Package: assesslite (via r-universe)

July 12, 2026

Title AssessLite: Test What Your Analysis Depends On

Version 0.4.0

Description Structural assumption assessment for causal analysis. Makes the invariance assumptions behind a result explicit, attacks them with transformation tests (unit permutation, cluster holdout, temporal split, subgroup stability), returns three-way verdicts (stable, unstable, not resolvable), and exports the reasoning path as a durable audit record. AssessLite is the product, structural_audit() runs the assessment, and write_audit() writes the audit record. Sibling to 'recoverlite': recoverlite asks whether a method can recover the quantity it estimates; AssessLite asks what a result depends on and whether it survives the violation of those assumptions. R implementation of the AssessLite core specification v0.1. Records claim-to-source support, never truth: a stable verdict means the conclusion survived the attacks that were run, at the precision the data allowed.

License Apache License (>= 2)

URL https://github.com/heidihelena/assesslite

BugReports https://github.com/heidihelena/assesslite/issues

Encoding UTF-8

Depends R (>= 4.1)

Imports jsonlite, stats, utils, tools

Suggests survival, testthat (>= 3.0.0), knitr, rmarkdown

VignetteBuilder knitr

RoxygenNote 7.3.3

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

Date/Publication 2026-07-12 21:26:59 UTC

RemoteUrl https://github.com/heidihelena/assesslite

RemoteRef HEAD

RemoteSha 278c394b0fe9182cfa3f26391df9ff560769db53

Contents

`assume_invariance` *Assume an invariance: the analysis relies on it and it should be attacked*

Description

Assume an invariance: the analysis relies on it and it should be attacked

Usage

```
assume_invariance(audit, invariance, rationale, licenses)
```

Arguments

`audit`	a structural_audit object.
`invariance`	a name from ‘invariance_vocabulary()‘.
`rationale`	why the claim is scientifically defensible here.
`licenses`	what inferential step the claim buys (pooling, transport, ...).

Value

the audit with the ledger entry added.

assumption_lattice *Build the pooling assumption lattice*

Description

Build the pooling assumption lattice

Usage

```
assumption_lattice(audit)
```

Arguments

audit	a structural_audit object (with a fitted estimate).

Value

the audit with a $lattice element.

decide *Apply the decision rules to a tested audit*

Description

Apply the decision rules to a tested audit

Usage

```
decide(
  audit,
  abstain_if = list(estimate_sign_changes = TRUE, effect_crosses_threshold = NULL)
)
```

Arguments

audit	a structural_audit after test_invariance().
abstain_if	list of user-declared abstention rules: estimate_sign_changes (logical) and effect_crosses_threshold (numeric on the natural scale, or NULL).

`declare_graph` *Declare a causal DAG for the graph_check and adjustment_check attacks*

Description

Declare a causal DAG for the graph_check and adjustment_check attacks

Usage

```
declare_graph(audit, edges, latent = character())
```

Arguments

`audit`	a structural_audit object.
`edges`	character vector of directed edges, e.g. c("age -> adherence", "stage -> survival"). Nodes are the union of everything named.
`latent`	character vector of node names that are part of the causal structure but not measured (e.g. an unmeasured confounder). Latent nodes may not enter an adjustment set, and implications that touch them are not testable.

`invariance_vocabulary` *Canonical invariance identifiers (core spec v0.1)*

Description

Canonical invariance identifiers (core spec v0.1)

Usage

```
invariance_vocabulary()
```

reject_invariance *Reject an invariance: the analyst asserts it does not hold here*

Description

Reject an invariance: the analyst asserts it does not hold here

Usage

```
reject_invariance(audit, invariance, rationale, licenses)
```

Arguments

audit	a structural_audit object.
invariance	a name from ‘invariance_vocabulary()‘.
rationale	why the claim is indefensible here.
licenses	what the rejection removes from scope.

Value

the audit with the ledger entry added.

render_report *Render the audit as a self-contained HTML report*

Description

Render the audit as a self-contained HTML report

Usage

```
render_report(audit, path)
```

Arguments

audit	a structural_audit object after ‘decide()‘.
path	file path to write the HTML report to.

Value

the path, invisibly.

`structural_audit` *Declare the structure of a causal analysis and open its audit*

Description

Declare the structure of a causal analysis and open its audit

Usage

```
structural_audit(
  data,
  outcome,
  exposure,
  covariates = character(),
  cluster = NULL,
  time = NULL,
  subgroups = character(),
  coords = NULL,
  unit_id = NULL,
  edges = NULL,
  unit = "unit",
  estimand = NULL
)
```

Arguments

`data`	data frame of the analysis sample.
`outcome`	a single column name for a GLM outcome, or c(time, status) column names for a Cox model (requires the survival package).
`exposure`	column name of the exposure of interest.
`covariates`	character vector of adjustment covariate names.
`cluster`	column name of the cluster variable (hospital, site), or NULL.
`time`	column name of the calendar-time variable, or NULL.
`subgroups`	character vector of subgroup variable names.
`coords`	length-2 character vector of coordinate columns c(x, y) for the spatial attack, or NULL.
`unit_id`	column naming each unit, required with ‘edges‘, or NULL.
`edges`	a two-column data frame of undirected unit-id pairs defining a network for the interference attack, or NULL.
`unit`	what one row is (e.g. "patient").
`estimand`	plain-language statement of the target quantity.

Value

a structural_audit object with the full-sample estimate fitted.

`test_invariance` *Run attacks against the declared invariances*

Description

Run attacks against the declared invariances

Usage

```
test_invariance(
  audit,
  tests = c("unit_permutation", "cluster_holdout", "temporal_split",
    "subgroup_stability"),
  seed = 1,
  confounding_benchmark = 1.25,
  outcome_node = NULL,
  spatial_k = 3,
  tip_ratio = NULL,
  confounder_prevalence = 0.2,
  spatial_knn = 8,
  exposure_map = "mean"
)
```

Arguments

`audit`	a structural_audit object with a populated ledger.
`tests`	character vector of attacks to run: unit_permutation, cluster_holdout, temporal_split, subgroup_stability, confounding_sensitivity, confounding_scenarios, graph_check, adjustment_check, positivity_check, spatial_holdout, interference_check.
`seed`	integer seed for the permutation test.
`confounding_benchmark`	plausible unmeasured-confounding strength on the E-value (risk-ratio) scale, used by confounding_sensitivity (default 1.25).
`outcome_node`	graph node to treat as the outcome for adjustment_check (default: the model's outcome column).
`spatial_k`	grid resolution (k x k blocks) for spatial_holdout (default 3).
`tip_ratio`	decision threshold on the ratio scale for confounding_scenarios, or NULL for the null (default NULL).
`confounder_prevalence`	assumed confounder prevalence for confounding_scenarios (default 0.2).
`spatial_knn`	neighbours for the spatial_autocorrelation weight matrix (default 8).
`exposure_map`	neighbour-exposure summary for interference_check: "mean" (default), "any", or "sum".

Value

the audit with the requested attacks recorded and ledger verdicts updated.

`write_audit` *Write the audit to a JSON file conforming to the core audit schema*

Description

Write the audit to a JSON file conforming to the core audit schema

Usage

```
write_audit(audit, path)
```

Arguments

`audit`	a structural_audit object after ‘decide()‘.
`path`	file path to write the JSON audit record to.

Value

the path, invisibly.

Index