint**
    - **data**
    - **prediction**
  - **ModelFile:** Select and upload the model file from your device.

After entering all the required data, click "execute."

If the model is successfully saved in the database, you will receive a response stating "Model added successfully."



3.4: /v1/workbench/model–

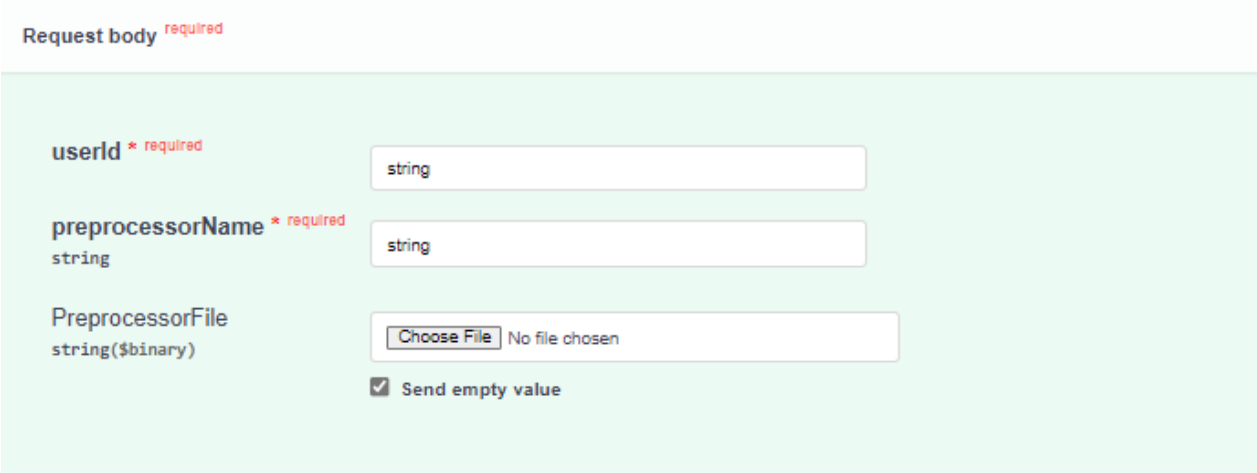
This API is used get the unique id for the uploaded model from the above API. We have to pass the user ID to this API and the API json response will return all the model details that was uploaded by the given user id.

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

3.5: /v1/workbench/addpreprocessor–

The required payloads need to be filled along with the actual preprocessors.

- **userId:** Provide the name of the user**.**
- **preprocessor:** provide the preprocessor name.
- **preprocessorFile:** upload the preprocessor.



After entering all the required data, click "execute."

If the preprocessor is successfully saved in the database, you will receive a response stating "preprocessor added successfully."

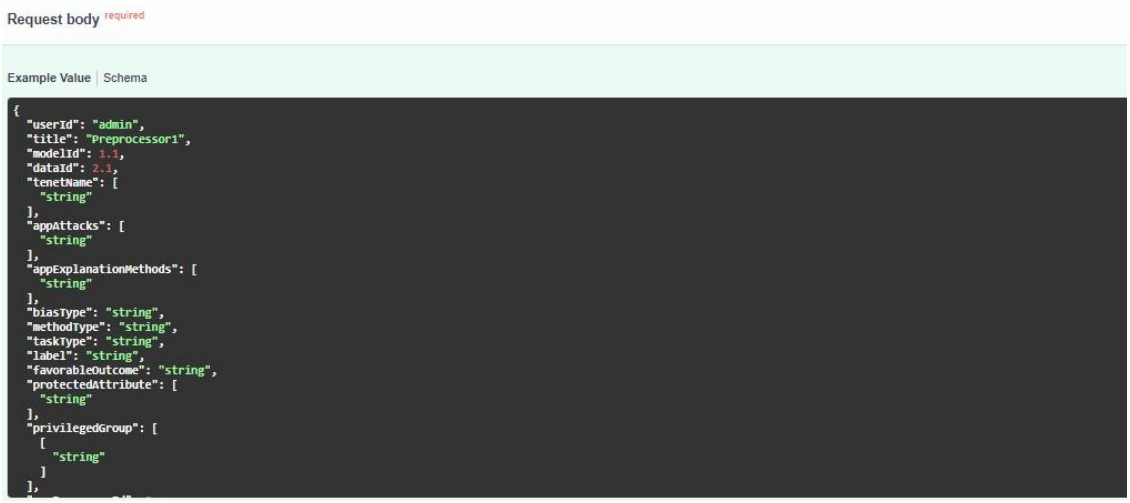
3.6: /v1/workbench/batchgeneration –

Batch generation is the process which combines all the uploaded data & model to generate a batch. In this batch the information of which explainer methods is required to provide explanation also is gathered. Unique batch id will be generated form this API which will be used for further processing to get the explanation report. Payload details need to be filled as shown below.

- **userId:** Provide the name of the user
- **title**: You can provide a suitable title for the task you are performing.
- **modelId**: Get "modelId" from the "/v1/workbench/model " API. (mentioned in 3.4 section).
- **dataId**: Get "dataId" from the "/v1/workbench/data" API. (mentioned in 3.2 section).
- **tenetName**: "**Explainability**" is the tenet name.
- **appExplanationMethods**: Please paste the methods you obtained from explainability repository using "/rai/v1/explainability/methods/get" API (mentioned 5.1 section).
- Leave the other fields unchanged, meaning no modifications or transformations should be applied to them.

Click on "execute" to get BatchId and TenetID.

Executing all the APIs mentioned above from the model details repository are mandatory for using the Explainability APIs.

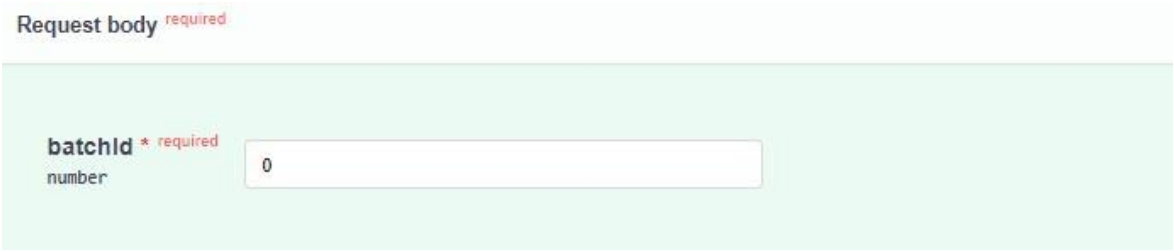


Reporting Tool Repository (responsible-ai-reporting-tool)

This repository is used to generates the required explanation report which will be a zip file containing a pdf report and excel file. PDF report gives detailed explanation of Global Explanation and Local Explanation along with required visualization charts.

4.1: /v1/report/downloadreport:

To download the report, we first need to generate it. The process of report generation will be discussed in the Explainability APIs section (mentioned in 3.3 section below).



- **batchId:** Provide the batchId that was used to generate the report.

We will receive a downloadable file as output, which can be used to download the report.

Explainablity APIs

We have three APIs available in the Explainability repository, each serving a different purpose as detailed below:

5.1: /rai/v1/explainability/methods/get - Returns a list of applicable explainability methods for the given model ID and dataset ID.

Payload

- **modelId**: Get "modelId" from model details repository using "/v1/workbench/model" API (mentioned in 3.4 section).
- **datasetId**: Get "datasetId" from model details repository using "/v1/workbench/data " API (mentioned in 3.2 section).
- **scope**: Please specify "LOCAL" if you want local applicable methods and "GLOBAL" for global applicable methods. If you want all methods irrespective of scope, remove this field or pass it as null.

Returns a list of applicable methods as the response

5.2: /rai/v1/explainability/explanation/get - Provides an explanation using an input instance for the given model ID and dataset ID.

Payload

- **modelId**: Get "modelId" from model details repository using "/v1/workbench/model" API (mentioned in 3.4 section).
- **datasetId**: Get "datasetId" from model details repository using "/v1/workbench/data " API (mentioned in 3.2 section).
- **scope**: Specify 'LOCAL' for a local explanation or provide 'GLOBAL' for a global explanation.
- **method:** Provide one method based on the scope (mentioned in 5.1 section).
- **inputRow:** Provide the input instance as dictionary which you want to use for the explanation.
- Pass the other fields as **null** values.

5.3: /rai/v1/explainability/report/generate - Generates a report for the specified use case by taking the batch ID as input.

- **batchId:** Get batchId from model details repository using "/v1/workbench/batchgeneration" API (mentioned in 3.5 section).

Once report is generated successfully, can download the report (mentioned in 4.1 section).

Endpoints Usage Flow

Endpoint	Description
/rai/v1/explainability/methods/get	Returns list of explanation methods applicable for selected model and dataset
/v1/workbench/adddata, /v1/workbench/data	To upload Data and retrieve dataid and name respectively
/v1/workbench/addmodel, /v1/workbench/model	To upload Modle file and retrieve modelid and name respectively (supports csv files)
/v1/workbench/addpreprocessor, /v1/workbench/preprocessor	To upload encoder file and retrieve encoderid and name respectively (supports pickle files)
/v1/workbench/batchgeneration	Maps data, model and preprocessor to create a pipeline for explainability generation
/rai/v1/explainability/explanation/get	Get explanation for the given model and dataset
/rai/v1/explainability/report/generate	Generates a doc report for the given input of data, model, preprocessor for a specific batchid
/v1/report/downloadreport	To download the report generated against specific batchid