Package: infercalcR (via r-universe)

November 14, 2026

Type Package

Title Statistical Inference and Diagnostic Calculations

Version 0.1.0

Description Provides explicit statistical procedures, descriptive summaries, diagnostic calculations and a common result contract. Each procedure is chosen through an explicit method specification rather than by data-driven pretests, and returns finalized estimates, intervals and p-values together with metadata that records the method identity and numerical provenance. Includes Boschloo's unconditional test for two-by-two tables (Boschloo, 1970, <doi:10.1111/j.1467-9574.1970.tb00104.x>), diagnostic accuracy and classifier metrics, and distribution calculators. Standard R methods are used where available; optional backends such as 'lme4', 'nlme' and 'geepack' are used only when installed.

License MIT + file LICENSE

Encoding UTF-8

Language en-US

Depends R (>= 4.1.0)

Imports stats, utils

Suggests digest, jsonlite, cluster, nlme, MASS, nnet, lme4, geepack, knitr, rmarkdown, testthat (>= 3.1.7), withr, DescTools (>= 0.99.60), exact2x2 (>= 1.7.0), Exact (>= 3.3), brunnermunzel (>= 2.0)

URL https://github.com/CTTIR/infercalcR, https://cttir.github.io/infercalcR/

BugReports https://github.com/CTTIR/infercalcR/issues

VignetteBuilder knitr

Config/testthat/edition 3

Config/Needs/website CTTIR/themakR

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

Date/Publication 2026-11-14 22:11:03 UTC

RemoteUrl https://github.com/CTTIR/infercalcR

RemoteRef HEAD

RemoteSha 5216753e4366becdd80626b274b81e3e7c4845a4

Contents

as_inference.htest 2
describe_data 4
kernel-providers 5
legacy-computation 10
method_spec 17
observed-methods 18
parallel_analysis 23
result-contract 25

Index **27**

`as_inference.htest` *Explicit conversion of supplied tests and fitted models*

Description

Convert recognized finalized test objects or summarize an existing fitted model once, without refitting.

Usage

```
## S3 method for class 'htest'
as_inference(x, method_id = NULL, n = NA_real_, population,
  groups = NULL, event = NULL, n_excluded = NA_real_, ...)
## S3 method for class 'lm'
as_inference(x, conf.level = .95, population, ...)
## S3 method for class 'glm'
as_inference(x, conf.level = .95, population,
  interval = "wald", ...)
```

Arguments

`x`	A recognized `htest`, `lm` or `glm` object.
`method_id`	Optional identity, which must agree with the stored test. For an imported signed-rank test, a compatible explicit identity declares the otherwise unrecoverable one-sample or paired design.
`n`	Known analysis sample counts, or explicit NA when unavailable. Never reconstructed from degrees of freedom.

`population`	Required description of the supplied analysis population.
`groups`	Two ordered labels for independent two-sample location, odds-ratio or variance-ratio contrasts; optionally, labels of the two measurements for a declared paired signed-rank contrast.
`event`	Explicit event definition for binary/proportion and odds-ratio tests.
`n_excluded`	Known exclusion count, otherwise NA.
`conf.level`	Requested confidence level for summaries of supplied models.
`interval`	Fitted-glm conversion explicitly supports `"wald"`. The named logistic procedure provides profile-likelihood slope intervals separately.
`...`	Reserved; unknown arguments are rejected.

Details

Wilcoxon intervals retain the native-reported confidence level. Older R versions can record a nominal or rounded level whose exact attainment differs; newer versions can return a fully unbounded conservative interval. The common interval-method label and metadata identify this convention. No level, endpoint, or captured native diagnostic is replaced.

Recognized htest objects include standard t, Wilcoxon, binomial, proportion, Fisher, correlation, variance and common omnibus tests. Stored estimates, intervals and p-values are preserved with method-specific estimands. Two group means become their difference; no odds ratio is invented for a general contingency table. Proportion tests across more than two groups retain an omnibus row without a fabricated scalar probability. Unknown htest methods are rejected.

A signed-rank htest has the same method label whether it came from a one-sample analysis or paired differences. Its default identity is therefore `imported_signed_rank`, with unknown design. Supplying `method_id = "signed_rank_wilcoxon_one_sample"` or `method_id = "signed_rank_wilcoxon"` declares the one-sample or paired design, respectively; metadata records that this design was declared by the caller. The converter never infers design from the printed data expression. Stored estimates, confidence intervals, null values and p-values are unchanged. For a paired result, supplied `n` counts analyzed pairs, not twice the pair count.

Linear models use classical model-based marginal t intervals. GLM conversion is restricted to binomial-logit, Poisson-log and Gaussian-identity models; it uses model-based covariance and explicitly labeled Wald or t intervals. It does not invoke generic profile dispatch or switch covariance backends. Binomial slopes may also be shown as odds ratios and Poisson slopes as rate ratios; intercepts are not ratios. Exponentiation overflow is a numerical failure, distinct from a legitimate unbounded sampling interval. Aliased coefficients remain unavailable.

Value

A common finalized `inference_result`. The supplied source object is unchanged.

Examples

```
fit <- stats::t.test(c(1, 3, 4, 8), c(2, 4, 5, 6, 9))
as_inference(fit, n = c(4, 5), population = "Synthetic independent groups",
  groups = c("x", "y"))
```

describe_data *Descriptive summaries without hypothesis tests*

Description

Compute continuous or categorical summaries with explicit denominators and missing counts using standard R.

Usage

```
describe_data(data, variables = setdiff(names(data), group), group = NULL,
  labels = NULL, units = NULL, types = NULL)
```

Arguments

data	A data frame.
variables	Distinct column names to summarize.
group	Optional single grouping column name. Missing groups are retained explicitly.
labels, units	Optional named display metadata keyed by variable name.
types	Optional named continuous/categorical overrides. Ordered factors are categorical by default.

Details

Numeric summaries are mean, sample standard deviation, median, quartiles (type 7), minimum and maximum. Categorical proportions use nonmissing observations within each group as their denominator; n.missing and n.total remain separate. Empty populations produce explicit unavailable summaries. Infinite measurements are rejected. No hypothesis tests or automatic p-value columns are calculated.

Value

A descriptive_result inheriting from inference_result.

Examples

```
describe_data(data.frame(value = c(1, 3, NA), category = factor(c("a", "b", "a"))))
```

kernel-providers *Explicit Shared Numerical and Compatibility Providers*

Description

Preserved numerical procedures extracted from inferguidR, with explicit inputs, typed statuses and method settings. Calculation and display are separate.

Usage

```
boschloo_design_kernel(n1, n2, alpha = 0.05, variant, control = boschloo_control())

binomial_region_probability(indices, n1, n2, p1, p2)

binomial_table_space(n1, n2)

boschloo_region_record(kernel, control = boschloo_control())

boschloo_kernel_status(kernel)

boschloo_control_record(control)

boschloo_control_from_record(record)

boschloo_validate_control(control)

boschloo_validate_design(n1, n2)

boschloo_variant_details(variant = NULL)

boschloo_variant_record(info)

boschloo_engine_info()

boschloo_control_call(control)

fisher_ordering_probabilities(n1, n2, variant, control = boschloo_control(),
  convention = "same")

boschloo_plot_data_record(plot_id, title, data, fixed, scope, reference = list(),
  notes = character(), status = ig_status("ok"))

classification_dictionary()

classification_contract()

classification_reason(code, lang = "en")
```

```
classification_metric_ids()

classification_short_label(id, lang = "en")

classification_check_thresholds(threshold_probability)

classification_check_bins(calibration_bins)

classification_check_partial(partial_auc)

legacy_canonical_json(x)

legacy_hash_value(x)

legacy_sha256(text)

legacy_parse_json(text)

legacy_format_double(x)

legacy_parse_decimal(text)

legacy_double_values(x)

legacy_nonfinite_paths(x, path = "")

legacy_r_literal(x)

legacy_unclass_frozen(x)

legacy_check_spec(spec, kind)

legacy_json_limits()

normal_probability(q, mean = 0, sd = 1, lower_tail = TRUE, log_p = FALSE)

normal_quantile(p, mean = 0, sd = 1, lower_tail = TRUE, log_p = FALSE)

t_probability(q, df, ncp = 0, lower_tail = TRUE, log_p = FALSE)

t_quantile(p, df, ncp = 0, lower_tail = TRUE, log_p = FALSE)

chisq_probability(q, df, ncp = 0, lower_tail = TRUE, log_p = FALSE)

chisq_quantile(p, df, ncp = 0, lower_tail = TRUE, log_p = FALSE)

f_probability(q, df1, df2, ncp = 0, lower_tail = TRUE, log_p = FALSE)
```

```
f_quantile(p, df1, df2, ncp = 0, lower_tail = TRUE, log_p = FALSE)

binomial_probability(q, size, prob, lower_tail = TRUE, log_p = FALSE)

log_probability_add(a, b)

log_probability_sum(x)

invert_log_cdf(logcdf, p, lower_tail, log_p, support, start)

f_lower_log_bound(q, df1, df2, ncp)

noncentral_t_density(x, df, ncp)
```

Arguments

n1	Number of independent binomial observations in group 1.
n2	Number of independent binomial observations in group 2.
alpha	Decision level strictly between zero and one; NULL omits a decision where supported.
variant	Explicit Boschloo variant: minlike, central, less, or greater. Classifier variants are retained in their records.
control	Numerical settings from boschloo_control().
indices	Distinct indices of rejected tables in x1-fastest two-binomial table order.
p1	Event probability in group 1.
p2	Event probability in group 2.
kernel	Unmodified design kernel returned by boschloo_design_kernel().
record	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.
info	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.
convention	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.
plot_id	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.
title	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.
data	A data frame containing observations, with explicit truth and score columns.
fixed	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.
scope	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

reference	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.
notes	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.
status	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.
code	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.
lang	Display-label language, en or de.
id	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.
threshold_probability	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.
calibration_bins	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.
partial_auc	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.
x	Input data, finalized result, or numeric values as specified by the selected function.
text	A data-only canonical JSON document or decimal string, according to the function.
path	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.
spec	A classification specification returned by classification_spec().
kind	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.
q	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.
mean	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.
sd	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.
lower_tail	Logical: use the lower tail when TRUE.
log_p	Logical: return or interpret probabilities on the logarithmic scale.
p	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.
df	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.
ncp	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

df1	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.
df2	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.
size	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.
prob	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.
a	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.
b	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.
logcdf	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.
support	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.
start	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

Details

Boschloo tests use Fisher ordering and bounded nuisance maximization. The central variant doubles the smaller separately maximized one-sided value. Rejection uses the numerical upper bound; this bound is not a sampling confidence interval. The shared Boschloo inference path requires only standard R. Historical hashed specifications and the legacy calculator decimal parser require the optional packages digest and/or jsonlite and give an explicit missing-dependency error when absent.

Classifier specifications preserve the positive class, direction, cutoff origin, missing-value exclusions, evaluation split and raw-data fingerprint. Raw observations are not retained in the common reporting payload. Basic count metrics, binormal rates and static dictionary access do not require contributed packages. Undefined metrics retain statuses and denominator reasons.

Provider functions are public interfaces for validated package consumers. Historical canonical primitives accept already converted plain data; guidance-owned schema dispatch remains in inferguidR. No provider evaluates code from input strings. Calculator expressions use a restricted parser.

The inherited validation metadata records prior scoped checks; package extraction and adapter parity do not certify all scientific designs.

Value

Computational functions return typed lists or ordinary data frames; common adapters return inference_result objects. Metadata providers return plain lists or vectors, probability kernels return numeric values, and validators signal typed errors for invalid inputs.

Examples

```
k <- boschloo_design_kernel(3, 4, variant = "minlike")
binomial_region_probability(k$idx, 3, 4, .2, .8)
t_probability(2, df = 10)
```

---

legacy-computation *Diagnostic and Exact Numerical Calculations*

Description

Preserved numerical procedures extracted from inferguidR, with explicit inputs, typed statuses and method settings. Calculation and display are separate.

Usage

```
attach_classification_data(spec, data)

binormal_model(auc, sigma = 1)

binormal_rates(cutoff, auc, sigma = 1)

boschloo_compare_fisher(n1, n2, alpha = 0.05, variant, p1 = NULL, p2 = NULL,
  fisher_convention = c("same", "minlike"), control = boschloo_control())

boschloo_control(tie_tolerance = 1e-07, grid_points = 1001L, nuisance_tol = 1e-09,
 polish_tol = 1e-12, max_levels = 60L, ambiguity_band = c(1e-12, 1e-06), cache = TRUE)

boschloo_plotdata_lattice(n1, n2, alpha = 0.05, variant, p1 = NULL, p2 = NULL,
  observed = NULL, pvalues = TRUE, control = boschloo_control())

boschloo_plotdata_nuisance(x, grid_points = 401L, control = NULL)

boschloo_region(n1, n2, alpha = 0.05, variant, control = boschloo_control())

boschloo_test(x1, n1, x2, n2, variant, alpha = NULL, groups = c("group 1", "group 2"),
  event = "event", control = boschloo_control())

boschloo_variants()

calculator_functions()

calibration_summary(spec, bins = 10)

classification_auc(spec, partial_auc = NULL, level = NULL)

classification_evaluated_spec(x)
```

```
classification_metrics(x, count_kind = c("observed", "frequency_weighted", "expected"),
  lang = c("en", "de"))

classification_plot_data(x, panel = .ig_cls_panels, model_id = NULL,
 basis = c("observed", "scenario"), lang = c("en", "de"), ppv_levels = c(0.25, 0.5,
  0.75), f1_levels = c(0.2, 0.4, 0.6, 0.8), ppv_target = NULL, metric_x = "fpr",
  metric_y = "tpr", realised_rules = NULL, bins = 40)

classification_spec(data = NULL, truth = NULL, positive_label = NULL, scores = NULL,
  counts = NULL, theory = NULL, rule = NULL, cutoff = NULL, models = NULL,
 direction = NULL, comparator = NULL, score_scale = c("raw_score", "probability"),
  evaluation_split = NULL, negative_label = NULL, prevalence = NULL,
 prevalence_source = NULL, sampling = NULL, weights = NULL, weight_type = "frequency",
  cluster = NULL, id = NULL, missing = c("error", "exclude"),
  uncertainty = classification_uncertainty())

classification_table(x, table = c("metrics", "intervals", "summaries",
  "operating_point", "curve", "selection", "bounds", "tied",      "scenario",
  "calibration", "net_benefit", "comparisons", "limitations"), model_id = NULL,
  lang = c("en", "de"))

classification_uncertainty(level = 0.95, proportion_methods = c("wilson",
  "clopper_pearson"), auc_method = c("delong", "none"), bootstrap_replicates = 0,
  seed = NULL, bootstrap_strata = c("class", "none"), bootstrap_metrics = c("tpr",
  "fpr", "ppv", "npv", "youden_j", "f1", "mcc", "roc_auc", "average_precision"))

classifier_model(model_id, name = model_id, version = NULL, score_meaning = NULL,
  population = NULL, unit = NULL, source_hash = NULL)

compare_classifiers(spec, level = NULL)

cutoff_rule(cutoff = NULL, selection = NULL, selection_data = NULL, constraints = NULL,
  optimise = NULL, capacity = NULL, costs = NULL, tie_break = NULL,
  prevalence_basis = NULL)

cutoff_sweep(x, metrics = .ig_cls_default_sweep_metrics, model_id = NULL, grid = 401,
  basis = c("observed", "scenario"))

distribution_calculator(expression)

evaluate_cutoff(spec, cutoff = NULL, model_id = NULL, partial_auc = NULL,
  threshold_probability = NULL, calibration_bins = 10, include_curves = TRUE)

ig_status(code, reason = "", details = list())

ig_status_codes()

is_ig_status(x)
```

```
metric_dictionary(lang = c("en", "de"), scope = NULL)

net_benefit(spec, threshold_probability)

ppv_gap_max_prevalence(tpr, fpr)

pr_summary(spec, prevalence = NULL)

predictive_values_at_prevalence(tpr, fpr, prevalence)

prevalence_for_ppv_target(tpr, fpr, target)

prevalence_scenario(x, prevalence, n_total = NULL, source = c("assumed", "external"),
  lang = c("en", "de"))

select_cutoff(spec, objective = NULL, constraints = NULL, optimise = NULL,
  capacity = NULL, costs = NULL, tie_break = NULL, prevalence_basis = NULL, ...)

boschloo_inference(x1, n1, x2, n2, variant, alpha = NULL, groups = c("group 1",
  "group 2"), event = "event", control = boschloo_control())
```

Arguments

spec	A classification specification returned by classification_spec().
data	A data frame containing observations, with explicit truth and score columns.
auc	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.
sigma	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.
cutoff	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.
n1	Number of independent binomial observations in group 1.
n2	Number of independent binomial observations in group 2.
alpha	Decision level strictly between zero and one; NULL omits a decision where supported.
variant	Explicit Boschloo variant: minlike, central, less, or greater. Classifier variants are retained in their records.
p1	Event probability in group 1.
p2	Event probability in group 2.
fisher_convention	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.
control	Numerical settings from boschloo_control().

tie_tolerance	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.
grid_points	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.
nuisance_tol	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.
polish_tol	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.
max_levels	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.
ambiguity_band	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.
cache	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.
observed	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.
pvalues	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.
x	Input data, finalized result, or numeric values as specified by the selected function.
x1	Number of events in group 1.
x2	Number of events in group 2.
groups	Two distinct group labels in group 1, group 2 order.
event	Explicit event label.
bins	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.
partial_auc	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.
level	Confidence level strictly between zero and one.
count_kind	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.
lang	Display-label language, en or de.
panel	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.
model_id	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.
basis	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.
ppv_levels	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.
f1_levels	Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

ppv_target Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

metric_x Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

metric_y Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

realised_rules Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

truth Name of the observed truth column.

positive_label Explicit positive-class value; never inferred.

scores Score column name or named score-column vector.

counts Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

theory Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

rule Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

models Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

direction Declared score direction, higher_is_positive or lower_is_positive.

comparator Declared cutoff comparison, consistent with direction.

score_scale Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

evaluation_split
Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

negative_label Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

prevalence Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

prevalence_source
Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

sampling Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

weights Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

weight_type Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

cluster Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

`id` Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

`missing` Explicit missingness policy; see the preserved specification settings.

`uncertainty` Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

`table` Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

`proportion_methods` Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

`auc_method` Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

`bootstrap_replicates` Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

`seed` Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

`bootstrap_strata` Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

`bootstrap_metrics` Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

`name` Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

`version` Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

`score_meaning` Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

`population` Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

`unit` Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

`source_hash` Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

`selection` Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

`selection_data` Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

`constraints` Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

`optimise` Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

`capacity` Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

`costs` Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

`tie_break` Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

`prevalence_basis` Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

`metrics` Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

`grid` Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

`expression` Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

`threshold_probability` Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

`calibration_bins` Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

`include_curves` Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

`code` Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

`reason` Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

`details` Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

`scope` Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

`tpr` Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

`fpr` Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

`target` Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

`n_total` Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

`source` Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

`objective` Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

`...` Procedure-specific input, setting, or metadata field. Its accepted values are validated by the function; retained settings are included in the result.

Details

Boschloo tests use Fisher ordering and bounded nuisance maximization. The central variant doubles the smaller separately maximized one-sided value. Rejection uses the numerical upper bound; this bound is not a sampling confidence interval. The shared Boschloo inference path requires only standard R. Historical hashed specifications and the legacy calculator decimal parser require the optional packages digest and/or jsonlite and give an explicit missing-dependency error when absent.

Classifier specifications preserve the positive class, direction, cutoff origin, missing-value exclusions, evaluation split and raw-data fingerprint. Raw observations are not retained in the common reporting payload. Basic count metrics, binormal rates and static dictionary access do not require contributed packages. Undefined metrics retain statuses and denominator reasons.

Provider functions are public interfaces for validated package consumers. Historical canonical primitives accept already converted plain data; guidance-owned schema dispatch remains in inferguidR. No provider evaluates code from input strings. Calculator expressions use a restricted parser.

The inherited validation metadata records prior scoped checks; package extraction and adapter parity do not certify all scientific designs.

Value

Computational functions return typed lists or ordinary data frames; common adapters return inference_result objects. Metadata providers return plain lists or vectors, probability kernels return numeric values, and validators signal typed errors for invalid inputs.

Examples

```
r <- boschloo_inference(1, 5, 4, 7, variant = "less")
result_data(r)
classification_metrics(c(tp = 8, fn = 2, fp = 3, tn = 7))
```

`method_spec` *Prespecify and execute an observed statistical method*

Description

Use an explicit method identity, sidedness, confidence level and method-specific options. Method selection is never based on observed pretests.

Usage

```
method_spec(method_id, alternative = "two.sided", conf.level = 0.95,
  missing = "error", options = list(), engine = "base")
method_catalogue()
run_inference(spec, ..., result_id = NULL)
```

Arguments

method_id	An identity from method_catalogue().
alternative	Prespecified two.sided, less or greater alternative.
conf.level	Confidence level strictly between zero and one.
missing	Explicit error or omit policy, subject to each procedure's supported scope.
options	Uniquely named list of supported procedure settings. Unknown settings are errors.
engine	"base" (default) computes with infercalcR's own implementation. "reference" computes with the validated original package listed in the reference column of method_catalogue(); it is an error if that package is not installed, and no other engine is used instead. Methods whose test is computed by R's stats package, or whose model is fitted by stats or an optional model backend, have a single engine, which either value runs. The engine used is recorded in result_metadata(x)$engine.
spec	A method_spec.
result_id	Optional explicit stable identifier for joining distinct results.
...	Observations and design arguments for the selected method.

Value

A specification, capability data frame, or finalized common result respectively.

Examples

```
run_inference(method_spec("descriptive_summary"), data = data.frame(x = 1:5))
```

---

observed-methods *Explicit observed methods and finalized numerical results*

Description

Execute a prespecified procedure through method_spec() and run_inference(). Each procedure retains its existing finite-sample, numerical, missingness and backend policies.

Details

Wilcoxon intervals retain the native-reported confidence level. Older R versions can record a nominal or rounded level whose exact attainment differs; newer versions can return a fully unbounded conservative interval. The common interval-method label and metadata identify this convention. No level, endpoint, or captured native diagnostic is replaced.

Arguments listed below are supplied by name to run_inference(); method controls may instead be declared in the named options list in method_spec(). Alternative and confidence level belong in the specification. An optional result_id argument to run_inference() supplies an explicit external result key for joins. Default IDs are stable only within one object, so supply distinct IDs when joining separate results. For classifier-template views, native payloads omit the original

specification and attached source result; finalized aggregate fields remain, with the raw-data-free classifier conversion payload stored separately. The legacy teaching wrappers continue to use public classifier APIs for their original specification shape. The method never selects an alternative or a test from observed diagnostic p-values. `result_data()` returns explicit effect, coefficient, omnibus or descriptive rows. `result_metadata()` records method identity, settings, population and source status. `result$payload$native` retains the complete finalized native output, including quantities that do not fit a scalar row. Numeric p-value bounds and Monte Carlo intervals are separate from sampling intervals.

For paired and repeated inputs, use the documented wide data with a confirmed unique subject identifier. Missing values are rejected unless that procedure's explicit complete-case, MAR, MCAR or pairwise policy is selected. Setting `missing = "omit"` alone never selects a missingness assumption. Unknown sample sizes remain unknown. Group, event, contrast and sign conventions follow the named procedure and its explicit inputs.

The base/standard-R procedures require no contributed calculation package. Gower/PAM uses **cluster**; Gaussian mixed models use **nlme**; NB2 and ordinal models use **MASS**; multinomial models use **nnet**; logistic random-intercept models use **lme4**; exchangeable GEE uses **geepack**. Missing backends produce an explicit error and never a statistical substitute. Legacy classifier objects may use the separately documented canonical JSON/fingerprint dependencies.

The binary logistic procedure retains profile-likelihood slope intervals and separate Wald columns. Generic fitted-glm conversion instead explicitly offers Wald intervals. Factorial ANOVA retains raw and Holm-adjusted term p-values for the declared whole family. ANCOVA intervals and contrasts are marginal and unadjusted. Rank location intervals are not general median-difference intervals. Prediction-development bootstraps refit preprocessing and preserve failed replicates; no external-validation claim follows. Structure/clustering outputs are exploratory and carry no fabricated hypothesis test.

`variance_chisq_one_sample` is an advisor-only identity and gives an unsupported condition. The following 68 implemented identities retain these native input signatures (`alternative` and `conf.level`, when shown, are supplied through the specification):

- `ancova_baseline_adjusted`: `data`, `covariates`, `group_levels`, `common_slopes`, `conf.level`, `missing`, `alternative`.
- `anova_factorial`: `data`, `factor_columns`, `independent_units`, `equal_variance_justified`, `alternative`, `conf.level`, `complete_cases`.
- `anova_one_way_welch`: `x`, `group`, `alternative`, `conf.level`.
- `anova_one_way`: `x`, `group`, `alternative`, `conf.level`.
- `anova_repeated_one_way_gg`: `data`, `occasion_columns`, `independent_subjects`, `alternative`, `conf.level`, `complete_cases`.
- `binomial_exact_one_sample`: `events`, `n`, `p0`, `alternative`, `conf.level`.
- `boschloo_one_sided`: `table`, `alternative`, `conf.level`.
- `boschloo_two_sided_central`: `table`, `alternative`, `conf.level`.
- `boschloo_two_sided_minlike`: `table`, `alternative`, `conf.level`.
- `bowker_symmetry`: `counts`, `alternative`, `conf.level`, `n_excluded`.
- `brunner_munzel`: `x`, `y`, `alternative`, `conf.level`.
- `chisq_goodness_of_fit_mc`: `observed`, `probabilities`, `seed`, `B`, `conf.level`, `mc.level`.

- chisq_goodness_of_fit: counts, p, alternative, conf.level.
- chisq_independence_rxc: counts, alternative, conf.level.
- classifier_binormal_theory: auc, sigma, cutoff, prevalence, n_total.
- classifier_cutoff_exploration: data, direction, comparator.
- classifier_heldout_validation: tuning, evaluation, direction, comparator, independent_splits, conf.level.
- classifier_predeclared_rule: data, cutoff, direction, comparator, predeclared, conf.level.
- cluster_gower_pam: data, variables, types, weights, k, independent_units, alternative, conf.level, missing, B, seed.
- cluster_hierarchical_ward: data, variables, k, standardize, independent_units, alternative, conf.level, complete_cases, B, seed.
- cluster_kmeans: data, variables, k, standardize, independent_units, alternative, conf.level, complete_cases, B, seed.
- cochran_armitage_trend: events, totals, scores, scores_prespecified, alternative, conf.level.
- cochran_q: data, alternative, conf.level, missing.
- correlation_kendall: data, alternative, conf.level.
- correlation_pearson: data, alternative, conf.level.
- correlation_spearman: data, alternative, conf.level.
- descriptive_summary: x, measurement, conf.level, event.
- diagnostic_2x2_fixed_cutoff: tp, fn, fp, tn, sampling, cutoff_origin, evaluation_split, conf.level, prevalence, prevalence_source.
- efa_common_factor_ml: data, variables, factors, independent_units, continuous_measurements, alternative, conf.level, complete_cases.
- fisher_exact_2x2: table, alternative, conf.level.
- fisher_exact_rxc: counts, alternative, conf.level.
- friedman: data, alternative, conf.level, missing.
- gee_logistic_exchangeable: data, independent_clusters, model_assumptions_justified, alternative, conf.level, missing_outcomes.
- glmm_logistic_random_intercept: data, independent_clusters, model_assumptions_justified, alternative, conf.level, missing_outcomes.
- jonckheere_terpstra: data, group_order, order_prespecified, independent_units, seed, alternative, B, conf.level, mc.level, complete_cases.
- kruskal_wallis: x, group, alternative, conf.level.
- lmm_cluster_random_intercept: data, independent_clusters, random_intercept_justified, alternative, conf.level, missing_outcomes.
- lmm_repeated_measures: data, independent_clusters, random_intercept_justified, alternative, conf.level, missing_outcomes.
- mcnemar_asymptotic_cc: counts, alternative, conf.level.

- mcnemar_asymptotic_uncorrected: counts, alternative, conf.level.
- mcnemar_exact_conditional: counts, alternative, conf.level.
- mcnemar_midp: counts, alternative, conf.level.
- pca_components: data, variables, components, standardise, missing.
- prediction_model_linear: data, predictors, independent_units, alternative, conf.level, B, seed, complete_cases, resamples.
- prediction_model_logistic: data, predictors, independent_units, alternative, conf.level, B, seed, complete_cases, resamples.
- prop_one_sample_score: events, n, p0, alternative, conf.level.
- prop_two_independent_pearson_chisq: counts, alternative, conf.level.
- rank_sum_mann_whitney: x, y, alternative, conf.level, exact, location_shift.
- regression_linear_multiple: data, predictors, tested_predictors, alternative, conf.level, missing.
- regression_linear_simple: data, predictor, alternative, conf.level, missing.
- regression_logistic_binary: data, predictors, independent_units, alternative, conf.level, complete_cases.
- regression_multinomial: data, predictors, independent_units, reference_category, alternative, conf.level, complete_cases.
- regression_negative_binomial: data, predictors, independent_units, alternative, conf.level, complete_cases.
- regression_ordinal_proportional_odds: data, predictors, independent_units, proportional_odds_just alternative, conf.level, complete_cases.
- regression_poisson: data, predictors, independent_units, alternative, conf.level, complete_cases.
- sign_test_one_sample: x, mu0, alternative, conf.level.
- sign_test_paired: data, alternative, conf.level.
- signed_rank_wilcoxon_one_sample: x, mu, alternative, conf.level, exact.
- signed_rank_wilcoxon: data, alternative, conf.level, exact.
- stuart_maxwell_marginal_homogeneity: counts, alternative, conf.level, n_excluded.
- t_mean_one_sample: x, mu0, alternative, conf.level.
- t_means_paired: data, alternative, conf.level.
- t_means_two_independent_student: x, y, alternative, conf.level.
- t_means_two_independent_welch: x, y, alternative, conf.level.
- unconditional_exact_z_pooled: table, alternative, conf.level, independent_binomial_groups, tolerance, max_subdivisions.
- variance_brown_forsythe: x, group, alternative, conf.level.
- variance_f_test_two: x, y, ratio0, alternative, conf.level.
- variance_fligner_killeen: x, group, alternative, conf.level.

Exact native argument defaults and scope constraints are also recorded in 'methods/implemented.csv' and the package source. Source templates are never evaluated at runtime.

Thirteen methods have two engines. engine = "base" (the default) uses infercalcR's own implementation, written from the method's publication; engine = "reference" calls the validated original: the function DescrTab2 uses for the test, Exact for Boschloo's test, or the established package for tests DescrTab2 does not offer. method_catalogue() lists each method's engines and original. The engines are benchmarked against each other on the originals' help-page examples ('methods/engine-benchmarks.rds', approved exceptions in 'methods/engine-deviations.csv'). 'methods/provenance.csv' records for every method its validated source, its original, the publication its base implementation follows and the outcome of the check that no base implementation was adapted from the code of a GPL-licensed original.

Value

run_inference() returns a common inference_result, or a descriptive_result or classification_result where appropriate. No reporting package is required.

References

Armitage P (1955). Tests for linear trends in proportions and frequencies. *Biometrics* 11(3):375–386. doi:10.2307/3001775

Boschloo RD (1970). Raised conditional level of significance for the 2 x 2-table when testing the equality of two probabilities. *Statistica Neerlandica* 24(1):1–9. doi:10.1111/j.14679574.1970.tb00104.x

Box GEP (1954). Some theorems on quadratic forms applied in the study of analysis of variance problems, I. *Annals of Mathematical Statistics* 25(2):290–302. doi:10.1214/aoms/1177728786

Brown MB, Forsythe AB (1974). Robust tests for the equality of variances. *Journal of the American Statistical Association* 69(346):364–367. doi:10.1080/01621459.1974.10482955

Brunner E, Munzel U (2000). The nonparametric Behrens-Fisher problem: asymptotic theory and a small-sample approximation. *Biometrical Journal* 42(1):17–25. doi:10.1002/(SICI)1521-4036(200001)42:1<17::AIDBIMJ17>3.0.CO;2U

Clopper CJ, Pearson ES (1934). The use of confidence or fiducial limits illustrated in the case of the binomial. *Biometrika* 26(4):404–413. doi:10.1093/biomet/26.4.404

Cochran WG (1950). The comparison of percentages in matched samples. *Biometrika* 37(3-4):256–266. doi:10.1093/biomet/37.34.256

Cochran WG (1954). Some methods for strengthening the common chi-squared tests. *Biometrics* 10(4):417–451. doi:10.2307/3001616

Fagerland MW, Lydersen S, Laake P (2013). The McNemar test for binary matched-pairs data: mid-p and asymptotic are better than exact conditional. *BMC Medical Research Methodology* 13:91. doi:10.1186/147122881391

Greenhouse SW, Geisser S (1959). On methods in the analysis of profile data. *Psychometrika* 24(2):95–112. doi:10.1007/BF02289823

Jonckheere AR (1954). A distribution-free k-sample test against ordered alternatives. *Biometrika* 41(1-2):133–145. doi:10.1093/biomet/41.12.133

Lancaster HO (1961). Significance tests in discrete distributions. *Journal of the American Statistical Association* 56(294):223–234. doi:10.1080/01621459.1961.10482105

Maxwell AE (1970). Comparing the classification of subjects by two independent judges. *British Journal of Psychiatry* 116(535):651–655. doi:10.1192/bjp.116.535.651

McNemar Q (1947). Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika* 12(2):153–157. doi:10.1007/BF02295996

Phipson B, Smyth GK (2010). Permutation p-values should never be zero: calculating exact p-values when permutations are randomly drawn. *Statistical Applications in Genetics and Molecular Biology* 9(1):39. doi:10.2202/15446115.1585

Stuart A (1955). A test for homogeneity of the marginal distributions in a two-way classification. *Biometrika* 42(3-4):412–416. doi:10.1093/biomet/42.34.412

Suissa S, Shuster JJ (1985). Exact unconditional sample sizes for the 2 x 2 binomial trial. *Journal of the Royal Statistical Society, Series A* 148(4):317–327. doi:10.2307/2981892

Terpstra TJ (1952). The asymptotic normality and consistency of Kendall's test against trend, when ties are present in one ranking. *Indagationes Mathematicae (Proceedings)* 55:327–333. doi:10.1016/S13857258(52)50043X

Examples

```
x <- c(2, 4, 5, 7, 9)
y <- c(1, 3, 4, 6, 8, 10)
spec <- method_spec("t_means_two_independent_welch")
result_data(run_inference(spec, x = x, y = y))
```

`parallel_analysis` *Common-factor exploration and prediction from stored development results*

Description

Two explicit computational helpers retained from existing procedures.

Usage

```
parallel_analysis(data, variables, independent_units = FALSE,
  continuous_measurements = FALSE, B = 199L, seed = 104729L,
  quantile_probability = .95, complete_cases = FALSE, permutations = NULL)
predict_inference(result, newdata, interval = "none", conf.level = NULL)
```

Arguments

`data`	Data frame with unique `id` and numeric measurement columns.
`variables`	Explicit measurement column names.
`independent_units`, `continuous_measurements`	Required declarations for the permutation design.
`B`	Prespecified number of replicates, subject to the method's computational limits.
`seed`	Fixed reproducible seed; caller RNG state is restored.

quantile_probability
: Prespecified null-spectrum quantile, using quantile type 6.

complete_cases
: Explicit complete-case choice; missingness assumptions remain the analyst's responsibility.

permutations
: Optional validated column permutation indices.

result
: A finalized prediction_model_linear or prediction_model_logistic result.

newdata
: New numeric predictor values matching the fitted model.

interval
: "none"; for linear predictions "mean" or "prediction"; for logistic predictions "risk".

conf.level
: Confidence level. NULL retains the development result's requested level.

Details

Parallel analysis replaces the correlation diagonal by squared multiple correlations and uses independent column permutations. It supplies exploratory leading-factor guidance without fitting or selecting an ML factor model. Failed replicates are retained and prevent a selected count.

Linear "mean" intervals are pointwise normal-model t confidence intervals for the conditional mean. Linear "prediction" intervals instead cover one independent new outcome and include new-outcome residual variation. Common-row estimands and interval-method labels distinguish these targets; metadata also records the interval target, type and native interpretation. Point predictions without requested intervals have no interval scope. Logistic risk intervals use already fitted bootstrap models and type-6 percentile limits, requiring at least five expected replicates per tail. They concern risk estimation, not a binary outcome, validation performance, or Monte Carlo precision. No model is refitted by predict_inference().

Value

A common result with finalized rows, metadata and full native helper output in payload$native.

Examples

```
# Parallel analysis of four measurements, three sharing one common factor
set.seed(1)
n <- 60
f <- rnorm(n)
dat <- data.frame(id = seq_len(n),
  v1 = f + rnorm(n, sd = 0.5), v2 = f + rnorm(n, sd = 0.5),
  v3 = f + rnorm(n, sd = 0.5), v4 = rnorm(n))
pa <- parallel_analysis(dat, c("v1", "v2", "v3", "v4"),
  independent_units = TRUE, continuous_measurements = TRUE, B = 100L)
result_data(pa)[, c("estimand", "estimate", "status")]

# Predictions with intervals from a finalized linear development result
x1 <- rep(-4:3, 5)
dev <- data.frame(id = seq_along(x1), x1 = x1, x2 = rep(c(0, 1), each = 20),
  y = 1 + 2 * x1 + rep(c(-1, 0, 1, 0.5, -0.5), 8))
fit <- run_inference(method_spec("prediction_model_linear"), data = dev,
  predictors = c("x1", "x2"), independent_units = TRUE, B = 200L)
```

```
pred <- predict_inference(fit, data.frame(x1 = c(0, 2), x2 = c(0, 1)),
  interval = "prediction")
result_data(pred)[, c("estimate", "conf.low", "conf.high", "status")]
```

`result-contract` *Finalized statistical result contract*

Description

Construct, validate and access plain-data statistical results. Numerical inference is finalized before presentation.

Usage

```
new_inference_result(data, metadata, kind = "inference", payload = list())
validate_inference(x)
result_data(x, ...)
result_metadata(x, ...)
as_inference(x, ...)
adjust_inference(x, method = "holm", family_id, rows = NULL)
```

Arguments

`data`	A data frame of finalized numerical rows, with unique row IDs.
`metadata`	A named list including method_id; optional variant_id, result_id, provenance, population, settings and validation evidence.
`kind`	One of inference, planning, classification or descriptive.
`payload`	Method-specific finalized quantities that do not fit scalar columns. No raw data are required.
`x`	An existing result.
`...`	Arguments to a documented adapter or accessor method.
`method`	A method supported by `stats::p.adjust`.
`family_id`	A unique identifier that has not already been used in this result. An explicit identifier of the complete declared hypothesis family.
`rows`	Unique row IDs comprising the complete family; all rows by default.

Details

Rows distinguish estimate, statistic, degrees of freedom, confidence limits and their level/method/scale/scope, raw p.value, p.adjusted and p.reported, family metadata, null/alternative, sample and exclusion counts, status and reason. NA means unavailable; legitimate unbounded confidence limits remain infinite. A numerical p-value bound is a separate method-specific quantity and is never a sampling confidence interval.

A finalized `conf.level` is NA when unavailable or a finite attained/reported level in `(0, 1]`. Conservative discrete intervals can attain level 1, including intervals with infinite endpoints. The

native-reported level, endpoints and diagnostics are preserved. Requested confidence levels in `method_spec()` remain strictly between zero and one; the requested level must not replace a different native-reported level. For Wilcoxon intervals, older R versions may report nominal or rounded confidence attributes; the legacy payload field `attained_confidence_level` preserves that attribute verbatim. Common interval labels identify the convention, and imported objects have unknown originating R provenance.

Adjustment rejects missing raw p-values and already adjusted families. It is performed once on the declared complete family before display filtering. It does not adjust confidence intervals. Results retain computation status separately from validation evidence. Result IDs are stable labels, not cryptographic hashes. RDS round trips are suitable only for trusted objects.

Value

Constructors and adapters return an S3 inference_result; result_data returns a data.frame, result_metadata a list. Validation returns the object invisibly.

Examples

```
x <- new_inference_result(data.frame(row_id = c("a", "b"),
  p.value = c(.01, .04)), list(method_id = "example"))
result_data(adjust_inference(x, family_id = "primary"))
```

Index

adjust_inference *(result-contract)*, 25
as_inference *(result-contract)*, 25
as_inference.glm *(as_inference.htest)*, 2
as_inference.htest, 2
as_inference.lm *(as_inference.htest)*, 2
attach_classification_data *(legacy-computation)*, 10

binomial_probability *(kernel-providers)*, 5
binomial_region_probability *(kernel-providers)*, 5
binomial_table_space *(kernel-providers)*, 5
binormal_model *(legacy-computation)*, 10
binormal_rates *(legacy-computation)*, 10
boschloo_compare_fisher *(legacy-computation)*, 10
boschloo_control *(legacy-computation)*, 10
boschloo_control_call *(kernel-providers)*, 5
boschloo_control_from_record *(kernel-providers)*, 5
boschloo_control_record *(kernel-providers)*, 5
boschloo_design_kernel *(kernel-providers)*, 5
boschloo_engine_info *(kernel-providers)*, 5
boschloo_inference *(legacy-computation)*, 10
boschloo_kernel_status *(kernel-providers)*, 5
boschloo_plot_data_record *(kernel-providers)*, 5
boschloo_plotdata_lattice *(legacy-computation)*, 10
boschloo_plotdata_nuisance *(legacy-computation)*, 10
boschloo_region *(legacy-computation)*, 10
boschloo_region_record *(kernel-providers)*, 5
boschloo_test *(legacy-computation)*, 10
boschloo_validate_control *(kernel-providers)*, 5
boschloo_validate_design *(kernel-providers)*, 5
boschloo_variant_details *(kernel-providers)*, 5
boschloo_variant_record *(kernel-providers)*, 5
boschloo_variants *(legacy-computation)*, 10

calculator_functions *(legacy-computation)*, 10
calibration_summary *(legacy-computation)*, 10
chisq_probability *(kernel-providers)*, 5
chisq_quantile *(kernel-providers)*, 5
classification_auc *(legacy-computation)*, 10
classification_check_bins *(kernel-providers)*, 5
classification_check_partial *(kernel-providers)*, 5
classification_check_thresholds *(kernel-providers)*, 5
classification_contract *(kernel-providers)*, 5
classification_dictionary *(kernel-providers)*, 5
classification_evaluated_spec *(legacy-computation)*, 10
classification_metric_ids *(kernel-providers)*, 5
classification_metrics *(legacy-computation)*, 10

classification_plot_data
(legacy-computation), 10
classification_reason
(kernel-providers), 5
classification_short_label
(kernel-providers), 5
classification_spec
(legacy-computation), 10
classification_table
(legacy-computation), 10
classification_uncertainty
(legacy-computation), 10
classifier_model (legacy-computation),
10
compare_classifiers
(legacy-computation), 10
cutoff_rule (legacy-computation), 10
cutoff_sweep (legacy-computation), 10

describe_data, 4
distribution_calculator
(legacy-computation), 10

evaluate_cutoff (legacy-computation), 10

f_lower_log_bound (kernel-providers), 5
f_probability (kernel-providers), 5
f_quantile (kernel-providers), 5
fisher_ordering_probabilities
(kernel-providers), 5

ig_status (legacy-computation), 10
ig_status_codes (legacy-computation), 10
invert_log_cdf (kernel-providers), 5
is_ig_status (legacy-computation), 10

kernel-providers, 5

legacy-computation, 10
legacy_canonical_json
(kernel-providers), 5
legacy_check_spec (kernel-providers), 5
legacy_double_values
(kernel-providers), 5
legacy_format_double
(kernel-providers), 5
legacy_hash_value (kernel-providers), 5
legacy_json_limits (kernel-providers), 5
legacy_nonfinite_paths
(kernel-providers), 5
legacy_parse_decimal
(kernel-providers), 5
legacy_parse_json (kernel-providers), 5
legacy_r_literal (kernel-providers), 5
legacy_sha256 (kernel-providers), 5
legacy_unclass_frozen
(kernel-providers), 5
log_probability_add (kernel-providers),
5
log_probability_sum (kernel-providers),
5

method_catalogue (method_spec), 17
method_spec, 17
metric_dictionary (legacy-computation),
10

net_benefit (legacy-computation), 10
new_inference_result (result-contract),
25
noncentral_t_density
(kernel-providers), 5
normal_probability (kernel-providers), 5
normal_quantile (kernel-providers), 5

observed-methods, 18

parallel_analysis, 23
ppv_gap_max_prevalence
(legacy-computation), 10
pr_summary (legacy-computation), 10
predict_inference (parallel_analysis),
23
predictive_values_at_prevalence
(legacy-computation), 10
prevalence_for_ppv_target
(legacy-computation), 10
prevalence_scenario
(legacy-computation), 10

result-contract, 25
result_data (result-contract), 25
result_metadata (result-contract), 25
run_inference (method_spec), 17

select_cutoff (legacy-computation), 10

t_probability (kernel-providers), 5
t_quantile (kernel-providers), 5

validate_inference (result-contract), 25