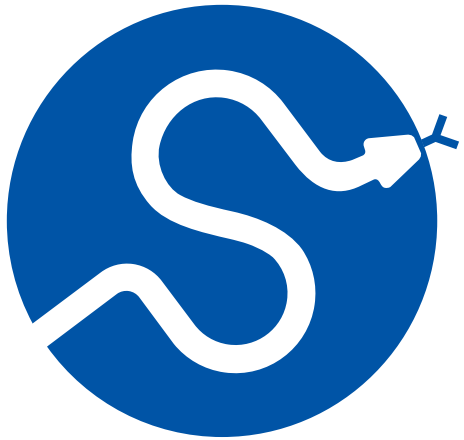
Support for the Python Array API Standard in SciPy

Jake Bowhay, University of Bristol & SciPy Maintainer



Background

Credit: Aaron Meurer, 'Python Array API Standard', SciPy 2023
https://github.com/data-apis/scipy-2023-presentation/blob/main/presentation/Slides.pdf

Background

Credit: Aaron Meurer, 'Python Array API Standard', SciPy 2023
https://github.com/data-apis/scipy-2023-presentation/blob/main/presentation/Slides.pdf

Background

Credit: Aaron Meurer, 'Python Array API Standard', SciPy 2023
https://github.com/data-apis/scipy-2023-presentation/blob/main/presentation/Slides.pdf

Solution:

1. The Array API standard defines what it means for a library to provide an array.
2. `array-api-compat` provides a compatibility layer for common libraries.

Example Usage

First set environment variable:

```
export SCIPY_ARRAY_API=1
```

Then:

```
>>> import torch
>>> from scipy.cluster.vq import vq
>>> code_book = torch.tensor([[1., 1., 1.],
                              [2., 2., 2.]])
>>> features  = torch.tensor([[1.9, 2.3, 1.7],
                              [1.5, 2.5, 2.2],
                              [0.8, 0.6, 1.7]])
>>> code, dist = vq(features, code_book)
>>> code
tensor([1, 1, 0], dtype=torch.int32)
>>> dist
tensor([0.4359, 0.7348, 0.8307])
```

Example Implementation

```
xp = array_namespace(a)

dtype = a.dtype
a, mask = _put_val_to_limits(...)
min = xp.min(a, axis=axis)
n = xp.sum(xp.asarray(~mask, dtype=a.dtype), axis=axis)
res = xp.where(n != 0, min, xp.nan)
if not xp.any(xp.isnan(res)):
    # needed if input is of integer dtype
    res = xp.astype(res, dtype, copy=False)
return res
```

Example Implementation

```
xp = array_namespace(a)

dtype = a.dtype
a, mask = _put_val_to_limits(...)
min = xp.min(a, axis=axis)
n = xp.sum(xp.asarray(~mask, dtype=a.dtype), axis=axis)
res = xp.where(n != 0, min, xp.nan)
if not xp.any(xp.isnan(res)):
    # needed if input is of integer dtype
    res = xp.astype(res, dtype, copy=False)
return res
```

Example Implementation

```
xp = array_namespace(a)


dtype = a.dtype
a, mask = _put_val_to_limits(...)
min = xp.min(a, axis=axis)
n = xp.sum(xp.asarray(~mask, dtype=a.dtype), axis=axis)
res = xp.where(n != 0, min, xp.nan)
if not xp.any(xp.isnan(res)):
    # needed if input is of integer dtype
    res = xp.astype(res, dtype, copy=False)
return res
```

More Info

- https://data-apis.org/array-api/2024.12/
- https://docs.scipy.org/doc/scipy/dev/api-dev/array_api.html
- https://scikit-learn.org/stable/modules/array_api.html