2024

Draft Document

DELIVERABLE 2.1

THIS DOCUMENT IS IN DRAFT FORM AND PENDING OFFICIAL APPROVAL. IT IS SUBJECT TO REVIEW AND MAY BE UPDATED.

D2.1: Pricing and Discovery Module I – Technical Specification

This project has received funding from the European Union's Horizon Research and Innovation Actions under Grant Agreement № 101093216.

Title:	Document version:
D2.1: Pricing and Discovery Module I	1.0

Project number:	Project Acronym	Project Tittle
101093216	UPCAST Project	UPCAST Project

Contractual Delivery Date:	Actual Delivery Date:	Deliverable Type*-Security*:
M15 (March 2024)	M15 (March 2024)	OTHER - PU
Responsible:	**Organization:**	**Contributing WP:**
Aditya Grover	CeADAR Ireland	WP2

Authors (organization):
Hanene Jemoui (CDR)
Aditya Grover (CDR)
Santiago Andres Azcoitia (LST)
Semih Yumusak (SOT)
Shanshan Jiang (SINTEF)

Abstract:

The UPCAST project is a pioneering endeavour that enhances data sharing across diverse sectors providing plugins to increase efficiency and effectiveness of data marketplaces through a unified platform. Leveraging advanced AI and data management technologies, the plugins address complex challenges in digital marketing, healthcare, public administration, and genomics research. The project integrates cutting-edge tools to facilitate seamless data discovery, processing, privacy enforcement, pricing, and environmental impact assessment. UPCAST promotes open science, gender neutrality, and adheres to ethical AI principles. With a comprehensive data governance structure, it optimizes data utilization while ensuring privacy and compliance. By fostering cross-sector collaboration, UPCAST accelerates innovation and empowers decision-making for a data-driven future.

This technical specification document supplements the demonstration of the first version of the pricing and data discovery plugin, which is the main focus of this deliverable. This document delves into the use cases, data flow, API specification and steps required to successfully run the demonstrators, for both the plugins.

Keywords:

Pricing, Static Pricing, Explainable AI, Discovery, Profiling

REVISION HISTORY

Revision:	Date:	Description:	Author (Organization)
V0.1	09.01.2024	Outline and Table of Contents	Aditya Grover (CDR)
V0.2	27.02.2024	Content added to sections	Hanene Jemoui (CDR) Aditya Grover (CDR) Santiago Andres Azcoitia (LST) Semih Yumusak (SOT) Shanshan Jiang (SINTEF)
V0.3	07.03.2024	Draft document sent for peer review	Aditya Grover (CDR)
V0.4	20.03.2024	Peer Review comments integrated	Hanene Jemoui (CDR) Aditya Grover (CDR) Santiago Andres Azcoitia (LST) Semih Yumusak (SOT) Shanshan Jiang (SINTEF)
V1.0	26.03.2024	Final Document Ready for Submission	Aditya Grover (CDR)

This project has received funding from the European Union's Horizon Research and Innovation Actions under Grant Agreement Nº 101093216.

More information available at https://upcastproject.eu/

COPYRIGHT STATEMENT

INDEX

LIST OF FIGURES

LIST OF TABLES

1 Introduction

1.1 Purpose of the Deliverable

This deliverable presents the first version of the Pricing and Resource Discovery plugins (D2.1), which is linked to tasks T2.3 and T2.4 of Work Package 2 in the UPCAST project. It serves as a public record of the progress achieved in these tasks, showcasing the plugin development through practical demonstrations.

Deliverable D1.2 in Work Package 1 defined the UPCAST MVP and its core functionalities. Figure 1 shows the relevant plugins highlighted in this deliverable.

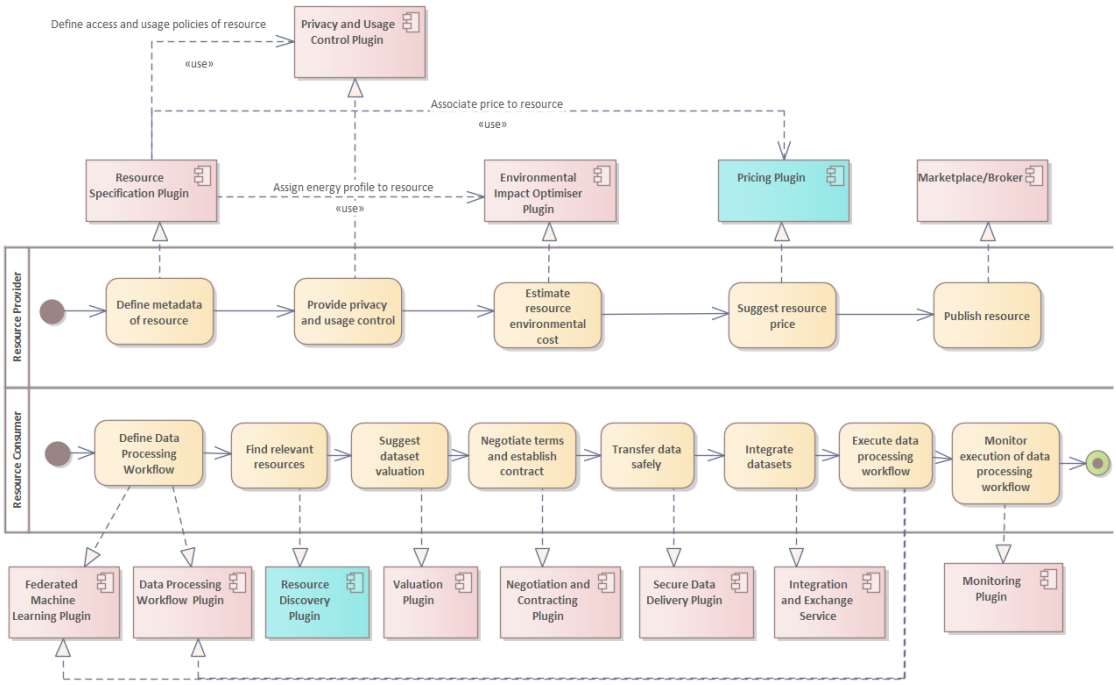

Figure 1: Core functionality included in the UPCAST MVP. Pricing & Discovery Plugins highlighted for this deliverable.

This document is accompanied by video demonstrations showcasing the use of the first version of the plugins that are presented here.

- Pricing: https://youtu.be/DY6l2sH1LT0
- Profiling: https://youtu.be/l8aLyg6QY34
- Discovery: https://youtu.be/_i9NNcqyJ6A

1.2 Scope of the Document

This document serves as the technical specification for D2.1 of the UPCAST project, a public document describing the progress of the development and the practical demonstrations of both the pricing and resource discovery plugins. This document is structured into the following key chapters:

- Chapter 2 describes the pricing plugin, linked to Task T2.3. It details the functionality and practical demonstration of the first version of the plugin focusing on static pricing models and explainable AI.
- Chapter 3 describes the first version of the resource discovery plugin, showcasing the progress carried out in Task T2.4. It includes the explanation and steps on how to run the demonstrator. Furthermore, it introduces the data profiling service linked to this plugin and describes how it can enhance the functionalities associated with data discovery.
- Chapter 4 concludes the document by summarizing the key activities performed in this deliverable and outlines the next steps for tasks T2.3 and T2.4.

2 Pricing Plugin

Pricing data assets is a key challenge for data providers in their go-to-market processes. The objective of the UPCAST pricing plugin is to help them in setting a price for their data, as well as helping potential data buyers know in advance what the reasonable price of the data product they are willing to acquire is. It does so by exposing the following endpoints of a REST API, that correspond to the use cases shown in Figure 2:

- A list of datasets existing in commercial data marketplaces whose description is similar to that submitted by the user as an input. This can be useful for data consumers to detect other potentially useful datasets or whose data can be a reference for setting the price of the product and knowing how much to pay for it.
- An estimation of the price of a dataset, which can be a one-off estimation based on the description and characteristics of the data product provided by the static pricing plugin, or a recurring estimation that also has into account the demand and recent other transactions closed for similar products in the market.
- An explanation of this price prediction including which were the most important features affecting (positively or negatively) the result of a prediction.

This section showcases the first version of the pricing plugin, which suggests prices using static pricing models and provides an explanation of such estimations. Furthermore, it also suggests similar resources.

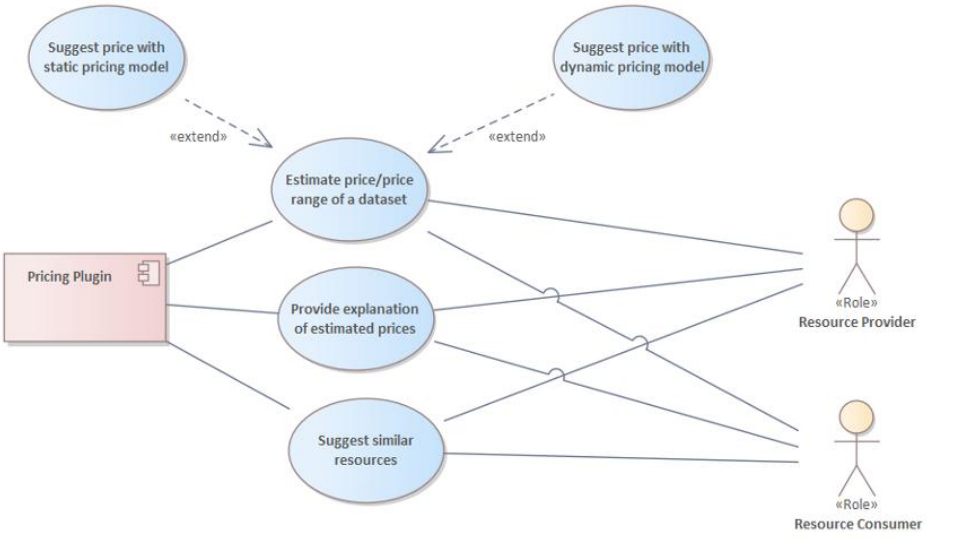

Figure 2: Use cases for the Pricing plugin.

2.1 Static Pricing Models

The static pricing plugin aims to provide consumers and providers with an estimated price of a dataset. Using as inputs textual descriptions and their metadata features, it produces a range of prices for that data asset based on market prices observed in already existing commercial marketplaces, as well as similar datasets existing in the market and a normalized classification of the dataset. For this purpose, the pricing plug-in will rely upon the data about data products and data vendors from different commercial marketplaces. Figure 3 summarizes the architecture of the static pricing.

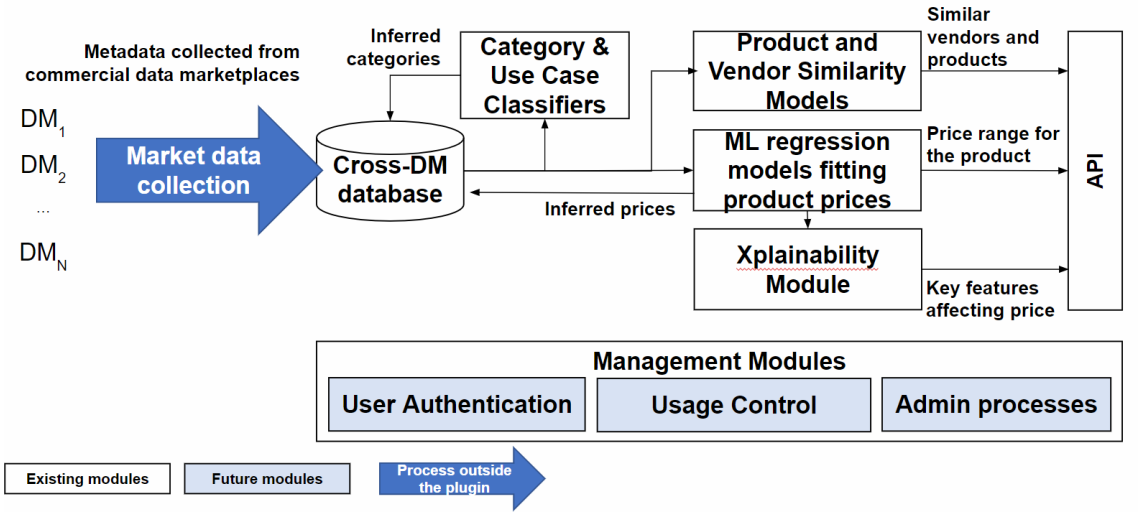

Figure 3: Architecture of UPCAST static pricing tool

There are already a few relevant online data marketplaces. Based on information collected from those public data marketplaces, a cross-DM database has been created containing metadata of those products, their vendors and their prices. A preliminary capture of this information has produced information about 200 thousand data products

from more than 2.100 vendors and contains more than 10 thousand price references[1]. They are used to train Machine Learning data product classifiers and price regressors. Since data markets use different categories and criteria to label data products, classifiers help homogenize data product classification by learning the criteria used to label a dataset as belonging in a certain data category, and then labelling data products that do not carry such information. Price regressors learn the features that data providers and marketplaces are using to price data products, and they can predict what the price of a new data product would be based on its features[2]. An AI explainability module, described in the next section, is also included to provide information of the key features that led to that prediction.

Classifiers and similar data products functionality are based on data product descriptions. Deep Neural Network (DNN) Classifiers have been optimized and trained to fit the categories used by AWS[3], the marketplace containing more products in the training sample: "Financial", "Healthcare", "M&E" (Media & Entertainment), "Telecom", "Gaming", "Automotive", "Manufacturing", "Resources", "Retail", "Public Sector", "Others". The results obtained are summarized in Figure 4, being the F1 accuracy above 0.88 in the case of "Resource" data products.

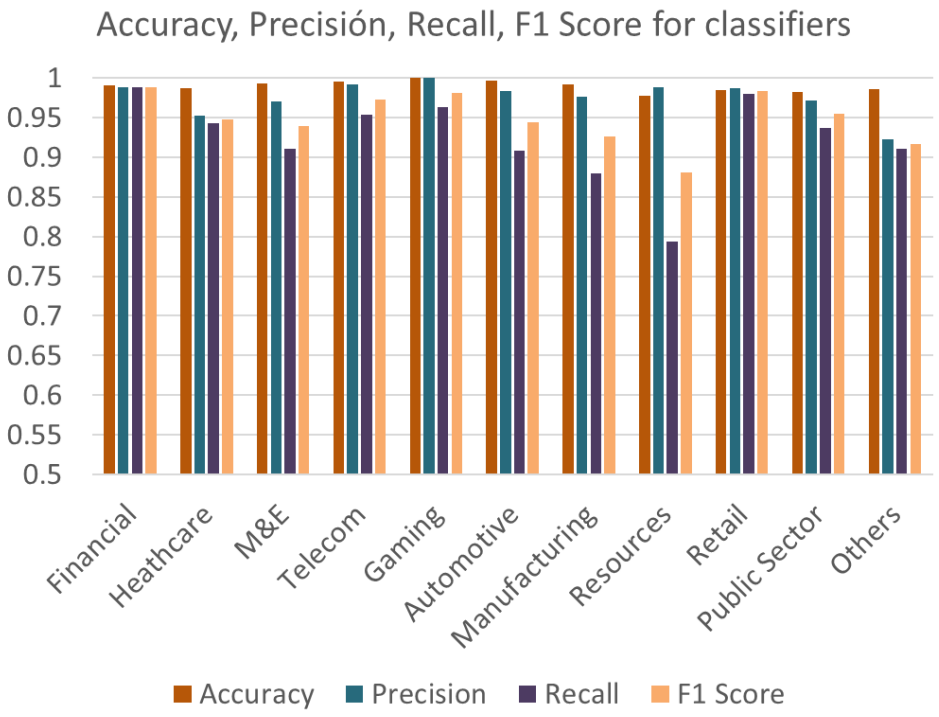

Figure 4: DNN Classifiers Accuracy.

Price regressors rely on a series of features of datasets that were found to be relevant

[1] S. Andres Azcoitia, C. Iordanou, N. Laoutaris, "Understanding the Price of Data in Commercial Data Marketplaces," **International Conference on Data Engineering ICDE'23**.

[2] S. Andres Azcoitia, C. Iordanou, N. Laoutaris, "Measuring the Price of Data in Commercial Data Marketplaces," **ACM Data Economy Workshop**, 2022.

[3] https://aws.amazon.com/marketplace/b/d5a43d97-558f-4be7-8543-cce265fe6d9d?ref_=mp_nav_category_d5a43d97-558f-4be7-8543-cce265fe6d9d&category=d5a43d97-558f-4be7-8543-cce265fe6d9d

when analysing commercial prices, namely:

- Data product description thoroughly describing what kind of data it brings
- Data product category in AWS (either ground truth or inferred)
- Time scope of the data product
- Nº relevant units of data (how many people, companies, records, events, locations, etc.) the data product carries information from
- The potential delivery methods (S3 buckets, bulk download, REST API, data exported from a UI/App, ...)
- The data update rate (e.g., real time, weekly, yearly, etc.)
- Formats in which the information is served (maps, images, csv files, etc.)
- Whether data is produced on demand (e.g., calling an API customizing data)
- Whether the product has any usage limitations (e.g., API / UI with limited users)
- Whether the product includes any kind of professional services (e.g., hours of an analyst)
- Whether the data is granular enough to track session of anonymised users
- Whether the product includes identification data of companies/organizations

The tool uses Random Forests and newly designed DNN models to fit the log of prices observed in the market. DNN models were developed using the TensorFlow and Keras libraries in Python, following all common good practices recommended for such activity by first standardizing the input data. RELU/Leaky RELU activation functions were tested for all hidden layers, and a linear activation function was chosen for the output layer. As loss function mean absolute error (MAE) was used (please note that the model predicts the log of prices). To avoid overfitting, Drop-out between training epochs was randomly applied and to avoid dying/exploding neurons Batch normalization was added between all layers. An Adam optimiser was chosen to optimise the model, using a tuned learning rate decay to train the model faster at the beginning and then decrease the learning rate with further epochs to make training more precise. Callbacks were used to stop the training at the optimal epoch.

Finally, different versions of the model with different number of blocks of layers were tested to see how the accuracy behaved as we reduced the complexity of the model. The version with two layers turned out to be the one with the best accuracy vs. time trade-off. The following figure summarises the configuration of the DNN after this process:

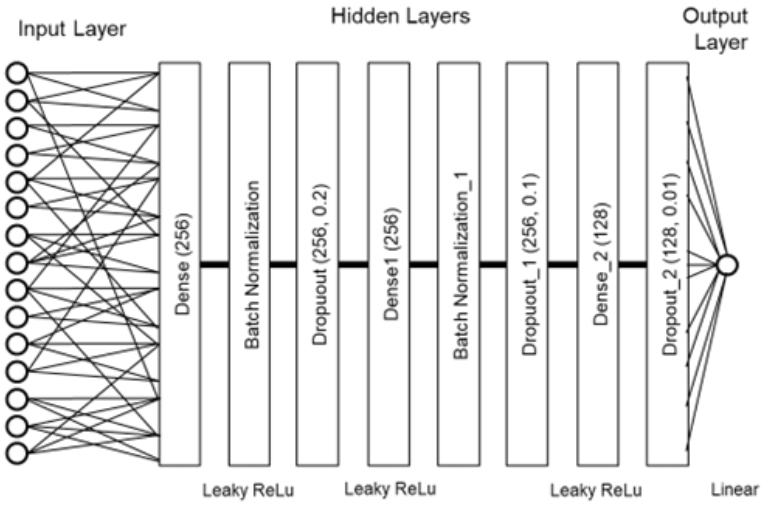

Figure 5: Architecture of DNN Regressors.

The range returned corresponds to the maximum, average and minimum price returned by these 10 regressors trained over 5 different train-test splits of the input data. In terms of performance, the models behave quite well in the sample and can predict the level of prices of data products. Table 1 summarises the performance of the 5 models of each type on the sample

	DNN	**Random Forest**
R^2 Score	0.90-0.92	0.72-0.75
Mean Absolute Error	0.16	0.35
Root mean square error	0.29	0.52
%samples < 5% abs error	60-63%	33-35%

Table 1: Static Model Performance

Mean absolute error and root mean square errors measure the difference between the model prediction and the actual prices of data products in test sets in logarithmic units. The results show that the models are able to perfectly estimate the order of magnitude of the prices of data products in the test set, and that the correlation (given by the R^2 score) is high, especially for the DNN models, which show the highest Pearson correlation (R^2 score) and return predictions close to 5% of the real prices in the majority of cases (60-63$).

2.2 Explainable AI Module

Explainable AI (XAI) refers to the task of making the complex decision-making processes of machine learning models understandable to humans. XAI is becoming very important in most data and AI-based applications due to the following key reasons:

- **Trust:** Users are more likely to trust and use models if there is an understanding why an AI or machine learning model made certain predictions.
- **Debugging & Improvement:** Understanding how a model works helps in finding any weaknesses or biases, and improvements can be made accordingly.
- **AI Regulation:** Use of AI across most domains or verticals now requires transparency and explanations for AI-powered decisions, based on regulatory and legal requirements to be enforced across the EU (e.g., the AI Act that imposes obligations on deployers of high-risk AI systems. such as maintaining operational logs to increase transparency, carrying out self-assessment, or passing model evaluations to ensure trustworthiness in their use, among others).

Figure 5 shows an overview of the XAI process within the static pricing plugin. The input metadata is fed into the machine learning models and explainability methods are applied to these models, which, along with the predictions obtained from the models, include also the explanations, which are then sent as a response to the requestor via the API being developed for this plugin.

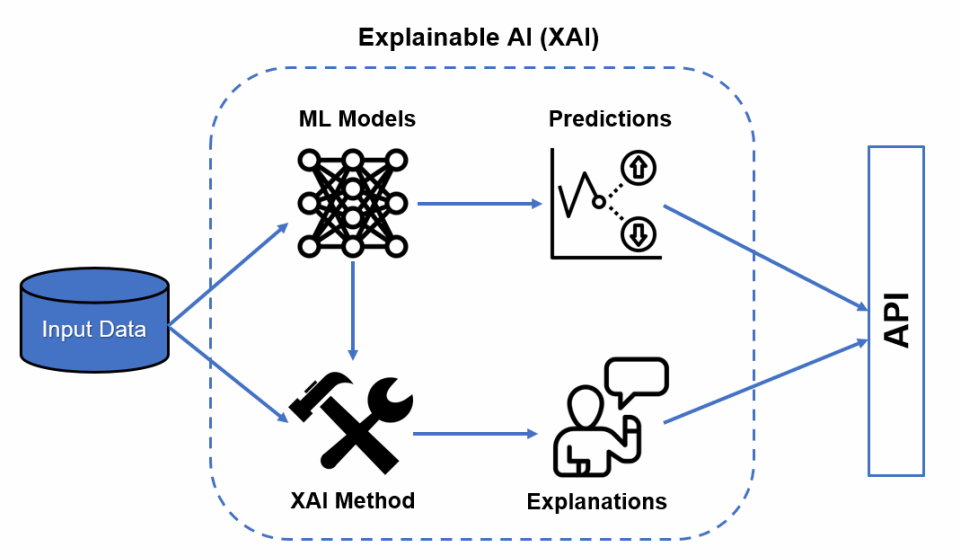

Figure 6. XAI Process Overview

There are several techniques that can be used to explain the decision-making of AI models. One of the main techniques is SHAP[4]. **SHapley Additive exPlanations** (SHAP) is an XAI method based on game theory. In the context of machine learning, a SHAP value represents the **average marginal contribution** of a feature to a model's prediction. Following is a quick explanation:

- **Average:** considers the contribution across all possible combinations of features, not just in isolation.
- **Marginal:** focuses on the change in the prediction when that specific feature is included compared to when it's absent.
- **Contribution:** This change can be positive (increased prediction of price) or negative (that can explain the decreased prediction in price), with the magnitude indicating the strength of the effect.

Essentially, SHAP values indicate how much each feature individually and collectively influences the model's final prediction. In the context of pricing, this allows users to understand how each metadata feature contributes to the price that has been predicted by a particular machine learning pricing model.

2.3 Pricing Plugin Demonstration

This section demonstrates the use of the static pricing plugin showcasing the API and the endpoints that have been developed. The video included with this deliverable explains how to interact with the Swagger API interface with an example and visual interpretation of the results. The OpenAPI specification is available on the project GitHub and can be accessed at: https://github.com/EU-UPCAST/OpenAPISpecification.

The static pricing API is a RESTful API. It uses HTTP response code and verbs. It accepts JSON-encoded bodies, and it returns JSON-encoded responses. The use of the OpenAPI specification makes it compatible with IDS standards.

Different options are available to use the API. We can utilize:

- Any API testing program: Postman, Insomnia, curl...
- Any programming language: Python, JS …

[4] https://shap.readthedocs.io/en/latest/

- The Swagger Interface (refer to Annex II).

Here, we will use cURL command line tool. We will demonstrate how to query the static pricing endpoints using cURL.

Predict the price range: To invoke the Pricing endpoint, send a POST request to the following API URL endpoint:

```
curl -X 'POST' \
  'https://a617-62-83-33-72.ngrok-free.app/getStaticPrice' \
  -H 'accept: application/json' \
  -H 'Content-Type: application/json' \
  -d '{
  "$schema": "string",
  "type": "string",
  "properties": {
            "category":  ["Financial", "Resources"],
            "History": 730,
            "units": [ ],
            "Limitations": 0,
            "ProfServices": 0,
            "IdIndividuals": 0,
            "IdCompanies": 1,
            "GeoScope": [ "Japan" ],
            "delivery": ["S3Bucket", "RESTAPI" ],
            "updatefrequency": ["weekly"],
            "format": [ "csv", "xls" ],
            "ondemand": 0,
            "realtime": 0,
            "locationdata": 0,
            "description": "Japan Imports of Naphtha. This listing contains a
dataset covering waterborne imports of naphtha into Japan over a period of 2 years.
This data is updated on a weekly basis."
        }

}'
```

Result: The result is the price range obtained by the different price regressors, as follows:

```
{
  "minimum": 310,
  "average": 415.2,
  "maximum": 569
}
```

The real price of this dataset in the market is 420 USD/month, which is within the predicted range.

Understand the price range: To invoke the Explanation endpoint, send a POST request to the following API URL endpoint:

```
curl -X 'POST' 'http://localhost:9000/estimates/getExplanation' -H 'accept:
application/json' -H 'Content-Type: application/json' -d '{

"description": "Japan Imports of Naphtha. This listing contains a dataset covering
waterborne imports of naphtha into Japan over a period of 2 years. This data is
updated on a weekly basis.",

"category": [
```

```
"Financial", "Resources"
],
"history": 730,
"limitations": 0,
"profServices": 0,
"idIndividuals": 0,
"idCompanies": 1,
"geoScope": [
"Japan"
],
"delivery": [
"S3Bucket", "RESTAPI"
],
"updatefrequency": ["weekly"],
"format": [
"csv", "xls"
],
"ondemand": 0,
"realtime": 0,
"locationdata": 0 ,
"units":[]
}'
```

Result: The result is the following in JSON format:

```
{
  "estimatedPriceRange": {
    "minimum": 310,
    "average": 415.2,
    "maximum": 569
  },
  "currency": "USD/Month",
  "models": {
    "model_predicted_minimum_price": "DNN_Ex_1",
    "model_predicted_maximum_price": "RF_Log_5"
  },
```

```
  "explanations": {
    "min_price": {
      "Categories": [
        {
          "Resources": -0.0875913080867529
        },
        {
          "Others": -0.10512363305451557
        }
      ],
      "Frequency": [
        {
          "quarterly": 0.09301252052512639
        }
      ],
      "Format": [
        {
          "images": -0.09785777449414144
        }
      ],
      "Units": [
        {
          "people": -0.15526863526915713
        },
        {
          "locations": 0.2662310369404526
        }
      ],
      "Description": {
        "red_txt": [
          "listing",
          "japan period 2",
          "japan"
        ],
        "red_txt_sv": -0.17269864739976343,
        "blue_txt": [],
```

```
        "blue_txt_sv": 0
      }
    },
    "max_price": {
      "Units": [
        {
          "entities": -0.012701212063786129
        },
        {
          "people": -0.033946736827222976
        },
        {
          "units": -0.08197821385636042
        }
      ],
      "DeliveryMethod": [
        {
          "S3Bucket": -0.021727289905975437
        },
        {
          "Download": 0.024439018289268172
        }
      ],
      "Format": [
        {
          "csv": 0.029223015238989323
        }
      ],
      "Description": {
        "red_txt": [
          "imports",
          "data updated weekly",
          "contains",
          "covering",
          "japan imports",
          "japan period 2"
```

```
      ],
      "red_txt_sv": -0.07028826201334869,
      "blue_txt": [],
      "blue_txt_sv": 0
    }
  },
  "avg_price": {
    "Categories": [
      {
        "Others": -0.04335710774548157
      }
    ],
    "Frequency": [
      {
        "quarterly": 0.05184159010503686
      }
    ],
    "Format": [
      {
        "csv": 0.05303007539521644
      },
      {
        "images": -0.05601048008143864
      }
    ],
    "Units": [
      {
        "entities": -0.05413771263322738
      },
      {
        "people": -0.10514111633873932
      },
      {
        "locations": 0.15063545692919797
      }
    ],
```

```
      "DeliveryMethod": [
        {
          "StreamingAPI": 0.10327726748345087
        }
      ],
      "Description": {
        "red_txt": [],
        "red_txt_sv": 0,
        "blue_txt": [],
        "blue_txt_sv": 0
      }
    }
  }
}
```


The JSON response object contains the price range, the model that predicted the maximum price and the model that predicted the minimum price. It contains also 3 explanation sections (min, max and average prices explanations).

The explanation sections follow the same structure. For every price, the most important features are followed by their SHAP values. For the dataset description, the only text field, the most important text is extracted and separated into red and blue text based on its SHAP value. The blue text refers to those with a positive SHAP value and the red text corresponds to negative values. In the demonstration video, we explain the results in a graphical manner explaining the SHAP values obtained for an example.

Get similar datasets: To invoke the Similar dataset endpoint, send a POST request to the following API URL endpoint:

```
curl -X 'POST' 'https://a617-62-83-33-72.ngrok-free.app/getSimilarProducts' -H
'accept: application/json' -H 'Content-Type: application/json' -d '{
"type": "string",
"$string": "",
"properties": {
"description": "Stock market quotations for companies in the S&P 500 index",
"nproducts": 3 } }'
```

Result: The JSON result includes a list of the closest nproducts data products in the database to the description provided by the user, which in this case is as follows:


```
[ {
    "DM": "DataRade",
    "Title": "Yacodata: S&P 500 Companies Data (up-to-date intelligence on US
largest 500 companies)",
    "Description": "The dataset consists of companies listed in the S&P500, stock
```

```
market index that measures the stock performance of 500 large companies listed on
stock exchanges in the United State._x000D_\n_x000D_\nThe S&P 500 stock market
index, maintained by S&P Dow Jones Indices, comprises 505 common stocks issued by
500 large-cap companies and traded on American stock exchanges (including the 30
companies that compose the Dow Jones Industrial Average)_x000D_\n_x000D_\nThe
S&P500 or SPX is the most commonly followed equity index, it covers about 80
percent of the American equity market by capitalization._x000D_\n_x000D_\nThe index
constituents and the constituent weights are updated regularly using rules
published by S&P Dow Jones Indices. Although called the S&P 500, the index contains
505 stocks",
    "Provider": "Yacodata"
  },  {
    "DM": "Snowflake",
    "Title": "SEC Reporting Analytics - US Equities Stock Quotes",
    "Description": "Stock price history for analysis & correlation to SEC Reporting
Analytics fundamental financial data set.",
    "Provider": "SEC Reporting Analytics"
  },
  { "DM": "DataRade",
    "Title": "Sentifi Companies Intelligence Analytics on 50k Global Stocks",
    "Description": "Companies intelligence analytics provide a granular view on the
significance of market-moving events as they occur (e.g. does the event make the
stock an outlier in the industry or sector? is the sentiment from the broader
moving shifting unusually ahead of an earnings announcement, are specific
influencer groups shifting their sentiment towards a stock with the potential to
impact the asset valuation?)._x000D_\n_x000D_\nWith Companies Intelligence
Analytics the investor/ analyst/ risk manager can:_x000D_\n_x000D_\n- surface
market-moving events reported by qualified influencers with the potential to impact
asset prices_x000D_\n- evaluate the historical significance of the event (5 years
of history available) and whether an event makes the stock an outlier in the sector
or industry_x000D_\n- evaluate whether unusual sentiment shifts are signals of
upcoming price movements (e.g. unusual pre-earning sentiment shifts).",
    "Provider": "Sentifi"  } ]
```


Classify a dataset: To invoke the Classification endpoint, send a POST request to the following API URL endpoint:

```
curl -X 'POST'
  'https://a617-62-83-33-72.ngrok-free.app/getClassification'
  -H 'accept: application/json'
  -H 'Content-Type: application/json'
  -d '{
  "$schema": "string",
  "type": "string",
  "properties": {
    "description": "Consumer sentiment data to predict sales and willingness to
pay"
  }
}'
```

Result: This data product has been classified as belonging in the *Financial* and *Retail* categories of AWS, as follows:


```
{"result": [
    "Financial: 1",
    "Heathcare: 0",
    "M&E: 0",
    "Telecom: 0",
    "Gaming: 0",
    "Automotive: 0",
```

```
    "Manufacturing: 0",
    "Resources: 0",
    "Retail: 1",
    "Public Sector: 0",
    "Others: 0"
  ] }
```

3 Resource Discovery Plugin

3.1 Data Discovery Plugin Demonstration

This demonstration provides information on how to use and integrate with the Discovery plugin. As described in Figure 6, the Discovery plugin primarily covers search, browse, and discover related/recommended resources functionalities. The demonstration of the plugin summarizes the background technologies and presents a demonstration starting from how to run the plugin services. Later in this section, examples about how to call the plugin API are provided, and the example reference UI implementations are explained.

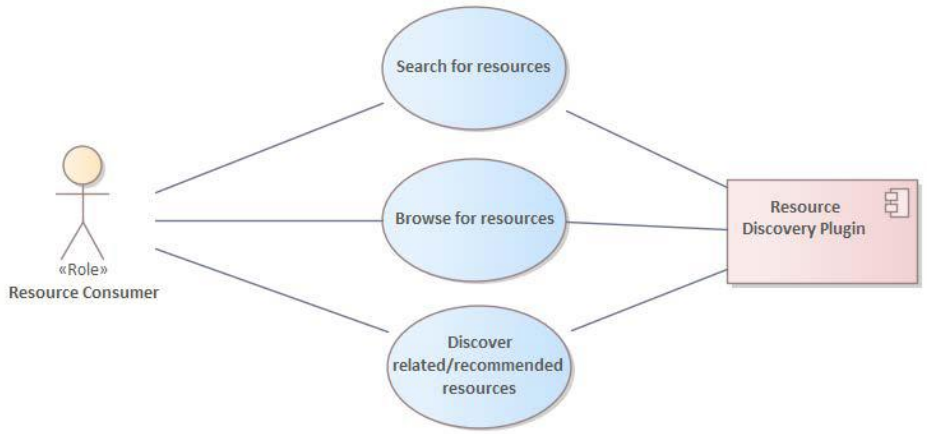

Figure 7: UPCAST Discovery Plugin use-case.

The Discovery plugin is a virtualized set of services including a web service API for integration. The services are configured in a docker compose file which contains the following services: API Service, SOLR Engine, PostgreSQL, and a CKAN Backend Service. Discovery plugin services can be run via docker commands, which are listed in the plugin repository[5]. The Discovery plugin API consists of multiple endpoints to fulfill the complete lifecycle of adding/updating/deleting datasets to the index, search within the index, and discover similar resources (See ANNEX II). In order to run these, we describe below the technical steps to run those API endpoints with a sample UI.

There are two types of discovery that are supported by the plugin: (1) dataset search, and (2) resource search. Dataset search can contain multiple parameters to make a Solr [6]query, filtering, sorting, and faceted results. In order to make a dataset searchable, it needs to be added to the index. Below is a sample cURL[7] request to create a dataset within the index:

```
curl --location 'http://localhost:8000/catalog/create_dataset/' \
--form 'package_name="sample"' \
--form 'package_title="Sample"' \
```

[5] https://github.com/EU-UPCAST/discovery-plugin
[6] https://solr.apache.org/guide/6_6/searching.html
[7] https://curl.se/docs/tutorial.html

```
--form 'organization_name="upcast"' \
--form 'package_notes="Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed consequat odio eget sem congue, eget condimentum risus volutpat. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Pellentesque sed ante id nunc ultrices viverra.'"
```

Additionally, updates and deletions of a dataset, as well as data uploads can be accomplished by using similar requests to the endpoints defined in ANNEX II.

The search functionality requires more detailed form parameters to be supplied, such as Solr query, facet options, and filters.

Below is a sample cURL request to run the dataset search API endpoint:

```
curl --location --request GET 'http://localhost:8000/discover/dataset_search?q=name%3Aweather*' \
--header 'Content-Type: application/json' \
--data '{
    "fq": "*",
    "sort": "metadata_modified desc",
    "rows": 10,
    "start": 0,
    "facet": true,
    "facet_mincount": 1,
    "facet_limit": 50,
    "facet_field": ["organization", "format"],
    "include_drafts": false,
    "include_private": false,
    "use_default_schema": true
}'
```

Below is a sample cURL request to run the research search API endpoint:

```
curl --location 'http://localhost:8000/discover/resource_search?query=name%3Atest&order_by=created'
```

The search API is presented in a simple user interface, which is a part of the discovery plugin repository as shown in Figure 7.

In addition to the search functionalities, there is a function for discovering and searching for similar datasets. The endpoint accepts a single input parameter as the dataset_id and searches for similar datasets.

```
curl --location --request POST
'http://localhost:8000/discover/discover_similar_datasets?dataset_id=city-1'
```

The discover similar datasets API is presented in a simple user interface, which is a part of the discovery plugin repository in Figure 8.

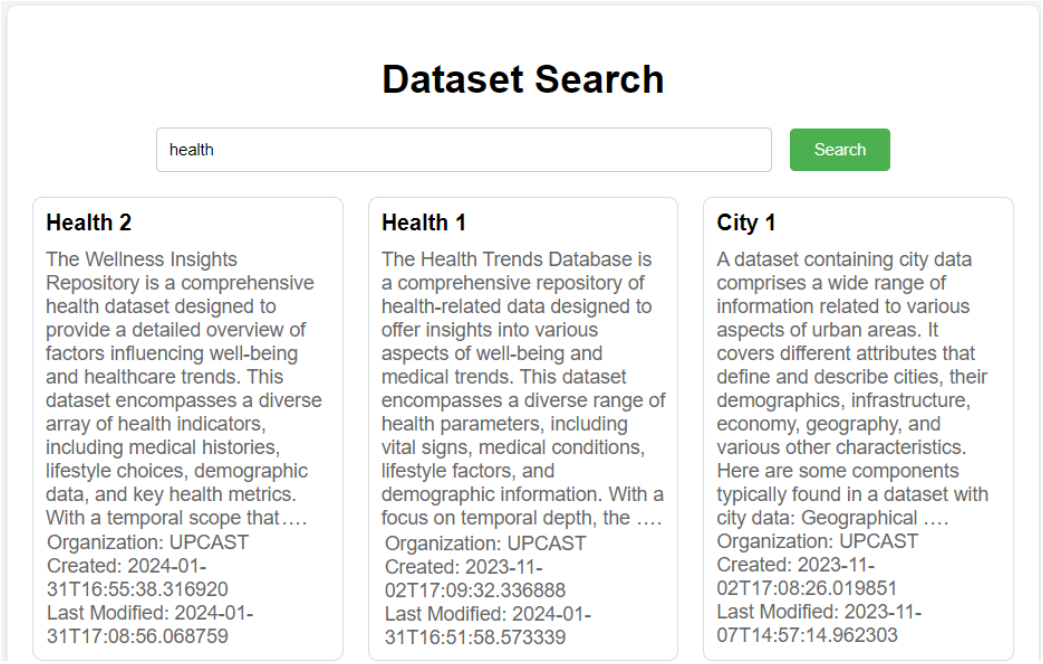

Figure 8: Sample Search UI using the Discovery Plugin API

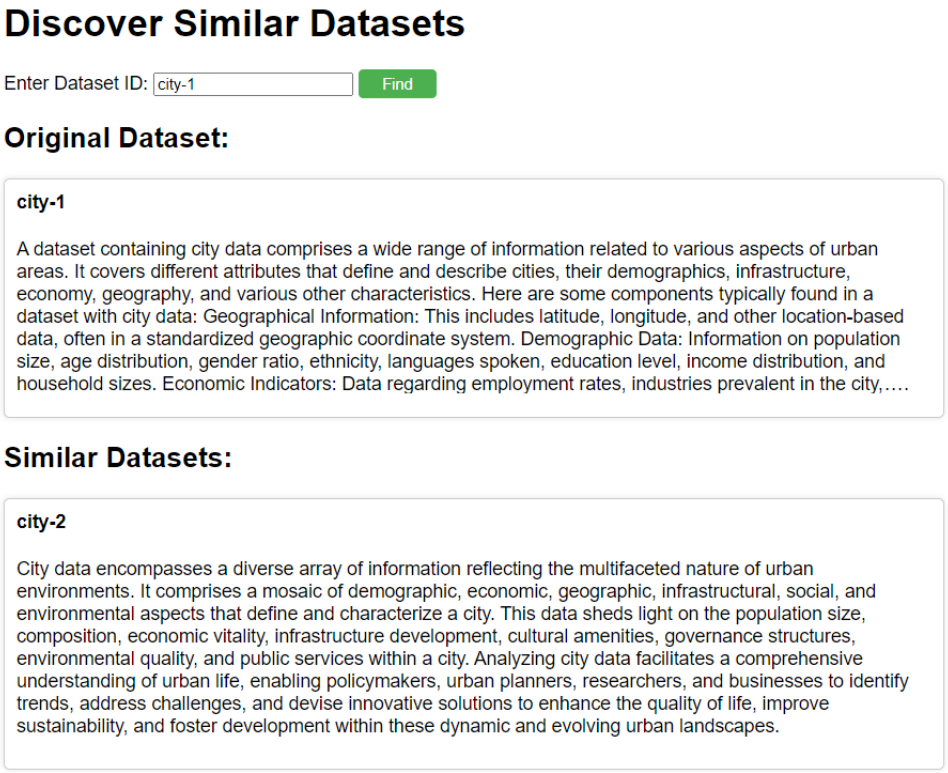

Figure 9: Sample discovery UI using the Discovery Plugin API

3.2 Data Profiling Demonstration

Data profiling is used to extract metadata and insights from the data. It can be used to find data representation for data discovery, machine learning tasks or other purposes. The data representation can be an explicit representation (with useful and understandable insights such as statistics about data) or an implicit representation (like embeddings or vectors). In UPCAST, our focus is to find the right mix of data representation methods to improve data discovery.

Both Resource Providers and Resource Consumers can make use of profiles generated by profilers (see Figure 9). Resource Provider can use the Resource Specification Plugin to generate resource profile as part of the "Define metadata of resource" task. When Resource Consumer uses the Resource Discovery Plugin to find relevant resources, the Resource Discovery Plugin can discover or recommend resources based on profiles. The UPCAST profiling service is an add-on to the Resource Discovery plugin.

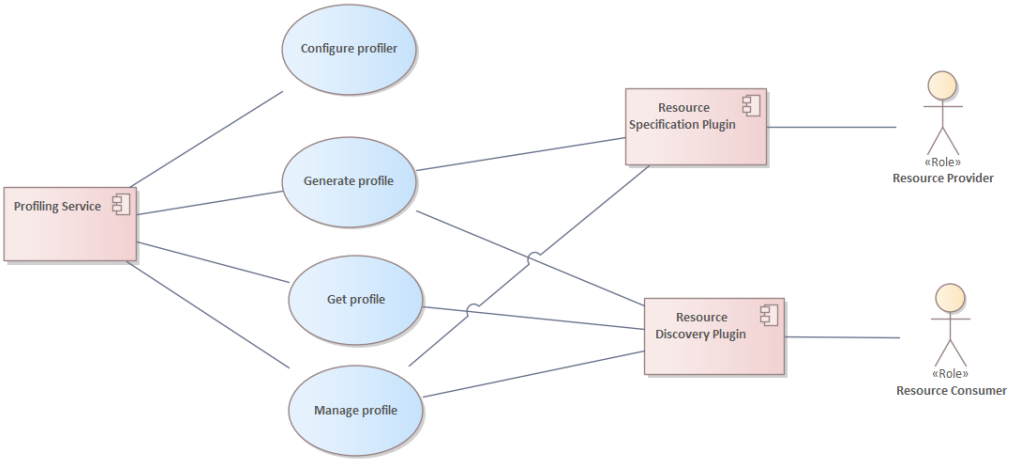

Figure 10: Use case diagram for the profiling service.

As there are different profilers with various functionality, a Profiling Service can be connected to a number of profilers and provide "plug-and-play" profiling service that can generate profiles using selected profilers based on the needs and requirements of the user.

The Profiling Service can be implemented as a standalone service or be integrated as an add-on to the Resource Discovery Plugin. Since the data profiling service needs access to the datasets to be profiled, consent to access the data should be obtained from the data provider. Two deployment scenarios can be considered (Figure 10): 1) For big dataset or dataset with privacy concern: the profiler is installed locally with the data provider, 2) For small dataset: the dataset can be sent to the (independent) profiler service to generate a profile.

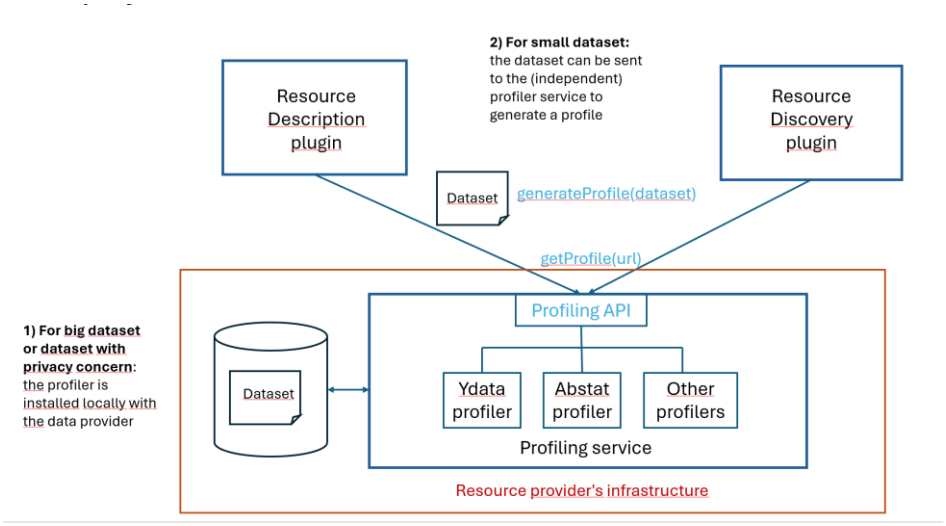

Figure 11: Deployment scenarios.

The current OpenAPI specification is shown in Figure 16 in ANNEX II. The detailed OpenAPI specification is available on the GitHub: https://github.com/EU-UPCAST/OpenAPISpecification/tree/main/profiling

An App has been created to demonstrate the creation of a profile for a local dataset (Figure 11). The user (resource provider) can select a profiler to be used from a list of available profilers (Figure 12, which uses the API endpoint: "GET /profile/get_profilers"), upload a local file for profiling through browsing files or by "drag and drop" files (Figure 13), then click the "Generate profile" button (using the API endpoint: "POST /profile/generate_profile"). The profile generated will be returned (Figure 13).

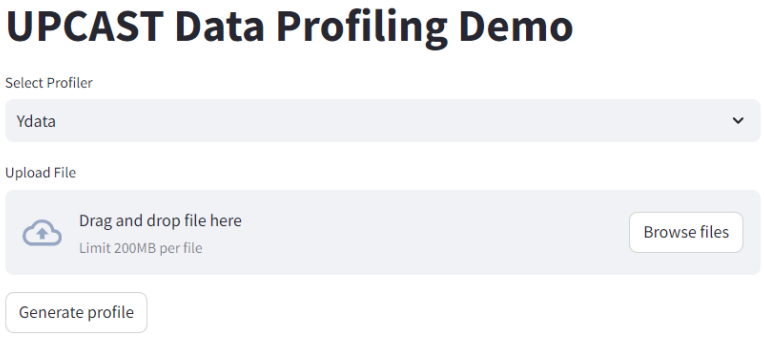

Figure 12: Demo for creation of a profile – start page

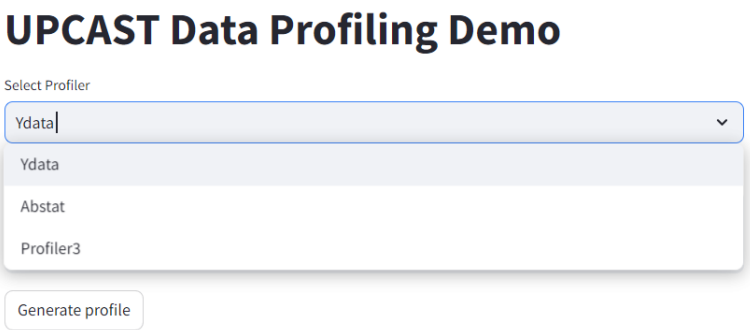

Figure 13: Select a profiler from a list of available profilers

Figure 14: Demo showing the generated profile.

In addition, the API endpoint "GET /profile/get_profile" will return a list of profiles associated with the dataset specified by the URL, optionally for the specified profiler. Below is a sample request for this endpoint:

```
curl -X 'GET' \
  'http://localhost:8000/profile/get_profile?url=http%3A%2F%2Flocalhost%2Fprofiler-demo%2Fsample_csv.csv&profiler_name=Ydata' \
  -H 'accept: application/json'
```

The API endpoint "PUT /profile/update_profile" will update the profile for a resource specified by the URL with a local file. Below is a sample request for this endpoint:

```
curl -X 'PUT' \
  'http://localhost:8000/profile/update_profile?url=C%3A%5CUPCAST%5Cprofiler-demo%5Coutput_files%5Csample_csv_profile_target.json' \
  -H 'accept: application/json' \
  -H 'Content-Type: multipart/form-data' \
  -F 'file_content=@sample_csv_profile.json;type=application/json'
```

The API endpoint "PUT /profile/delete_profile" will delete the resource profile specified by the URL. Below is a sample request for this endpoint:

```
curl -X 'DELETE' \
  'http://localhost:8000/profile/delete_profile?url=C%3A%5CUPCAST%5Cprofiler-demo%5Coutput_files%5Csample_csv_profile_target.json' \
  -H 'accept: application/json'
```

4 Conclusions and Next Steps

This deliverable is the first out of two expected in the project. It describes the technical specification and practical demonstration of the first version of the Pricing and Data Discovery plugins.

These are the next steps that will be carried out towards the development of the final pricing plugin:

- Refinement and improvement of the static pricing models by training and testing on more data products.
- Development of a user interface as a layer on top of the API to simplify interaction of the end user with the plugin.
- The final plugin will incorporate dynamic pricing models taking into account market conditions and data from marketplace(s).
- Integration with the UPCAST platform and showcasing the final plugin in pilot experiments.

As for the resource discovery plugin, here are the next steps:

- Discovery of datasets based on data processing workflow. In order to achieve this, the UPCAST data vocabulary and data processing workflow input will be aligned with the discovery functionalities.
- Integrate data profiling into resource discovery plugin to provide data discovery based on profiles. Alternative profiles provided by different profilers will be considered.

ANNEX I ACRONYMS

Acronyms List	
API	Application Programming Interface
DNN	Deep Neural Network
XAI	Explainable Artificial Intelligence
JSON	JavaScript Object Notation
AWS	Amazon Web Services
DM	Data Marketplace
UI	User Interface
WP	Work Package
IDS	International Data Spaces
GDPR	General Data Protection Regulation

Table 2: Acronyms List

ANNEX II Swagger API Interfaces

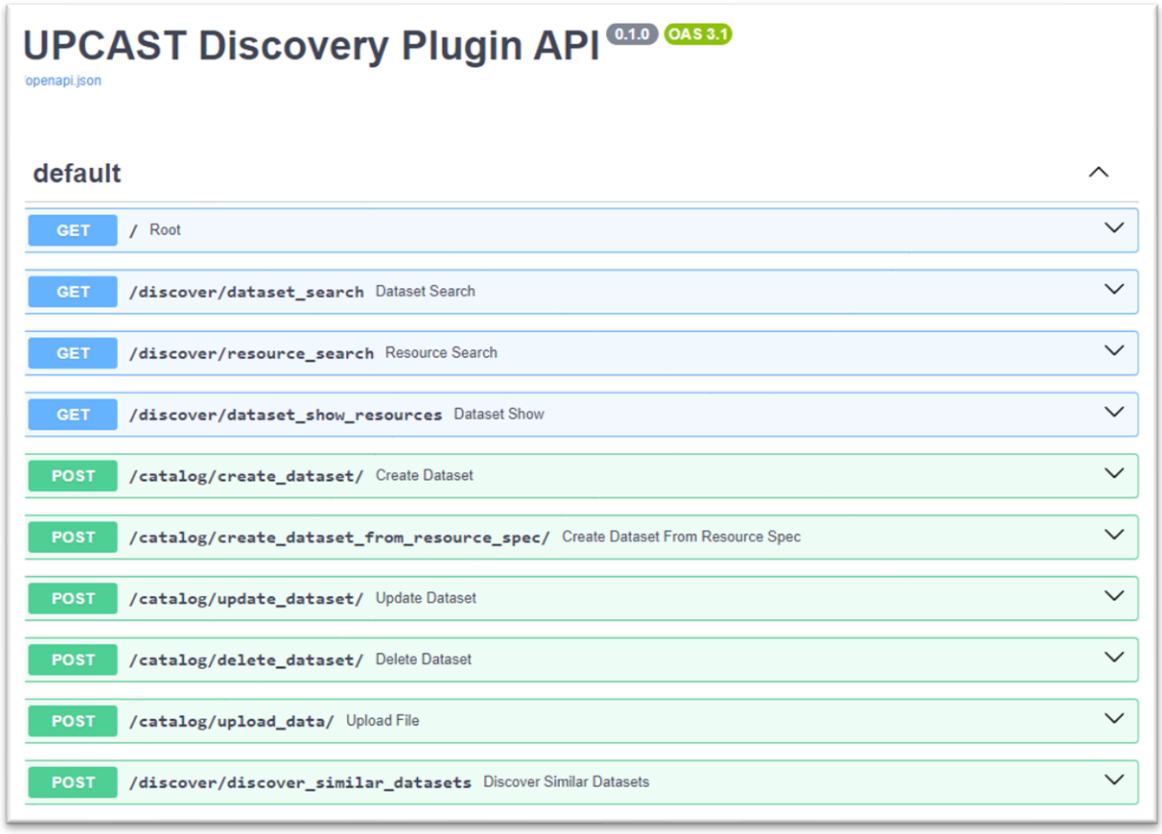

Figure 15: Discovery Plugin API Swagger Interface.

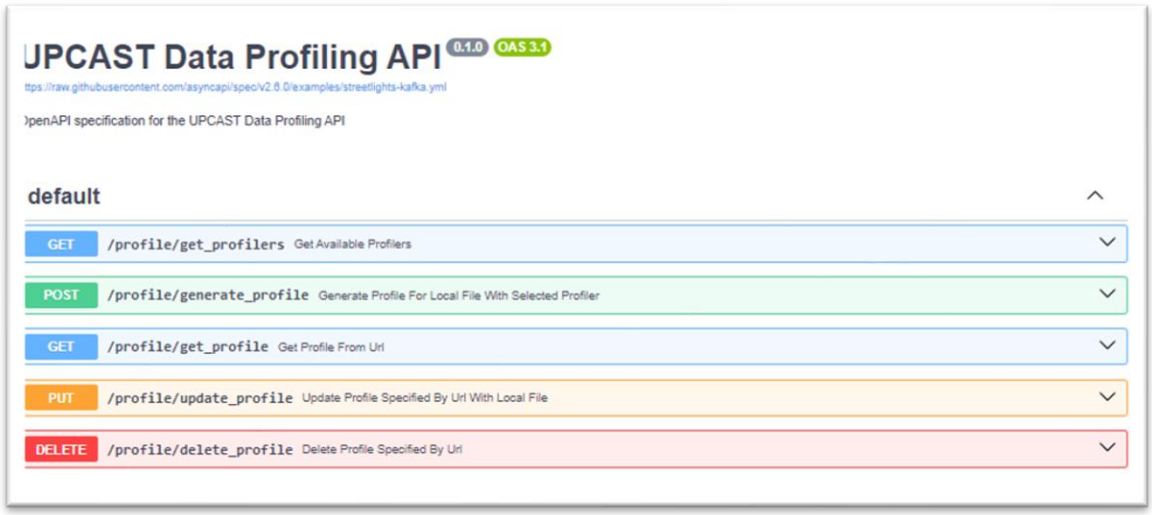

Figure 16: Data Profiling API Swagger Interface.

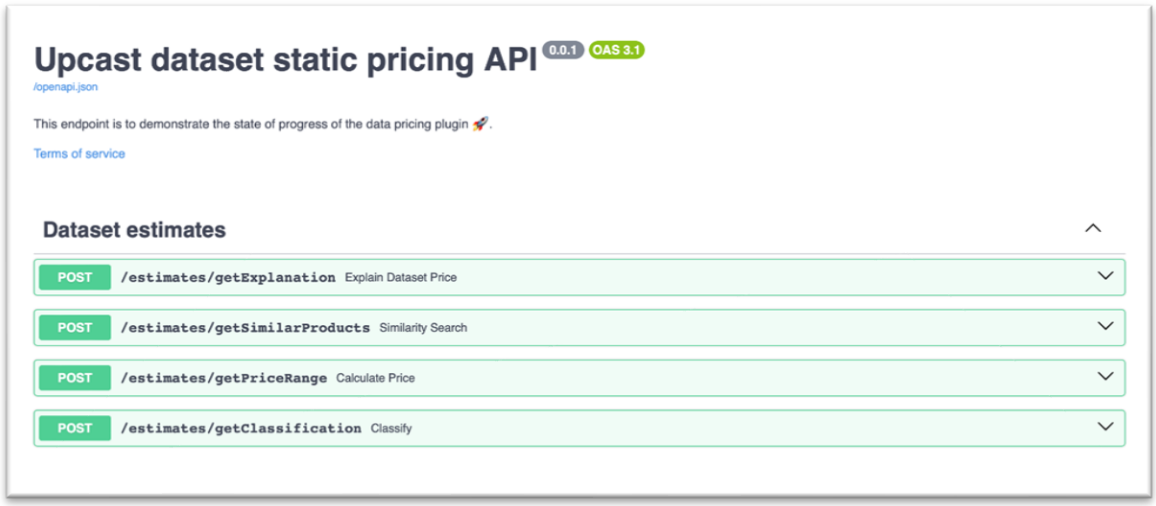

Figure 17: Static Pricing Plugin API Swagger Interface.