



Geo Engine

Geo Engine: Harmonized data access for data analysis pipelines

Dr. Christian Beilschmidt

GWF 2023

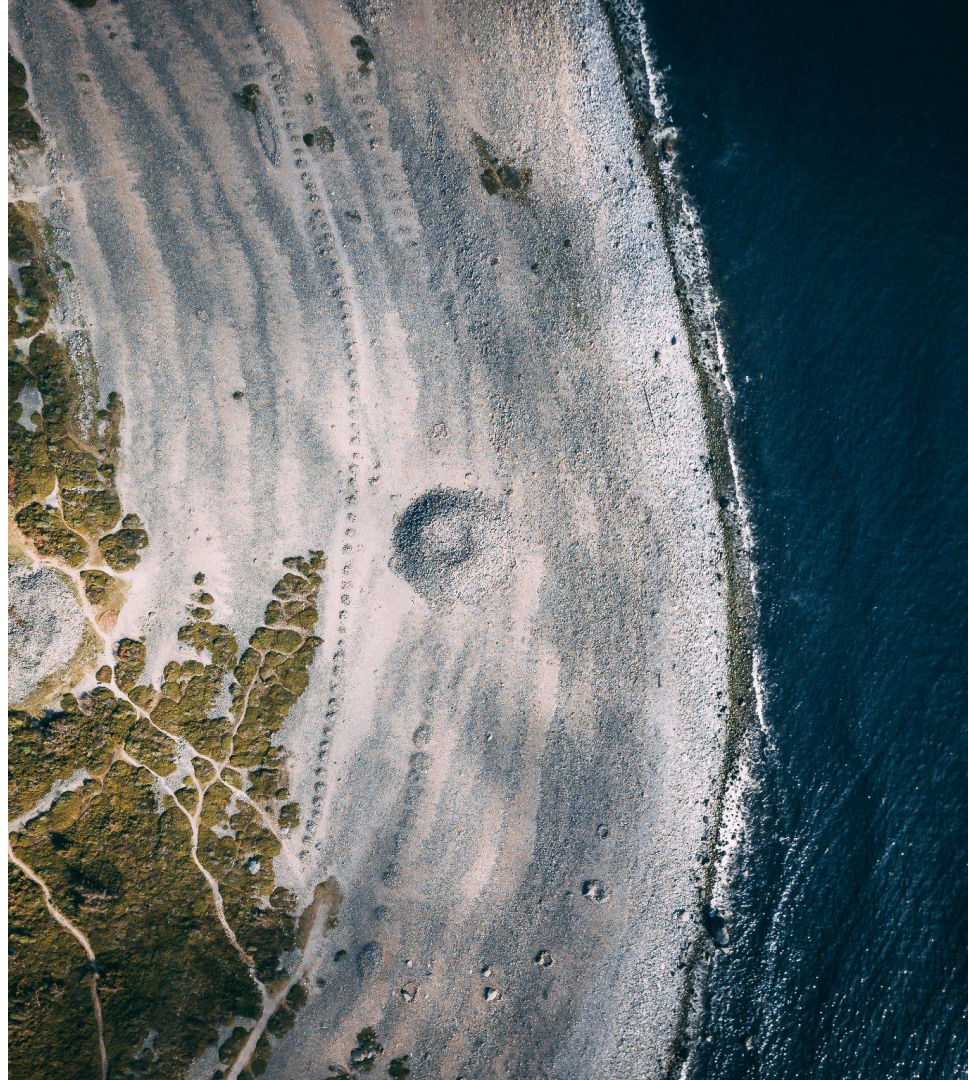
Ein Start-up der



Gefördert durch:



aufgrund eines Beschlusses
des Deutschen Bundestages



About us

- Geo Engine GmbH
est. 2021
- Start-up of University of Marburg,
Germany
- Geo informatics researchers from
biodiversity and remote sensing
projects
- EXIST research transfer

Participation in
NFDI4Biodiversity

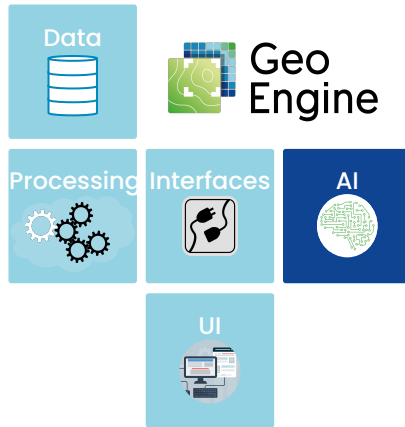
Co-applicant in
CropHype (EnMap)

ML training data pipelines
(Sentinel 2 aggregates)

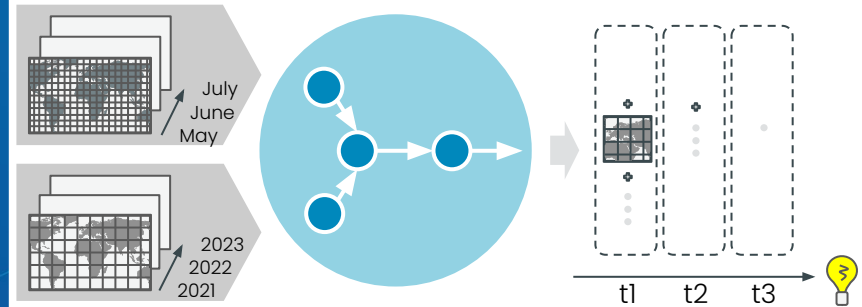
Geo Engine in a nutshell

- Platform for spatio-temporal processing
- Transparent ad-hoc integration of external data via Data Providers
- Process data as time-series instead of files → Transition from static to temporal analytics

Architecture

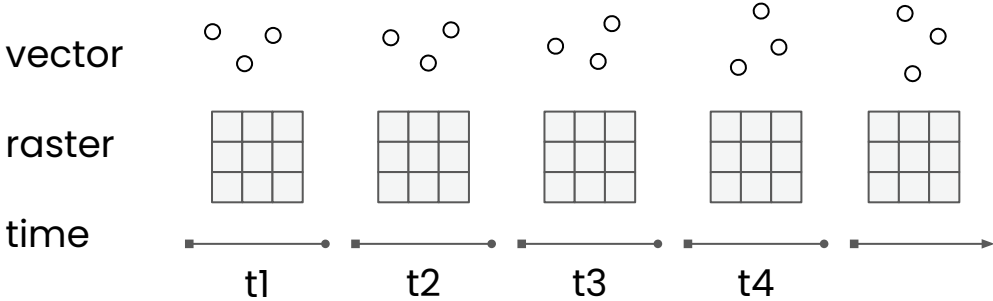


Approach

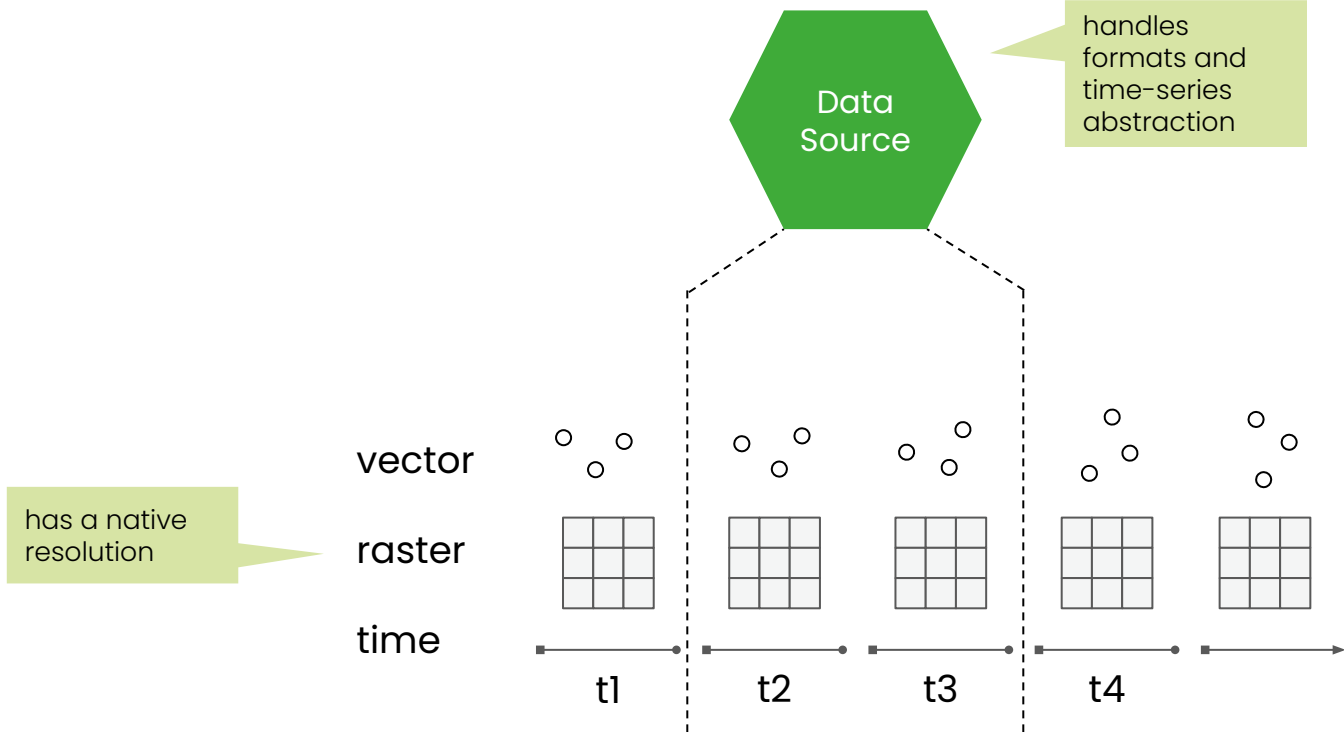


Concept 1: All datasets are time series

has a native resolution



Concept 1: All datasets are time series



Concept 2: Internal and external data




Internal datasets

e.g. point observations

① GeoPackage



② Annotation

Geom	Time	Text	Number
	2023-05-04	This	0.4
	2023-05-04	is	1.3
	2023-05-04	Text	20.4
	...		

③ Integrated time series dataset

External data

e.g. Sentinel-2 data from STAC service

① Sentinel-2 STAC

②

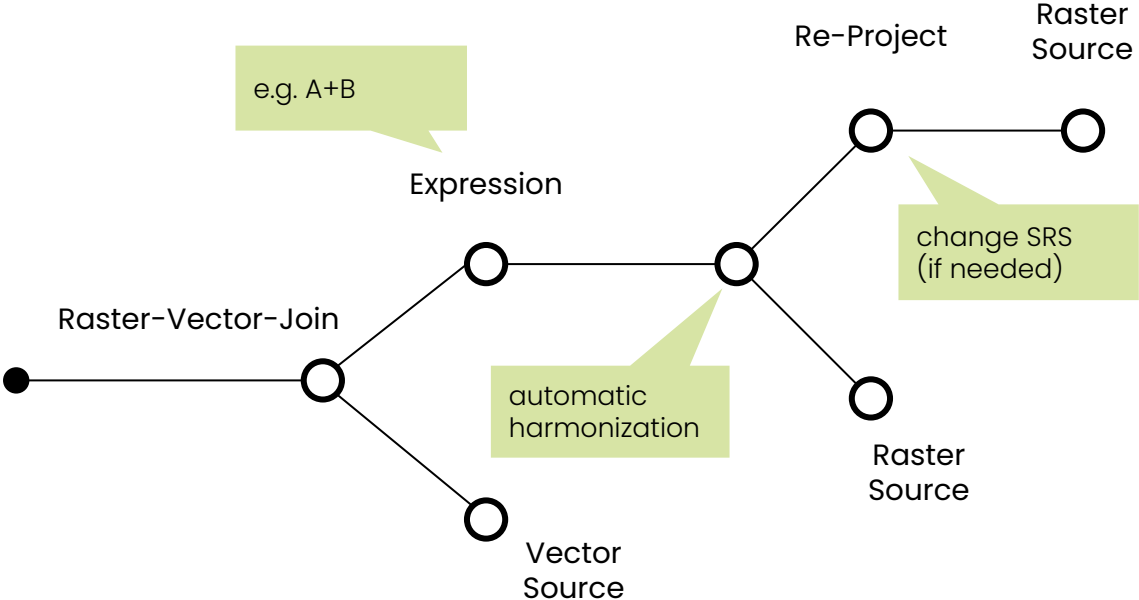
Instantiate STAC Provider

- Implements listing
- Implements resolving datasets

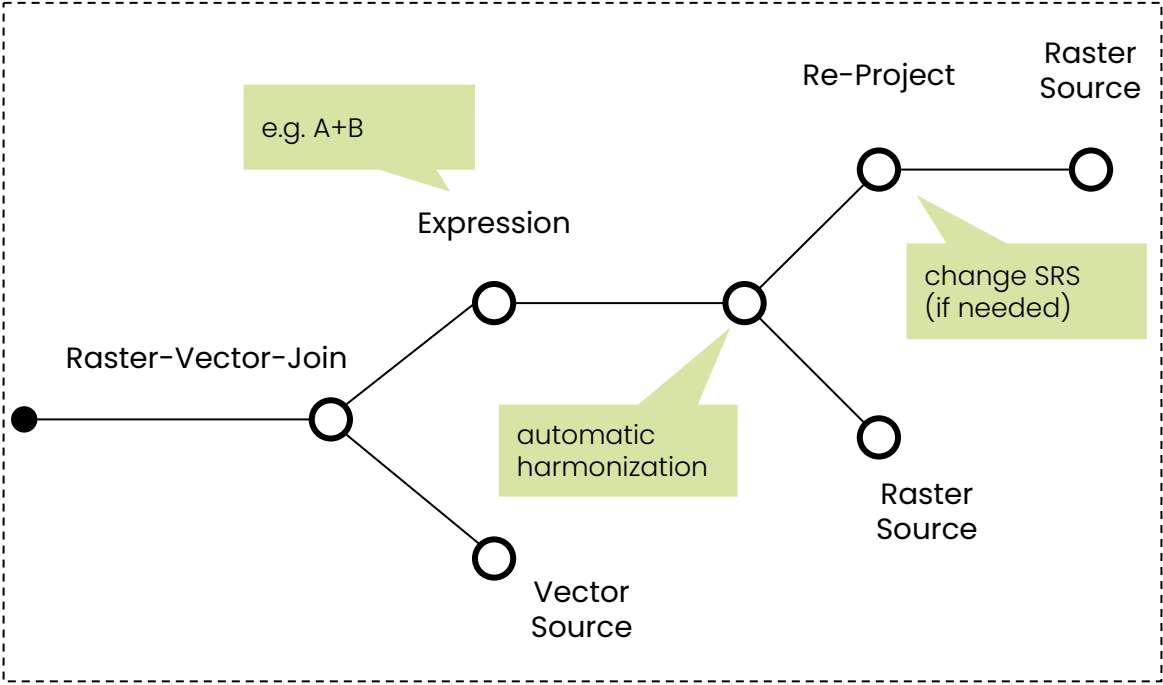
③

Integrated time series dataset

Concept 3: Workflows



Concept 3: Workflows

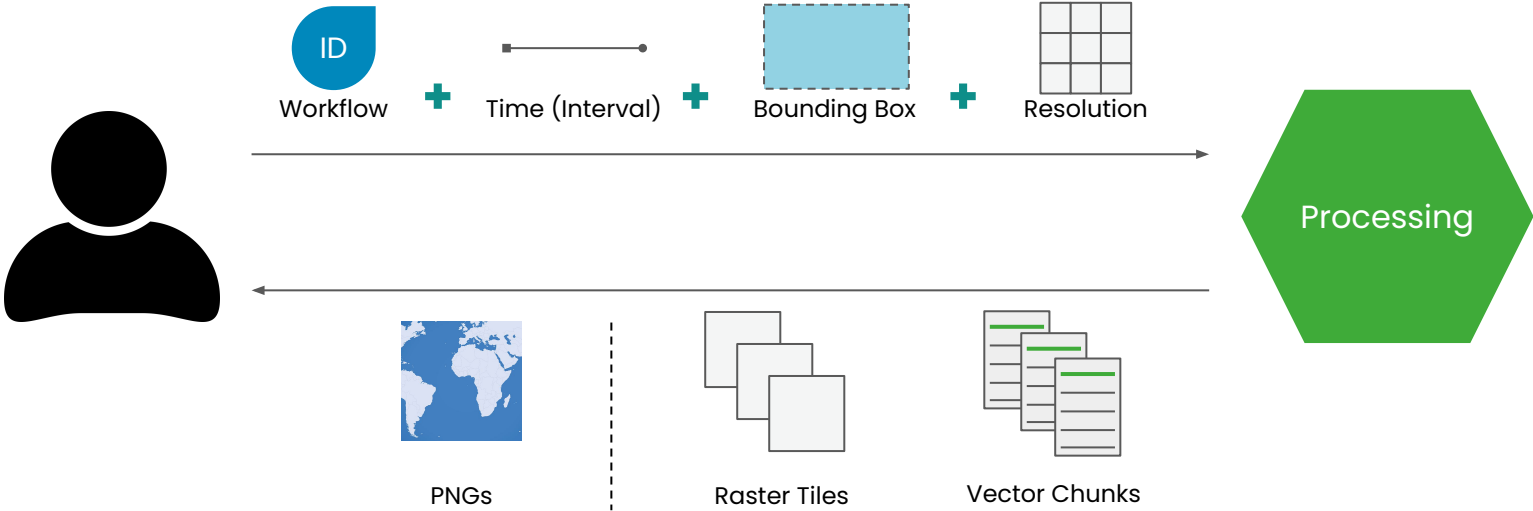


Workflow Description



Processing

Concept 4: Query like a data cube



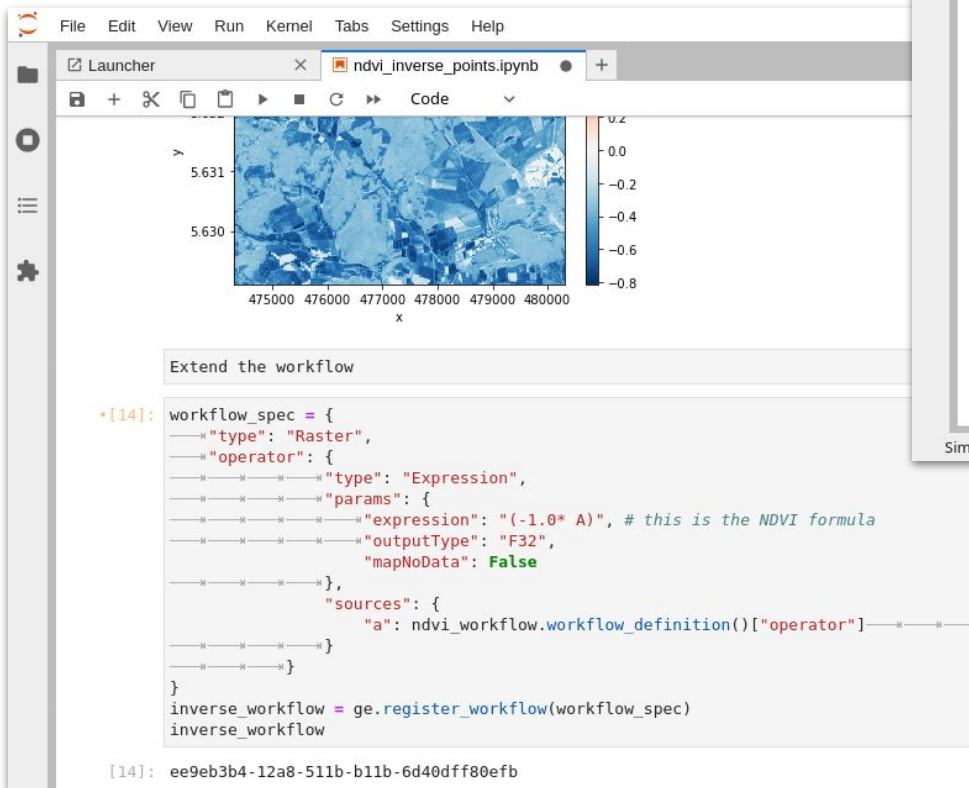
Concept 5: Access from your favorite environment

The image displays the GeoEngine Web UI interface, illustrating the process of calculating NDVI. The interface is divided into several panels:

- Left Panel (Lineage for NDVI):** Shows the workflow for calculating NDVI. It starts with two **GdalSource** nodes, each with a `data` property of type `"intern..."`. Arrows labeled 'a' and 'b' point to an **Expression** node. The **Expression** node contains the formula:
$$\text{express... } (A - B) / (A + B)$$
 and has an `output... F32` property. A dashed arrow points from the **Expression** node to the final **NDVI** node.
- Top Panel (GeoEngine Header):** Displays the application name, search icons, and a timestamp: `01.05.2019 12:00:00`.
- Left Sidebar (Layers):** Lists the layers in the workspace: **NDVI**, **Sentinel 2 Band 08**, and **Sentinel 2 Band 04**.
- Right Panel (Operators):** A search interface for operators. It includes a search bar and a list of operators categorized by type:
 - Mixed:** **Raster Vector Join** (Attach raster values to multi-point data).
 - Plots:** **Basic Statistics** (Get statistics for raster layer), **Box Plot** (Box plot your data), **Class Histogram** (Create a class histogram from categorical vector or raster data), **Histogram** (Create a histogram from vector or raster data), **Scatter Plot** (Scatter plot your data), **Temporal Feature Attribute Plot** (Create a multi line chart over the attribute values of a feature layer), **Temporal Raster Mean Plot** (Create an area chart over the mean pixel values of the images of a raster time series).
 - Raster:** **Convert Raster Data Type** (Converts (casts) the raster type), **Expression** (Calculate an expression on a raster), **Interpolation** (Interpolates raster data), **Temporal Raster Aggregation** (Aggregate raster time series).
- Map View (Center):** A satellite-style map showing a landscape with a color scale from blue to yellow. A cluster of red circular markers is overlaid on the map.
- Bottom Panel (Color Map):** A color map legend for the NDVI layer, showing a scale from -0.1 to 0.8. The current color map is set to **VIRIDIS**, and the **Reverse colormap** option is unchecked.

Web UI
WYSIWYG

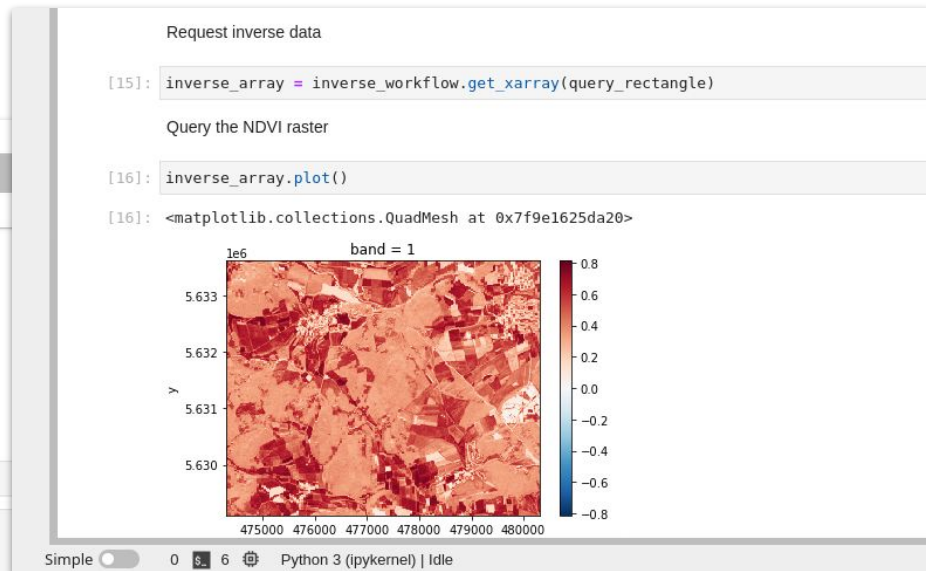
Concept 5: Access from your favorite environment



The screenshot shows a Jupyter Notebook window titled "ndvi_inverse_points.ipynb". The interface includes a menu bar (File, Edit, View, Run, Kernel, Tabs, Settings, Help) and a toolbar with icons for file operations and code execution. Below the toolbar, there is a plot of a raster image with a color scale ranging from -0.8 to 0.2. The plot is labeled with x and y coordinates: x-axis from 475000 to 480000, and y-axis from 5630 to 5631. Below the plot, the text "Extend the workflow" is visible. The main content area contains a code cell with the following Python code:

```
[14]: workflow_spec = {
  "type": "Raster",
  "operator": {
    "type": "Expression",
    "params": {
      "expression": "(-1.0* A)", # this is the NDVI formula
      "outputType": "F32",
      "mapNoData": False
    },
    "sources": {
      "a": ndvi_workflow.workflow_definition()["operator"]
    }
  }
}
inverse_workflow = ge.register_workflow(workflow_spec)
inverse_workflow
```

At the bottom of the code cell, the output is displayed as: [14]: ee9eb3b4-12a8-511b-b11b-6d40dff80efb



The screenshot shows a Jupyter Notebook window with the following code cells:

```
Request inverse data

[15]: inverse_array = inverse_workflow.get_xarray(query_rectangle)

Query the NDVI raster

[16]: inverse_array.plot()

[16]: <matplotlib.collections.QuadMesh at 0x7f9e1625da20>
```

Below the code cells, there is a plot of a raster image with a color scale ranging from -0.8 to 0.8. The plot is labeled with x and y coordinates: x-axis from 475000 to 480000, and y-axis from 5630 to 5633. The plot is titled "1e6 band = 1". Below the plot, the text "Simple" is visible, followed by a radio button and the number "0". To the right of the radio button, there is a small icon and the text "Python 3 (ipykernel) | Idle".

Jupyter Notebooks
Programmatically

Example: Random Forest using custom point and Sentinel data

Retrieve S2 data
via STAC



Monthly mean of
cloud-free NDVI



Load sensor training
points



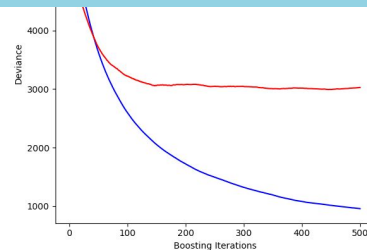
Attach Sentinel-2
product values to points

```
end=end_int,
),
resolution=ge.SpatialResolution(
    10.0,
    10.0,
),
srs="EPSG:32632",
})
SP_res

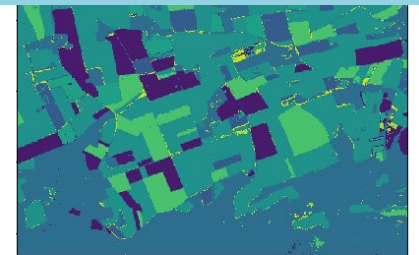
Out[18]:
```

	B04_4	B04_8	B08_4	ID	NDVI_9	USE_TXT	NDVI_7	B03_6	B08_5	B08_8	...	B10_4
0	670.0	580.0	4467.0	0	0.195496	Ösösten	0.263462	838.0	6426.0	1614.0	...	1165.0
1	555.0	693.0	3485.0	1	0.204141	Getreide	0.321335	744.0	4763.0	3556.0	...	703.0
2	467.0	730.0	4042.0	2	0.198928	Getreide	0.333275	559.0	NaN	2369.0	...	1081.0

Stream data in RF model
using Python & Jupyter



Apply to all pixel values



Summary

- Geo Engine platform for geo processing & analysis
- Native time series
- Workflows
- Data harmonization
- Low-code
- Jupyter notebooks

Outlook

- Integrate machine learning framework(s) into Geo Engine
- ML ingestion inside
- ML as workflow operator

Call for Cooperation

- Data ingestion use case
- Data products from different EO sources
- Proof-of-concepts



Thank you!



Geo
Engine

Geo Engine GmbH
Am Kornacker 68
35041 Marburg



info@geoengine.de



[geoengine-de](https://www.linkedin.com/company/geoengine-de)



[@teamgeoengine](https://twitter.com/teamgeoengine)



www.geoengine.de