Package: uddbart (via r-universe)

June 30, 2026

Type Package

Title Unified Dynamic Deep 'BART' for Interval-Censored Survival

Version 0.2.0

Author Xulin Pan [aut, cre] (ORCID:
<https://orcid.org/0000-0002-1072-2539>)

Maintainer Xulin Pan <xulinpanias@gmail.com>

Description Implements U-DDBART-IC, a unified Bayesian workflow for dynamic risk prediction from irregular longitudinal biomarkers when event times are interval-censored between clinical visits. The package turns long-format biomarker histories and patient-level interval endpoints L, R, C and delta into a discrete-time follow-up grid, summarises each landmark history with nine interpretable trajectory features (current, baseline and previous biomarker values, last visit gap, local slope, cumulative decline, best value, elapsed time and visit count), fits discrete-time interval hazards using optional logit-link Bayesian additive regression trees, a generalized linear model fallback, or a lightweight variational approximation, accumulates survival from the discrete-time product, and evaluates the interval-censored likelihood. Fitted models return landmark risk predictions over user-specified horizons with posterior or bootstrap uncertainty by evaluating survival ratios across fitted hazard draws. Utilities are provided for simulation, staged model fitting, plotting and summarising dynamic risk curves, IPCW Brier scores, cumulative/dynamic time-dependent area under the curve, calibration tables, and an anonymised chronic myeloid leukaemia molecular-monitoring example data set.

License MIT + file LICENSE

Encoding UTF-8

Language en-US

LazyData true

Depends R (>= 4.1.0)

Imports stats, graphics, grDevices

Suggests testthat (>= 3.0.0), knitr, rmarkdown, BART (>= 2.9)

VignetteBuilder knitr

RoxygenNote 7.3.3

Config/testthat/edition 3

URL https://github.com/xulinpan/uddbart

BugReports https://github.com/xulinpan/uddbart/issues

NeedsCompilation no

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

Date/Publication 2026-06-30 21:11:33 UTC

RemoteUrl https://github.com/cran/uddbart

RemoteRef HEAD

RemoteSha 736d3500b662321c0f0d3abe2f195c51261c2e85

Contents

auc_td . . . 3
brier_score . . . 4
calibration_table . . . 5
cml_data . . . 6
compute_survival_from_hazard . . . 7
construct_latent_state . . . 8
fit_bart_hazard . . . 9
fit_uddbart_vi . . . 10
interval_survival_likelihood . . . 11
make_followup_grid . . . 12
plot.uddbart . . . 13
predict.uddbart . . . 13
prepare_uddbart_data . . . 15
simulate_uddbart_data . . . 16
summary.uddbart . . . 18
uddbart . . . 18

Index **21**

auc_td *Time-dependent (cumulative/dynamic) AUC*

Description

IPCW estimator of the cumulative/dynamic time-dependent AUC for landmark dynamic predictions: among subjects at risk at the landmark, cases are those with an event within the horizon and controls are those event-free beyond it.

Usage

```
auc_td(
  pred,
  event,
  id = "patient_id",
  landmark_col = "landmark",
  ipcw = TRUE,
  eps_G = 0.001
)
```

Arguments

pred	A prediction data frame from predict.uddbart() containing the id column, the landmark column, horizon and risk. All rows must share a single landmark time.
event	Observed outcomes with the id column and R, C, delta (as in the event element of prepare_uddbart_data()); the evaluation time is R for events and C for censored subjects.
id	Name of the patient identifier column.
landmark_col	Name of the landmark column in pred.
ipcw	Logical; apply IPCW (TRUE) or the unweighted Brier score.
eps_G	Lower clamp on the censoring-survival weights for numerical stability.

Value

A data frame with landmark, horizon, auc, n_cases and n_controls.

See Also

brier_score(), calibration_table()

Examples

```
pred <- data.frame(patient_id = c("a", "b", "c"), landmark = 0,
                   horizon = 12, risk = c(0.9, 0.2, 0.1))
event <- data.frame(patient_id = c("a", "b", "c"),
                    R = c(6, 20, 30), C = c(6, 20, 30), delta = c(1, 1, 0))
auc_td(pred, event)
```

`brier_score` *Inverse-probability-of-censoring-weighted Brier score*

Description

Computes the time-dependent Brier score (and integrated Brier score) for landmark dynamic predictions, using inverse-probability-of-censoring weighting (IPCW) to account for right censoring (Graf et al., 1999).

Usage

```
brier_score(
  pred,
  event,
  id = "patient_id",
  landmark_col = "landmark",
  ipcw = TRUE,
  eps_G = 0.001
)
```

Arguments

`pred`	A prediction data frame from `predict.uddbart()` containing the id column, the landmark column, `horizon` and `risk`. All rows must share a single landmark time.
`event`	Observed outcomes with the id column and `R`, `C`, `delta` (as in the `event` element of `prepare_uddbart_data()`); the evaluation time is R for events and C for censored subjects.
`id`	Name of the patient identifier column.
`landmark_col`	Name of the landmark column in `pred`.
`ipcw`	Logical; apply IPCW (`TRUE`) or the unweighted Brier score.
`eps_G`	Lower clamp on the censoring-survival weights for numerical stability.

Value

A data frame with one row per horizon: `landmark`, `horizon`, `brier`, `n` (subjects at risk) and `n_events` (events within the horizon). The attribute `"IBS"` holds the integrated Brier score (trapezoidal average over the horizons).

References

Graf, E., Schmoor, C., Sauerbrei, W. and Schumacher, M. (1999). Assessment and comparison of prognostic classification schemes for survival data. *Statistics in Medicine* 18, 2529-2545.

See Also

auc_td(), calibration_table(), predict.uddbart()

Examples

```
pred <- data.frame(patient_id = c("a", "b", "c"), landmark = 0,
                   horizon = 12, risk = c(0.9, 0.2, 0.1))
event <- data.frame(patient_id = c("a", "b", "c"),
                    R = c(6, 20, 30), C = c(6, 20, 30), delta = c(1, 1, 0))
brier_score(pred, event)
```

---

calibration_table *Calibration table for a landmark prediction*

Description

Groups subjects into bins of predicted risk and compares the mean predicted risk with the IPCW-estimated observed event probability within the horizon, for a single landmark and horizon.

Usage

```
calibration_table(
  pred,
  event,
  horizon,
  id = "patient_id",
  landmark_col = "landmark",
  groups = 5,
  eps_G = 0.001
)
```

Arguments

pred	A prediction data frame from predict.uddbart() containing the id column, the landmark column, horizon and risk. All rows must share a single landmark time.
event	Observed outcomes with the id column and R, C, delta (as in the event element of prepare_uddbart_data()); the evaluation time is R for events and C for censored subjects.
horizon	A single horizon at which to calibrate.
id	Name of the patient identifier column.

landmark_col	Name of the landmark column in pred.
groups	Number of equal-count predicted-risk bins.
eps_G	Lower clamp on the censoring-survival weights for numerical stability.

Value

A data frame with one row per bin: bin, n, pred_mean (mean predicted risk) and obs_rate (IPCW observed event probability).

See Also

brier_score(), auc_td()

Examples

```
pred <- data.frame(patient_id = letters[1:6], landmark = 0, horizon = 12,
                   risk = c(0.9, 0.8, 0.5, 0.4, 0.2, 0.1))
event <- data.frame(patient_id = letters[1:6],
                    R = c(6, 8, 20, 9, 30, 40), C = c(6, 8, 20, 9, 30, 40),
                    delta = c(1, 1, 0, 1, 0, 0))
calibration_table(pred, event, horizon = 12, groups = 2)
```

---

cml_data *CML molecular-monitoring data (real cohort) for U-DDBART-IC*

Description

Anonymised BCR-ABL monitoring data for 84 imatinib-treated chronic myeloid leukaemia (CML) patients, formatted for uddbart(). The event of interest is time to deep molecular response (MR4.5). Provided as two linked tables.

Usage

```
cml_long

cml_event
```

Format

cml_long Long-format biomarker monitoring data (469 rows): one row per patient-visit, with patient_id, t_months (months since imatinib start) and log_mrd (observed log molecular residual disease).

cml_event Patient-level interval-censoring data (84 rows): one row per patient, with patient_id, L and R (the visit times bracketing the first observed MR4.5; the event time lies in $(L, R]$), C (the right-censoring time) and delta (1 = MR4.5 observed, 0 = censored). 67 patients reach MR4.5; 17 are censored.

Source

Anonymised CML molecular-monitoring cohort (2026).

---

compute_survival_from_hazard

Discrete-time survival from interval hazards

Description

Accumulates the discrete-time survival function $S_i(a_k) = \prod_{\ell \le k}(1-\lambda_{i\ell})$ from per-interval hazards, per patient, respecting interval order.

Usage

```
compute_survival_from_hazard(hazard, risk, id = "patient_id", level = 0.95)
```

Arguments

`hazard`	Either a numeric vector of length `nrow(risk)` (posterior-mean hazards) or a numeric matrix of posterior draws with `nrow(risk)` columns (one column per at-risk interval, rows = MCMC draws).
`risk`	A risk set (from `prepare_uddbart_data()`) with the id column, `interval` and `t_end`, aligned row-for-row to `hazard`.
`id`	Name of the patient identifier column.
`level`	Width of the equal-tailed credible interval for the tidy summary (used only when `hazard` is a matrix of draws).

Value

A list with:

`surv_draws` matrix (draws x intervals) of $S_i(a_k)$ aligned to the rows of `risk` (a single-row matrix if `hazard` is a vector).

`summary` a tidy data frame with the id, `interval`, `t_end`, `hazard`, `surv` (posterior mean), and, for draws, `surv_lower` / `surv_upper`.

See Also

`fit_bart_hazard()`, `interval_survival_likelihood()`

Examples

```
sim <- simulate_uddbart_data(n = 20, seed = 1)
prep <- prepare_uddbart_data(sim$long, sim$event)
## toy hazards for illustration
h <- rep(0.1, nrow(prep$risk))
S <- compute_survival_from_hazard(h, prep$risk)
head(S$summary)
```

construct_latent_state

Construct the latent molecular-response state

Description

Builds the engineered latent-state feature matrix Z_{ik} used by the BART hazard. For each row of a prepared interval risk set, the biomarker history available at the start of the interval (`t_start`, i.e. a_{k-1}) is summarised into the nine features of the U-DDBART-IC version 0.1 latent state: current biomarker $X^{(cur)}$, baseline $X^{(base)}$, previous $X^{(prev)}$, last visit gap Δt, `slope`, cumulative `decline`, `best` response, treatment duration `treattime`, and number of visits `nvisit`.

Usage

```
construct_latent_state(
  long,
  risk,
  id = "patient_id",
  time = "t_months",
  marker = "log_mrd"
)
```

Arguments

`long`	Long-format biomarker data, one row per patient-visit.
`risk`	A prepared interval risk set (from `prepare_uddbart_data()`); must contain the id column and `t_start`.
`id`	Name of the patient identifier column (in both `long` and `risk`).
`time`	Name of the visit-time column in `long`.
`marker`	Name of the biomarker column in `long` (e.g. log10 BCR-ABL).

Details

The history for interval $(a_{k-1}, a_k]$ is all visits with $t_{ij} \leq a_{k-1}$, so the latent state uses only information available at the start of the interval. This makes the same feature construction valid for dynamic prediction, where future biomarkers are unknown.

Value

A numeric matrix with one row per row of `risk` and nine named columns: `x_cur`, `x_base`, `x_prev`, `dt`, `slope`, `decline`, `best`, `treattime`, `nvisit`.

See Also

`prepare_uddbart_data()`, `fit_bart_hazard()`

Examples

```
sim <- simulate_uddbart_data(n = 30, seed = 1)
prep <- prepare_uddbart_data(sim$long, sim$event)
Z <- construct_latent_state(sim$long, prep$risk)
head(Z)
```

`fit_bart_hazard` *Fit the logit-link interval hazard*

Description

Fits the discrete-time interval hazard $\mathrm{logit}(\lambda_{ik}) = f(Z_{ik}, a_k)$. If the optional `BART` package is installed, a logit-link BART model is used; otherwise the function falls back to a binomial `glm`.

Usage

```
fit_bart_hazard(
  Z,
  grid_time,
  y,
  ntree = 200L,
  ndpost = 1000L,
  nskip = 250L,
  keepevery = 1L,
  sparse = FALSE,
  seed = NULL,
  verbose = FALSE
)
```

Arguments

`Z`	Latent-state design matrix from `construct_latent_state`, one row per at-risk interval.
`grid_time`	Numeric vector of interval end times a_k (the `t_end` column), aligned to the rows of `Z`.
`y`	Integer 0/1 event indicator, aligned to the rows of `Z`.
`ntree`, `ndpost`, `nskip`, `keepevery`	BART hyper-parameters used when the optional `BART` package is installed.
`sparse`	Logical; use the sparse Dirichlet variable-selection prior when using `BART`.
`seed`	Optional integer seed.
`verbose`	Logical; print BART progress when using `BART`.

Value

A list of class `"uddbart_hazard"` with the fitted model, backend method, training design, and design-matrix column names `xnames`.

See Also

uddbart, compute_survival_from_hazard

Examples

```
sim <- simulate_uddbart_data(n = 40, seed = 1)
prep <- prepare_uddbart_data(sim$long, sim$event)
Z <- construct_latent_state(sim$long, prep$risk)
h <- fit_bart_hazard(Z, prep$risk$t_end, prep$risk$y,
                     ntree = 20, ndpost = 50, nskip = 25, seed = 1)
```

---

fit_uddbart_vi *Fit the UDDBART-VI interval hazard*

Description

Fits a lightweight Gaussian variational approximation to the logistic discrete-time hazard coefficients using the same latent-state design as fit_bart_hazard. This backend requires only base R and stats.

Usage

```
fit_uddbart_vi(
  Z,
  grid_time,
  y,
  prior_sd = 2.5,
  max_iter = 100L,
  tol = 1e-06,
  ndraw = 1000L,
  seed = NULL,
  verbose = FALSE
)
```

Arguments

Z	Latent-state design matrix from construct_latent_state, one row per at-risk interval.
grid_time	Numeric vector of interval end times, aligned to rows of Z.
y	Integer 0/1 event indicator, aligned to rows of Z.
prior_sd	Prior standard deviation for non-intercept coefficients.
max_iter	Maximum Newton/variational optimization iterations.
tol	Convergence tolerance.
ndraw	Number of coefficient draws from the variational approximation.
seed	Optional integer seed.
verbose	Logical; currently reserved for future progress output.

Value

A list of class `c("uddbart_vi", "uddbart_hazard")` containing the variational coefficient mean, covariance, coefficient draws, hazard draws, and training design.

Examples

```
sim <- simulate_uddbart_data(n = 20, seed = 1)
prep <- prepare_uddbart_data(sim$long, sim$event)
Z <- construct_latent_state(sim$long, prep$risk)
h <- fit_uddbart_vi(Z, prep$risk$t_end, prep$risk$y, ndraw = 20, seed = 1)
h$method
```

`interval_survival_likelihood` *Interval-censored survival (log-)likelihood*

Description

Evaluates the U-DDBART-IC interval-censored likelihood contributions

$$\mathcal{L}_i = [S_i(L_i) - S_i(R_i)]^{\delta_i}\,[S_i(C_i)]^{1-\delta_i}$$

given a per-patient discrete-time survival function. Survival is treated as a right-continuous grouped step function that changes only at the grid points, evaluated by last-value-carried-forward.

Usage

```
interval_survival_likelihood(
  surv_summary,
  event,
  id = "patient_id",
  eps = 1e-12
)
```

Arguments

`surv_summary`	The `summary` data frame from `compute_survival_from_hazard()` (columns: id, `t_end`, `surv`), giving each patient's survival at their grid endpoints.
`event`	Per-patient endpoints with the id column and `L`, `R`, `C`, `delta` (as produced in the `event` element of `prepare_uddbart_data()`).
`id`	Name of the patient identifier column.
`eps`	Lower clamp applied to each likelihood contribution before taking logs, to avoid $\log 0$.

Value

A list with `per_patient` (a data frame with the id, `lik` and `loglik`) and `loglik` (the total observed-data log-likelihood). The plug-in survival in `surv_summary` (posterior mean) is used.

See Also

compute_survival_from_hazard(), uddbart()

Examples

```
sim <- simulate_uddbart_data(n = 20, seed = 1)
prep <- prepare_uddbart_data(sim$long, sim$event)
S <- compute_survival_from_hazard(rep(0.1, nrow(prep$risk)), prep$risk)
ll <- interval_survival_likelihood(S$summary, prep$event)
ll$loglik
```

---

make_followup_grid *Construct a discrete follow-up grid*

Description

Builds the discrete-time grid $0 = a_0 < a_1 < \cdots < a_K$ on which the interval hazard is defined.

Usage

```
make_followup_grid(by = 3, max_time = 60, breaks = NULL)
```

Arguments

by	Grid spacing (e.g. 3 for quarterly intervals, giving $a_k = 3k$).
max_time	The largest grid time a_K (the end of follow-up).
breaks	Optional explicit, increasing vector of grid times beginning at 0. If supplied, by and max_time are ignored.

Value

A numeric vector of grid breakpoints starting at 0.

See Also

prepare_uddbart_data()

Examples

```
make_followup_grid(by = 3, max_time = 24)
```

plot.uddbart — *Plot dynamic risk curves from a U-DDBART-IC fit*

Description

Plots the dynamic risk $\pi_i(t, \tau)$ (or survival ratio) as a function of the horizon, one curve per (patient, landmark) prediction, with a shaded posterior credible band.

Usage

```
## S3 method for class 'uddbart'
plot(
  x,
  pred,
  what = c("risk", "survival"),
  landmark = "landmark",
  add = FALSE,
  ...
)
```

Arguments

`x`	A `uddbart` object (the fitted model).
`pred`	A data frame from `predict.uddbart()` evaluated at several horizons. Required.
`what`	`"risk"` (the default) or `"survival"`.
`landmark`	Name of the landmark column used in `pred`.
`add`	Logical; add to an existing plot.
`...`	Further graphical parameters passed to `graphics::lines()`.

Value

`pred`, invisibly.

predict.uddbart — *Dynamic landmark risk prediction from a U-DDBART-IC fit*

Description

Computes the dynamic prediction target

$$\pi_i(t, \tau) = P(T_i \leq t + \tau \mid T_i > t, \mathcal{H}_i(t)) = 1 - \frac{S_i(t + \tau)}{S_i(t)},$$

conditional on each patient's biomarker history up to a landmark time t.

Usage

```
## S3 method for class 'uddbart'
predict(
  object,
  long,
  newdata,
  horizon,
  landmark = "landmark",
  level = 0.95,
  ...
)
```

Arguments

`object`	A fitted `uddbart()` model.
`long`	Long-format biomarker data for the patients to predict (same columns as used in fitting). Histories are taken up to the landmark.
`newdata`	A data frame with one row per prediction, containing the patient id column and a landmark-time column.
`horizon`	Numeric vector of prediction horizons τ. The risk of an event by $t+\tau$ is returned for each.
`landmark`	Name of the landmark-time column in `newdata`.
`level`	Width of the equal-tailed posterior credible interval.
`...`	Unused; for S3 compatibility.

Details

The latent state for grid interval $(a_{k-1}, a_k]$ is built from the history up to $\min(a_{k-1}, t)$, freezing the biomarker trajectory at the landmark for future intervals (no biomarkers are assumed after t). Survival is accumulated from the posterior hazard draws and the ratio $S(t+\tau)/S(t)$ is formed draw by draw, so the reported credible interval carries full posterior uncertainty.

Value

A data frame with one row per (prediction, horizon): the patient id, `landmark`, `horizon`, `risk` (posterior-mean $\pi_i(t, \tau)$), `risk_lower`, `risk_upper`, and `survival` ($1 - \text{risk}$).

See Also

`uddbart()`

Examples

```
sim <- simulate_uddbart_data(n = 40, seed = 1)
fit <- uddbart(sim$long, sim$event, grid_by = 3,
               ntree = 20, ndpost = 50, nskip = 25, seed = 1)
nd <- data.frame(patient_id = sim$event$patient_id[1:3], landmark = 12)
predict(fit, sim$long, nd, horizon = c(6, 12, 24))
```

prepare_uddbart_data *Prepare interval-censored data for U-DDBART-IC*

Description

Converts longitudinal biomarker data and patient-level interval-censored event information into the discrete-time interval ("risk set") representation used by `uddbart()`. Each patient contributes one row per grid interval in which they are at risk, with a binary indicator `y` equal to 1 for the interval in which the event is observed.

Usage

```
prepare_uddbart_data(
  long,
  event,
  id = "patient_id",
  L = "L",
  R = "R",
  C = "C",
  delta = "delta",
  grid = NULL,
  grid_by = 3
)
```

Arguments

`long`	Long-format biomarker data, one row per patient-visit (used only to determine each patient's id; the biomarker values are summarised later by `construct_latent_state()`).
`event`	A patient-level data frame with the id column and the interval-censoring endpoints. For an observed event the event time lies in $(L_i, R_i]$ with `delta = 1`; for right censoring the patient is known event-free up to C_i with `delta = 0`.
`id`	Name of the patient identifier column.
`L, R`	Names of the left and right interval-endpoint columns in `event` (used when `delta == 1`).
`C`	Name of the right-censoring time column in `event` (used when `delta == 0`). If `NULL`, `R` is used as the censoring time for `delta == 0` rows.
`delta`	Name of the 0/1 event-indicator column in `event`.
`grid`	A grid from `make_followup_grid()`. If `NULL`, a grid is built from `grid_by` up to the maximum observed endpoint.
`grid_by`	Grid spacing used when `grid` is `NULL`.

Details

A patient is at risk in interval $(a_{k-1}, a_k]$ while $a_{k-1} < \text{obs_time}_i$. For an event patient (`delta = 1`), `y = 1` in the interval containing R_i (the first grid interval with $a_k \geq R_i$) and `y = 0` earlier. Right-censored patients have `y = 0` throughout. This is the standard discrete-time (grouped) survival encoding whose Bernoulli factorisation matches the interval-censored likelihood evaluated on the grid.

Value

A list with elements:

`risk` the interval risk set: one row per at-risk grid interval with columns `patient_id` (the id), `interval` (k), `t_start` (a_{k-1}), `t_end` (a_k) and `y` (event indicator).

`grid` the grid used.

`event` the per-patient endpoints used, with a resolved `obs_time` (R for events, C for censored).

See Also

`make_followup_grid()`, `construct_latent_state()`, `uddbart()`

Examples

```
sim <- simulate_uddbart_data(n = 20, seed = 1)
prep <- prepare_uddbart_data(sim$long, sim$event)
head(prep$risk)
```

`simulate_uddbart_data` *Simulate irregular longitudinal biomarkers with interval-censored events*

Description

Generates synthetic U-DDBART-IC data: each patient has a latent linear log-biomarker trajectory $X_i^*(t) = b_{0i} + b_{1i}t$, observed with measurement error at irregular visit times. The event (reaching deep molecular response, MR4.5) occurs when the latent trajectory first crosses a threshold; because the biomarker is only measured at visits, the event time is interval-censored between the last visit below threshold (L_i) and the first visit at or beyond it (R_i). Patients whose trajectory never crosses within follow-up are right-censored at C_i.

Usage

```
simulate_uddbart_data(
  n = 200,
  grid_by = 3,
  max_followup = 60,
  min_followup = 24,
  visit_jitter = 0.25,
```

```
  threshold = -4.5,
  b0_mean = 0.7,
  b0_sd = 0.6,
  slope_mean = -0.1,
  slope_sd = 0.06,
  sigma = 0.3,
  seed = NULL
)
```

Arguments

`n`	Number of patients.
`grid_by`	Nominal spacing between scheduled visits (months).
`max_followup`	Maximum administrative follow-up time (months); each patient's censoring time is drawn between `min_followup` and this value.
`min_followup`	Minimum administrative follow-up time (months).
`visit_jitter`	SD of multiplicative timing noise on visit times.
`threshold`	Latent log-biomarker threshold defining the event (e.g. `-4.5` for MR4.5 on a $\log_{10}$ ratio scale).
`b0_mean`, `b0_sd`	Mean and SD of the patient baseline b_{0i}.
`slope_mean`, `slope_sd`	Mean and SD of the patient slope b_{1i} (negative slopes correspond to responders).
`sigma`	Measurement-error SD of the observed biomarker.
`seed`	Optional integer seed.

Value

A list with:

`long` long-format biomarker data: `patient_id`, `t_months`, `log_mrd`, and the noise-free `latent` value.

`event` per-patient interval-censoring data: `patient_id`, `L`, `R`, `C`, `delta`, and the true crossing time `true_T`.

`params` per-patient true `b0`, `b1`, and the generative settings.

See Also

`prepare_uddbart_data()`, `uddbart()`

Examples

```
sim <- simulate_uddbart_data(n = 50, seed = 1)
str(sim$event)
```

summary.uddbart *Summarise a U-DDBART-IC fit*

Description

Summarise a U-DDBART-IC fit

Usage

```
## S3 method for class 'uddbart'
summary(object, ...)

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

Arguments

object	A fitted uddbart() model.
...	Unused.
x	A summary.uddbart object.

Value

An object of class "summary.uddbart" with fit metadata and quantiles of the in-sample interval hazard.

uddbart *Fit the unified U-DDBART-IC model*

Description

End-to-end fit of the U-DDBART-IC pipeline: prepare the interval-censored risk set on a discrete grid, build the engineered latent state, fit the logit-link interval hazard, accumulate the discrete-time survival and evaluate the interval-censored likelihood. The interval hazard can be fitted with optional BART, a glm fallback, or the lightweight UDDBART-VI backend.

Usage

```
uddbart(
  long,
  event,
  id = "patient_id",
  time = "t_months",
  marker = "log_mrd",
  L = "L",
```

```
  R = "R",
  C = "C",
  delta = "delta",
  grid = NULL,
  grid_by = 3,
  engine = c("bart", "vi"),
  ntree = 200L,
  ndpost = 1000L,
  nskip = 250L,
  keepevery = 1L,
  sparse = FALSE,
  vi_prior_sd = 2.5,
  vi_max_iter = 100L,
  vi_tol = 1e-06,
  seed = NULL,
  verbose = FALSE
)

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

Arguments

`long`	Long-format biomarker data, one row per patient-visit.
`event`	Patient-level interval-censoring data with the id column and `L`, `R`, `C`, `delta`.
`id`, `time`, `marker`	Column names for the patient id, visit time and biomarker in `long`.
`L`, `R`, `C`, `delta`	Column names of the interval-censoring endpoints and the event indicator in `event`.
`grid`	Optional grid from `make_followup_grid`; built from `grid_by` if `NULL`.
`grid_by`	Grid spacing (months) used when `grid` is `NULL`.
`engine`	Hazard backend. `"bart"` uses optional BART with a glm fallback; `"vi"` uses the lightweight UDDBART-VI approximation.
`ntree`, `ndpost`, `nskip`, `keepevery`, `sparse`	BART hyper-parameters passed to `fit_bart_hazard` when `engine = "bart"`.
`vi_prior_sd`, `vi_max_iter`, `vi_tol`	UDDBART-VI controls used when `engine = "vi"`.
`seed`	Optional integer seed.
`verbose`	Logical; print progress.
`x`	A `uddbart` object.
`...`	Further arguments (ignored).

Value

An object of class `"uddbart"` containing the fitted `hazard` model, the `risk` set, the `grid`, in-sample survival `surv`, the interval-censored log-likelihood `loglik`, the per-patient `event` endpoints, column-name metadata, and the `call`.

See Also

predict.uddbart, simulate_uddbart_data

Examples

```
sim <- simulate_uddbart_data(n = 40, seed = 1)
fit <- uddbart(sim$long, sim$event, grid_by = 3,
               engine = "vi", ndpost = 50, seed = 1)
fit
```

Index

∗ **datasets**
    cml_data, 6

auc_td, 3
auc_td(), *5, 6*

brier_score, 4
brier_score(), *3*, *6*

calibration_table, 5
calibration_table(), *3*, *5*
cml_data, 6
cml_event *(*cml_data*)*, 6
cml_long *(*cml_data*)*, 6
compute_survival_from_hazard, 7, *10*
compute_survival_from_hazard(), *11, 12*
construct_latent_state, 8, *9, 10*
construct_latent_state(), *15, 16*

fit_bart_hazard, 9, *10*, *19*
fit_bart_hazard(), *7, 8*
fit_uddbart_vi, 10

graphics::lines(), *13*

interval_survival_likelihood, 11
interval_survival_likelihood(), *7*

make_followup_grid, 12, *19*
make_followup_grid(), *15, 16*

plot.uddbart, 13
predict.uddbart, 13, *20*
predict.uddbart(), *3–5*, *13*
prepare_uddbart_data, 15
prepare_uddbart_data(), *3–5*, *7, 8*, *11, 12*, *17*
print.summary.uddbart *(*summary.uddbart*)*, 18
print.uddbart *(*uddbart*)*, 18

simulate_uddbart_data, 16, *20*

summary.uddbart, 18

uddbart, *10*, 18
uddbart(), *6*, *12*, *14–18*