Package: nirs4allio (via r-universe)

June 12, 2026

Title Dataset-Assembly Bridge for the nirs4all Ecosystem

Version 0.1.1

Description R binding over the nirs4all-io C ABI (n4io_*): normalize inputs into a canonical DatasetSpec, infer a DatasetPlan, and validate a DatasetSpec. The JSON surface crosses the stable C ABI; the low-level 'n4io_*' functions take and return canonical JSON strings, while the idiomatic 'nio_*' layer accepts native R inputs (a path, a vector of files, or a config list) and returns typed S3 objects with print and as.data.frame methods.

License CeCILL-2.1 | AGPL-3

Encoding UTF-8

Imports jsonlite

SystemRequirements Cargo (Rust toolchain), rustc

NeedsCompilation yes

RoxygenNote 7.3.2

Config/pak/sysreqs libclang-dev

Repository https://gbeurier.r-universe.dev

Date/Publication 2026-06-12 07:41:38 UTC

RemoteUrl https://github.com/GBeurier/nirs4all-io

RemoteRef HEAD

RemoteSha a0d5587deb0ccd28c34f93c7abf7d91053ec8604

RemoteSubdir bindings/r

Contents

`as.data.frame.n4io_plan` *Scored decisions of a plan as a data.frame.*

Description

One row per scored decision (structure, signal type, task type, ...), with the chosen `value`, the confidence `score`, whether it is `ambiguous`, and the count of `alternatives`.

Usage

```
## S3 method for class 'n4io_plan'
as.data.frame(x, ...)
```

Arguments

`x`	An `n4io_plan`.
`...`	Ignored.

Value

A data.frame with columns `decision`, `value`, `score`, `ambiguous`, `n_alternatives`.

`n4io_abi_version` *The C ABI version string.*

Description

The C ABI version string.

Usage

```
n4io_abi_version()
```

n4io_infer — *Inspect a data input and return the scored DatasetPlan (JSON string).*

Description

Inspect a data input and return the scored DatasetPlan (JSON string).

Usage

```
n4io_infer(input_json, conventions_json = NULL)
```

Arguments

`input_json`	A JSON path string or file-list array.
`conventions_json`	Optional JSON array of convention names, or NULL.

Value

The DatasetPlan as a JSON string.

n4io_to_spec — *Normalize an input into a canonical DatasetSpec (JSON string).*

Description

Normalize an input into a canonical DatasetSpec (JSON string).

Usage

```
n4io_to_spec(input_json, conventions_json = NULL)
```

Arguments

`input_json`	A JSON value: a spec object, a path string (e.g. `'"/data/run"'`), or a file-list array (e.g. `'["a.csv","b.csv"]'`).
`conventions_json`	Optional JSON array of convention names, or NULL.

Value

The canonical DatasetSpec as a JSON string.

n4io_validate — *Validate a DatasetSpec JSON string; raises an error if invalid.*

Description

Validate a DatasetSpec JSON string; raises an error if invalid.

Usage

```
n4io_validate(spec_json)
```

Arguments

spec_json	A DatasetSpec as a JSON string.

Value

Invisibly NULL on success.

nio_infer — *Infer a scored dataset plan from a native R input.*

Description

Native-R wrapper over [n4io_infer()]: `input` is JSON-encoded internally (with jsonlite) and the scored `DatasetPlan` is parsed back into a typed `n4io_plan` object.

Usage

```
nio_infer(input, conventions = NULL)
```

Arguments

input	A path (character scalar), a vector of files (character vector), or a config `list`.
conventions	Optional character vector of convention names, or `NULL` for the binding default.

Value

An object of class `n4io_plan`: the parsed plan (recommendations, scored decisions, `resolved_spec`, `overall_score`). Use `as.data.frame.n4io_plan` for the scored decisions as a data.frame and `nio_resolved_spec` for the editable spec.

See Also

[n4io_infer()] for the raw JSON surface.

Examples

```
## Not run:
plan <- nio_infer("/data/run")
print(plan)
as.data.frame(plan)          # one row per scored decision
spec <- nio_resolved_spec(plan)

## End(Not run)
```

`nio_resolved_spec` *The resolved (editable) DatasetSpec carried inside a plan.*

Description

The resolved (editable) DatasetSpec carried inside a plan.

Usage

```
nio_resolved_spec(plan)
```

Arguments

`plan`	An `n4io_plan` from [nio_infer()].

Value

An `n4io_spec` object built from `plan$resolved_spec`.

`nio_to_spec` *Normalize a native R input into a canonical DatasetSpec.*

Description

Native-R wrapper over [n4io_to_spec()]: `input` is JSON-encoded internally and the canonical `DatasetSpec` is parsed back into a typed `n4io_spec` object that round-trips byte-for-byte across the ABI.

Usage

```
nio_to_spec(input, conventions = NULL)
```

Arguments

`input`	A path (character scalar), a vector of files (character vector), or a config `list`.
`conventions`	Optional character vector of convention names, or `NULL`.

Value

An object of class `n4io_spec`: the parsed canonical spec. Pass it directly to [nio_validate()] or [nio_infer()].

See Also

[n4io_to_spec()] for the raw JSON surface.

Examples

```
## Not run:
spec <- nio_to_spec("/data/run")
print(spec)
nio_validate(spec)

## End(Not run)
```

`nio_validate` *Validate a DatasetSpec.*

Description

Native-R wrapper over [n4io_validate()] that accepts an `n4io_spec`, a plain `list`, or a JSON string. Returns invisibly `TRUE` on success and signals an informative error otherwise.

Usage

```
nio_validate(spec)
```

Arguments

`spec`	An `n4io_spec`, a `list`, or a JSON string.

Value

Invisibly `TRUE` on success; otherwise an error is signalled.

See Also

[n4io_validate()] for the raw JSON surface.

Examples

```
## Not run:
spec <- nio_to_spec("/data/run")
nio_validate(spec)

## End(Not run)
```

Index