Package 'ReproStat'

May 7, 2026

Type Package

Title Reproducibility Diagnostics for Statistical Modeling

Version 0.1.2

Date 2026-03-25

Description Tools for diagnosing the reproducibility of statistical model outputs under data perturbations. Implements bootstrap, subsampling, and noise-based perturbation schemes and computes coefficient stability, p-value stability, selection stability, prediction stability, and a composite reproducibility index on a 0 to 100 scale. Includes cross-validation ranking stability for model comparison and visualization utilities. Optional 'backends' support robust M-estimation ('MASS') and penalized regression ('glmnet'). Bootstrap perturbation follows 'Efron' and 'Tibshirani' (1993, ISBN:9780412042317); selection stability follows 'Meinshausen' and 'Buhlmann' (2010) <doi:10.1111/j.1467-9868.2010.00740.x>; reproducibility framework follows 'Peng' (2011) <doi:10.1126/science.1213847>.

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.3.2

Config/testthat/edition 3

Imports stats, graphics

Suggests testthat (>= 3.0.0), knitr, rmarkdown, MASS (>= 7.3), glmnet (>= 4.0), ggplot2 (>= 3.0)

VignetteBuilder knitr

URL https://ntigideon.github.io/ReproStat/, https://github.com/ntiGideon/ReproStat

BugReports https://github.com/ntiGideon/ReproStat/issues

NeedsCompilation no

Author Gideon Nti Boateng [aut, cre]

Maintainer Gideon Nti Boateng <gidiboateng200@gmail.com>

Repository CRAN
Date/Publication 2026-03-30 17:10:02 UTC

Contents

`coef_stability` *Coefficient stability*

Description

Computes the variance of each regression coefficient across perturbation iterations. Lower variance indicates greater stability.

Usage

```
coef_stability(diag_obj)
```

Arguments

`diag_obj`	A `reprostat` object from `run_diagnostics`.

Value

A named numeric vector of per-coefficient variances.

Examples

```
set.seed(1)
d <- run_diagnostics(mpg ~ wt + hp, data = mtcars, B = 50)
coef_stability(d)
```

cv_ranking_stability *Cross-validation ranking stability*

Description

Evaluates model selection stability by repeatedly running K-fold cross-validation and recording the error-metric rank of each candidate model across repetitions. Supports four modeling backends: `"lm"`, `"glm"`, `"rlm"` (robust regression via **MASS**), and `"glmnet"` (penalized regression via **glmnet**).

Usage

```
cv_ranking_stability(
  formulas,
  data,
  v = 5L,
  R = 30L,
  seed = 20260307L,
  family = NULL,
  backend = c("lm", "glm", "rlm", "glmnet"),
  en_alpha = 1,
  lambda = NULL,
  metric = c("auto", "rmse", "logloss")
)
```

Arguments

`formulas`	A named list of formulas, one per candidate model.
`data`	A data frame.
`v`	Number of cross-validation folds. Must satisfy $2 \leq v \leq n$. Default is `5`.
`R`	Number of cross-validation repetitions. Default is `30`.
`seed`	Integer random seed for reproducibility. Default is `20260307`.
`family`	A GLM family object (e.g. `stats::binomial()`) or `NULL` (default) to use `stats::lm`. Used only with `backend = "glm"`.
`backend`	Modeling backend. One of `"lm"` (default), `"glm"`, `"rlm"`, or `"glmnet"`.
`en_alpha`	Elastic-net mixing parameter for `glmnet` (default `1` for LASSO). Ignored for other backends.
`lambda`	Regularization parameter for `glmnet`. When `NULL` (default), selected per fold via `glmnet::cv.glmnet`. Ignored for other backends.
`metric`	Error metric used for ranking. One of `"auto"` (default: `"rmse"` for `lm`/`"rlm"`/`"glmnet"`, `"logloss"` for `glm` with a non-Gaussian family), `"rmse"`, or `"logloss"`. The `"logloss"` metric assumes a **binary** (0/1) response; it is not defined for multi-class outcomes.

Value

A list with components:

`settings` List with `v`, `R`, `seed`, `metric`, `family`, `backend`, `en_alpha`, and `lambda`.

`rmse_mat` $R \times M$ matrix of per-repeat mean error values (RMSE or log-loss depending on `metric`).

`rank_mat` $R \times M$ integer matrix of per-repeat model ranks (rank 1 = best).

`summary` Data frame with columns `model`, `mean_rmse`, `sd_rmse`, `mean_rank`, and `top1_frequency`, ordered by mean rank. Note: the columns `mean_rmse` and `sd_rmse` store the mean and standard deviation of the chosen error metric (RMSE or log-loss); the column names are retained for backwards compatibility.

Examples

```
# Linear models
models <- list(m1 = mpg ~ wt + hp, m2 = mpg ~ wt + hp + disp)
cv_ranking_stability(models, mtcars, v = 5, R = 20)

# Logistic models
glm_models <- list(m1 = am ~ wt + hp, m2 = am ~ wt + hp + qsec)
cv_ranking_stability(glm_models, mtcars, v = 5, R = 20,
                     family = stats::binomial(), metric = "logloss")

# Robust regression
if (requireNamespace("MASS", quietly = TRUE)) {
  cv_ranking_stability(models, mtcars, v = 5, R = 20, backend = "rlm")
}

# Penalized (LASSO)
if (requireNamespace("glmnet", quietly = TRUE)) {
  lasso_models <- list(m1 = mpg ~ wt + hp, m2 = mpg ~ wt + hp + disp + qsec)
  cv_ranking_stability(lasso_models, mtcars, v = 5, R = 20, backend = "glmnet")
}
```

`perturb_data` *Perturb a dataset*

Description

Generates a perturbed version of a dataset using one of three strategies: bootstrap resampling, subsampling without replacement, or Gaussian noise injection.

Usage

```
perturb_data(
  data,
  method = c("bootstrap", "subsample", "noise"),
```

```
  frac = 0.8,
  noise_sd = 0.05,
  response_col = NULL
)
```

Arguments

data	A data frame.
method	Character string specifying the perturbation method. One of "bootstrap" (default), "subsample", or "noise".
frac	Fraction of rows to retain for subsampling. Ignored for other methods. Must be in $(0, 1]$. Default is 0.8.
noise_sd	Noise level as a fraction of each column's standard deviation. Ignored unless method = "noise". Default is 0.05.
response_col	Optional character string naming the response (outcome) column to *exclude* from noise injection. Useful when you want to perturb predictors only and leave the outcome unchanged. Ignored for method = "bootstrap" and method = "subsample". When NULL (default) all numeric columns including the response receive noise.

Value

A data frame with the same columns as data. The number of rows equals nrow(data) for bootstrap and noise, and floor(frac * nrow(data)) for subsampling.

Examples

```
set.seed(1)
d_boot <- perturb_data(mtcars, method = "bootstrap")
d_sub  <- perturb_data(mtcars, method = "subsample", frac = 0.7)
d_nois <- perturb_data(mtcars, method = "noise", noise_sd = 0.1)

# Perturb predictors only, leave the response (mpg) unchanged:
d_pred_only <- perturb_data(mtcars, method = "noise",
                            noise_sd = 0.1, response_col = "mpg")
```

---

plot_cv_stability *Plot cross-validation ranking stability*

Description

Produces a bar chart of either the top-1 selection frequency or the mean CV rank for each candidate model.

Usage

```
plot_cv_stability(cv_obj, metric = c("top1_frequency", "mean_rank"))
```

Arguments

cv_obj	A list returned by cv_ranking_stability.
metric	Character string. One of "top1_frequency" (default) or "mean_rank".

Value

Invisibly returns NULL; called for its side effect.

Examples

```
models <- list(m1 = mpg ~ wt + hp, m2 = mpg ~ wt + hp + disp)
cv_obj <- cv_ranking_stability(models, mtcars, v = 5, R = 20)
plot_cv_stability(cv_obj, metric = "top1_frequency")
```

---

plot_cv_stability_gg *ggplot2-based CV ranking stability plot*

Description

Produces a **ggplot2** horizontal bar chart of either the top-1 frequency or the mean rank of each candidate model from a repeated cross-validation stability run.

Usage

```
plot_cv_stability_gg(cv_obj, metric = c("top1_frequency", "mean_rank"))
```

Arguments

cv_obj	A list returned by cv_ranking_stability.
metric	One of "top1_frequency" (default) or "mean_rank".

Value

A ggplot object.

Examples

```
if (requireNamespace("ggplot2", quietly = TRUE)) {
  models <- list(m1 = mpg ~ wt + hp, m2 = mpg ~ wt + hp + disp)
  cv <- cv_ranking_stability(models, mtcars, v = 5, R = 20)
  plot_cv_stability_gg(cv, "top1_frequency")
  plot_cv_stability_gg(cv, "mean_rank")
}
```

plot_stability *Plot stability diagnostics*

Description

Produces a bar chart or histogram summarising one stability dimension from a `reprostat` object.

Usage

```
plot_stability(
  diag_obj,
  type = c("coefficient", "pvalue", "selection", "prediction")
)
```

Arguments

`diag_obj`	A `reprostat` object from `run_diagnostics`.
`type`	Character string specifying the plot type. One of `"coefficient"` (default), `"pvalue"`, `"selection"`, or `"prediction"`.

Value

Invisibly returns `NULL`; called for its side effect.

Examples

```
set.seed(1)
d <- run_diagnostics(mpg ~ wt + hp, data = mtcars, B = 50)
plot_stability(d, "coefficient")
plot_stability(d, "selection")
```

plot_stability_gg *ggplot2-based stability plot*

Description

Produces a **ggplot2** bar chart for either the coefficient variance or the selection frequency of each predictor term, ordered from lowest to highest value.

Usage

```
plot_stability_gg(diag_obj, type = c("coefficient", "selection"))
```

Arguments

diag_obj	A "reprostat" object returned by run_diagnostics.
type	One of "coefficient" (default) or "selection".

Value

A ggplot object.

Examples

```
if (requireNamespace("ggplot2", quietly = TRUE)) {
  d <- run_diagnostics(mpg ~ wt + hp, mtcars, B = 50)
  plot_stability_gg(d, "coefficient")
  plot_stability_gg(d, "selection")
}
```

---

prediction_stability *Prediction stability*

Description

Computes the pointwise variance of predictions across perturbation iterations, and the mean of these variances as a scalar summary.

Usage

```
prediction_stability(diag_obj)
```

Arguments

diag_obj	A reprostat object from run_diagnostics.

Details

By default predictions are made on the training data used to fit the base model. For method = "subsample" this means the held-out rows receive genuine out-of-sample predictions, while for method = "bootstrap" the predictions are a mix of in-bag and out-of-bag. Pass predict_newdata to run_diagnostics for a dedicated held-out evaluation set.

Value

A list with components:

pointwise_variance Numeric vector of per-observation prediction variances.

mean_variance Mean of pointwise variances.

Examples

```
set.seed(1)
d <- run_diagnostics(mpg ~ wt + hp, data = mtcars, B = 50)
prediction_stability(d)$mean_variance
```

---

print.reprostat *Print a reprostat object*

Description

Print a reprostat object

Usage

```
## S3 method for class 'reprostat'
print(x, ...)
```

Arguments

x	A reprostat object.
...	Further arguments (ignored).

Value

Invisibly returns x.

Examples

```
set.seed(1)
d <- run_diagnostics(mpg ~ wt + hp, data = mtcars, B = 20)
print(d)
```

---

pvalue_stability *P-value stability*

Description

Computes the proportion of perturbation iterations in which each predictor is statistically significant (p-value below alpha). The intercept is excluded. Values near 0 or 1 indicate stable decisions; values near 0.5 indicate high instability.

Usage

```
pvalue_stability(diag_obj)
```

Arguments

diag_obj	A reprostat object from run_diagnostics.

Value

A named numeric vector of significance frequencies in $[0, 1]$, excluding the intercept. All NaN for backend = "glmnet" (p-values are not defined).

Examples

```
set.seed(1)
d <- run_diagnostics(mpg ~ wt + hp, data = mtcars, B = 50)
pvalue_stability(d)
```

---

reproducibility_index *Reproducibility index*

Description

Computes a composite reproducibility index (RI) on a 0–100 scale by aggregating normalized versions of coefficient, p-value, selection, and prediction stability.

Usage

```
reproducibility_index(diag_obj)
```

Arguments

diag_obj	A reprostat object from run_diagnostics.

Details

Each component is mapped to $[0, 1]$ as follows:

Coefficient component (C_β) $C_\beta = \frac{1}{p}\sum_{j=1}^{p} \exp\left(-S_{\beta,j}/(|\hat{\beta}_j^{(0)}| + \bar{\beta})\right)$, where $\bar{\beta} = \max(\text{median}_j|\hat{\beta}_j^{(0)}|,\ 10^{-4})$ is a global scale reference. This prevents the exponential from collapsing to zero for weakly-identified (near-zero) predictors. Includes all model terms (intercept and predictors).

P-value component (C_p) $C_p = \frac{1}{p'}\sum_{j \neq \text{intercept}} |2S_{p,j} - 1|$, where $S_{p,j}$ is the proportion of perturbation iterations in which term j is significant at level alpha and p' is the number of predictor terms. Values near 0 or 1 (consistent decisions) score high; 0.5 (random) scores zero. NA for backend = "glmnet".

Selection component (C_{sel}) For "lm", "glm", "rlm": the mean *sign consistency* across predictors — the proportion of perturbation iterations in which each predictor's estimated sign agrees with the base-fit sign. Captures stability of effect direction, which is distinct from significance stability (C_p). For "glmnet": the mean *non-zero selection frequency* — proportion of perturbation iterations in which each predictor's coefficient is non-zero. Always available (never NA). Excludes the intercept.

Prediction component (C_{pred}) $C_{\text{pred}} = \exp(-\bar{S}_{\text{pred}}/(\text{Var}(y) + \varepsilon))$. Prediction variance relative to outcome variance drives the decay.

The RI is $100 \times (C_\beta + C_p + C_{\text{sel}} + C_{\text{pred}})/k$, where k is the number of non-NA components. For `backend = "glmnet"`, C_p is NA so the RI is based on three components ($k = 3$): C_β, C_{sel}, and C_{pred}. All other backends contribute all four components ($k = 4$).

Comparability across backends: because `"glmnet"` uses three components and `"lm"`/`"glm"`/`"rlm"` use four, RI values are not directly comparable across backends.

Value

A list with components:

`index` Scalar RI on a 0–100 scale.

`components` Named numeric vector with four sub-scores: `coef`, `pvalue`, `selection`, `prediction`. `pvalue` is NA for `backend = "glmnet"`; `selection` is always available.

Examples

```
set.seed(1)
d <- run_diagnostics(mpg ~ wt + hp, data = mtcars, B = 50)
reproducibility_index(d)
```

---

ri_confidence_interval

Bootstrap confidence interval for the reproducibility index

Description

Estimates uncertainty in the RI by resampling the perturbation iterations already stored in a `reprostat` object (no additional model fitting).

Usage

```
ri_confidence_interval(diag_obj, level = 0.95, R = 1000L, seed = NULL)
```

Arguments

`diag_obj`	A `reprostat` object from `run_diagnostics`.
`level`	Confidence level, e.g. `0.95` for a 95% interval. Default is `0.95`.
`R`	Number of bootstrap resamples of the perturbation draws. Default is `1000`. Values of 300–500 are sufficient for most uses.
`seed`	Integer random seed passed to `set.seed`, or `NULL` (default) to leave the global RNG state undisturbed. Pass an integer for fully reproducible intervals.

Value

A named numeric vector of length 2 giving the lower and upper quantile bounds of the RI bootstrap distribution.

Examples

```
set.seed(1)
d <- run_diagnostics(mpg ~ wt + hp, data = mtcars, B = 50)
ri_confidence_interval(d, R = 200, seed = 1)
```

---

run_diagnostics *Run reproducibility diagnostics*

Description

Fits a model on the original data, then repeatedly fits on perturbed versions to collect coefficient estimates, p-values, and predictions for downstream stability analysis. Four modeling backends are supported: ordinary least squares (`"lm"`), generalized linear models (`"glm"`), robust regression (`"rlm"` via **MASS**), and penalized regression (`"glmnet"` via **glmnet**).

Usage

```
run_diagnostics(
  formula,
  data,
  B = 200,
  method = c("bootstrap", "subsample", "noise"),
  alpha = 0.05,
  frac = 0.8,
  noise_sd = 0.05,
  predict_newdata = NULL,
  family = NULL,
  backend = c("lm", "glm", "rlm", "glmnet"),
  en_alpha = 1,
  lambda = NULL,
  perturb_response = FALSE
)
```

Arguments

`formula`	A model formula.
`data`	A data frame.
`B`	Integer. Number of perturbation iterations. Default is `200`.
`method`	Perturbation method passed to `perturb_data`. One of `"bootstrap"` (default), `"subsample"`, or `"noise"`.

`alpha` Significance threshold for p-value and selection stability. Default is `0.05`.

`frac` Subsampling fraction. Passed to `perturb_data`. Default is `0.8`.

`noise_sd` Noise level. Passed to `perturb_data`. Default is `0.05`.

`predict_newdata` Optional data frame for out-of-sample prediction stability. Defaults to `data`.

`family` A GLM family object (e.g. `stats::binomial()`, `stats::poisson()`). Used only when `backend = "glm"` (or when `family` is non-`NULL` and `backend = "lm"`, in which case the backend is silently promoted to `"glm"`).

`backend` Modeling backend. One of `"lm"` (default), `"glm"`, `"rlm"` (robust M-estimation via `MASS::rlm`), or `"glmnet"` (penalized regression via `glmnet::glmnet`).

`en_alpha` Elastic-net mixing parameter passed to `glmnet::glmnet`: `1` (default) gives the LASSO, `0` gives ridge, intermediate values give elastic net. Ignored for other backends.

`lambda` Regularization parameter for `glmnet`. When `NULL` (default) the penalty is selected by `glmnet::cv.glmnet` using `lambda.min`. Ignored for other backends.

`perturb_response` Logical. When `FALSE` (default) and `method = "noise"`, noise is added only to predictor columns; the response variable is left unchanged. Set to `TRUE` to also perturb the response (e.g. to simulate measurement error in the outcome). Has no effect for `method = "bootstrap"` or `"subsample"`, where row-level resampling naturally carries the response along.

Value

An object of class `"reprostat"`, a list with components:

`call` The matched call.

`formula` The model formula.

`B` Number of iterations used.

`alpha` Significance threshold used.

`base_fit` Model fitted on the original data.

`y_train` Response vector from the original data (used internally for RI computation).

`coef_mat` B x p matrix of coefficient estimates.

`p_mat` B x p matrix of p-values, or all-`NA` for `backend = "glmnet"` (where p-values are not defined).

`pred_mat` n x B matrix of predictions.

`method` Perturbation method used.

`family` GLM family used, or `NULL`.

`backend` Backend used.

`en_alpha` Elastic-net mixing parameter used (only relevant for `backend = "glmnet"`, otherwise `NA`).

Examples

```
set.seed(1)
# Linear model (small B for a quick check)
diag_lm <- run_diagnostics(mpg ~ wt + hp, data = mtcars, B = 20)
print(diag_lm)


# Logistic regression
diag_glm <- run_diagnostics(am ~ wt + hp + qsec, data = mtcars, B = 50,
                            family = stats::binomial())
reproducibility_index(diag_glm)

# Robust regression (requires MASS)
if (requireNamespace("MASS", quietly = TRUE)) {
  diag_rlm <- run_diagnostics(mpg ~ wt + hp, data = mtcars, B = 50,
                              backend = "rlm")
  reproducibility_index(diag_rlm)
}

# Penalized regression / LASSO (requires glmnet)
if (requireNamespace("glmnet", quietly = TRUE)) {
  diag_glmnet <- run_diagnostics(mpg ~ wt + hp + disp + qsec, data = mtcars,
                                 B = 50, backend = "glmnet", en_alpha = 1)
  reproducibility_index(diag_glmnet)
}
```

---

selection_stability *Selection stability*

Description

Measures how consistently each predictor is *selected* across perturbation iterations. The definition depends on the modeling backend:

Usage

```
selection_stability(diag_obj)
```

Arguments

diag_obj	A reprostat object from run_diagnostics.

Details

`"lm"`, `"glm"`, `"rlm"` **Sign consistency**: the proportion of perturbation iterations in which the estimated coefficient has the same sign as in the base fit. A value of 1 means the direction of the effect is perfectly stable; 0.5 means the sign is random. Returns `NA` for a predictor whose base-fit coefficient is exactly zero.

`"glmnet"` **Non-zero selection frequency**: the proportion of perturbation iterations in which the coefficient is non-zero (i.e. the variable survives the regularisation penalty).

The intercept is always excluded.

Value

A named numeric vector of selection stability values in $[0, 1]$, excluding the intercept.

Examples

```
set.seed(1)
d <- run_diagnostics(mpg ~ wt + hp, data = mtcars, B = 50)
selection_stability(d)
```

Index