Package: picMort (via r-universe)

May 30, 2026

Title Reproducible Pediatric ICU Mortality Benchmark on PIC V1.1.0

Version 0.1.1

Description Tools and pipelines for a calibration-first pediatric ICU mortality benchmark on the open-access PIC v1.1.0 database. Authoritative cohort specification, T+24h prediction-window feature extraction, PIM3 reconstruction, calibration and decision-curve evaluation. The package is the methodological substrate for the companion manuscript, illustrated end to end in a worked vignette.

License MIT + file LICENSE

URL https://github.com/max578/picMort, https://max578.github.io/picMort

BugReports https://github.com/max578/picMort/issues

Encoding UTF-8

Language en-US

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Depends R (>= 4.2)

Imports data.table (>= 1.15.0), glmnet (>= 4.1), xgboost (>= 1.7), recipes (>= 1.0), rsample (>= 1.2), ggplot2 (>= 3.5), lifecycle, stats, utils, fs, rprojroot

Suggests yardstick (>= 1.3), probably (>= 1.0), dcurves (>= 0.4), brms (>= 2.20), rstan (>= 2.30), BH, posterior, bayesplot, targets (>= 1.5), tarchetypes, testthat (>= 3.0.0), knitr, rmarkdown, arrow, fst, viridisLite, withr

VignetteBuilder knitr

Config/testthat/edition 3

LazyData false

Config/pak/sysreqs cmake make libicu-dev libuv1-dev

Repository https://max578.r-universe.dev

Date/Publication 2026-05-25 00:14:04 UTC

RemoteUrl https://github.com/max578/picMort

RemoteRef main

RemoteSha f95941a1507043883cdaf3105d0030f60e669bb9

Contents

`assert_cohort_invariants` *Cohort invariants (used by tests and as a sanity gate at runtime)*

Description

[Experimental]

Verifies that a cohort `data.table` satisfies the structural and epidemiological invariants committed in `vignettes/cohort_spec.Rmd`. Raises a structured error listing every violation; passes silently (invisibly returning `TRUE`) when all invariants hold.

Usage

```
assert_cohort_invariants(cohort, expected = default_cohort_expectations())
```

Arguments

cohort	A cohort data.table from build_cohort().
expected	Named list of expected ranges. Defaults to the values committed in vignettes/cohort_spec.Rmd.

Value

Invisibly returns TRUE if all invariants hold; raises an error listing every violation otherwise.

Examples

```
toy <- readRDS(system.file("extdata", "toy_cohort.rds",
                           package = "picMort"))
toy_expectations <- list(
  n_min            = 50L,
  n_max            = 150L,
  mortality_rate   = c(0.05, 0.20),
  age_range_years  = c(0, 18),
  sex_levels       = c("F", "M"),
  distinct_subject = TRUE,
  no_overlap_stays = TRUE
)
assert_cohort_invariants(toy, expected = toy_expectations)
```

---

audit_no_leakage *Audit feature matrix for prediction-window leakage*

Description

[Experimental]

Checks (i) no variable name contains a forbidden temporal token (LOS, discharge, deathtime, etc.); (ii) every dictionary entry declares the same window_hours; (iii) optional provenance check on raw events confirms every observation occurred in [0, window).

Usage

```
audit_no_leakage(feature_dict, raw_events = NULL, window_hours = 24L)
```

Arguments

feature_dict	Feature dictionary from build_features().
raw_events	Optional list of long-format event tables. Each table must contain either numeric t_hours offsets from ICU admission, or POSIXct columns t and intime from which the offset can be derived. When present, the function verifies that every event is strictly within [0, window_hours).
window_hours	Prediction-window length used at extraction.

Value

Invisibly `TRUE` on pass; raises a structured error on fail.

Examples

```
# A minimal in-spec dictionary
ok_dict <- data.table::data.table(
  variable       = c("age_years", "hr_min", "spo2_min"),
  source         = c("cohort", "CHARTEVENTS", "CHARTEVENTS"),
  transformation = c("static", "min over [0,24)h", "min over [0,24)h"),
  clinical_group = c("demographics", "vitals", "vitals"),
  window_hours   = 24L
)
audit_no_leakage(ok_dict, window_hours = 24L)

# A leaky dictionary (LOS is forbidden post-window information)
bad_dict <- data.table::copy(ok_dict)
bad_dict <- rbind(bad_dict,
  data.table::data.table(variable = "los_hours", source = "cohort",
                         transformation = "static",
                         clinical_group = "demographics",
                         window_hours = 24L))
tryCatch(audit_no_leakage(bad_dict, window_hours = 24L),
         error = function(e) conditionMessage(e))
```

`build_cohort` *Build the canonical study cohort*

Description

[Experimental]

Constructs the canonical study cohort per `vignettes/cohort_spec.Rmd`: first ICU stay per patient; age 0-18 y at admission; valid ICU admit/discharge timestamps; in-hospital mortality outcome attached.

Cohort assembly is the single source of truth for the calibration-first analysis. Deviations are forbidden; sensitivity analyses operate on the same base cohort with documented filters.

Usage

```
build_cohort(
  paths,
  min_los_hours = 24L,
  prediction_window_hours = min_los_hours,
  verbose = TRUE
)
```

Arguments

`paths`	Output of `pic_paths()`.
`min_los_hours`	Minimum ICU length-of-stay in hours required for the prediction-window framing. Default 24 (T+24h lock). Set to 12 for the T+12h sensitivity arm.
`prediction_window_hours`	Prediction window in hours. Used by `is_surgical` to filter `SURGERY_VITAL_SIGNS` evidence to rows that occur strictly before `T0 + prediction_window_hours`. Without this filter, surgery evidence appearing only after the prediction-window close leaks future information. Default = `min_los_hours` (T+24h main analysis); pass `12L` for the T+12h sensitivity arm.
`verbose`	Logical. Emit cohort-attrition log lines.

Value

A `data.table` keyed by `subject_id`, `hadm_id`, `icustay_id`. The cohort table carries an `"attrition"` attribute documenting the exclusion cascade, accessed via `cohort_attrition()`.

Examples

```
# `build_cohort()` reads the registered PIC v1.1.0 CSVs. A pre-built
# synthetic 80-row toy cohort with the same schema ships in
# `inst/extdata/toy_cohort.rds` for examples and tests.
toy <- readRDS(system.file("extdata", "toy_cohort.rds",
                           package = "picMort"))
nrow(toy)
names(toy)

## Not run:
paths  <- pic_paths()
cohort <- build_cohort(paths, min_los_hours = 24L, verbose = FALSE)
assert_cohort_invariants(cohort)

## End(Not run)
```

`build_features` *Build T+window prediction-window feature matrix*

Description

[Experimental]

Extracts demographics (from cohort), vitals (`CHARTEVENTS`), and labs (`LABEVENTS`) within `window_hours` of ICU admission. **No feature uses any timestamp at or after `intime + window_hours`.** `audit_no_leakage()` runs as a runtime invariant.

Usage

```
build_features(
  cohort,
  paths,
  window_hours = 24L,
  feature_set = c("simple", "rich")
)
```

Arguments

`cohort`	A cohort `data.table` from `build_cohort()`.
`paths`	Output of `pic_paths()`.
`window_hours`	Prediction-window length in hours. Default 24.
`feature_set`	One of `"simple"` (default; min/max/mean/last + missingness indicators) or `"rich"` (adds slopes / count rates; reserved for sensitivity analysis).

Value

A list with elements:

- `x` - feature `data.table` keyed on `icustay_id`
- `y` - outcome integer vector aligned to `x$icustay_id`
- `dict` - feature dictionary (`data.table`)
- `audit` - result of `audit_no_leakage()` (TRUE on pass)
- `window_hours`, `feature_set`

Examples

```
## Not run:
paths    <- pic_paths()
cohort   <- build_cohort(paths, min_los_hours = 24L, verbose = FALSE)
features <- build_features(cohort, paths, window_hours = 24L)
dim(features$x)
table(features$y)

## End(Not run)
```

`calibration_suite` *Calibration suite (the manuscript headline)*

Description

[Experimental]

Computes calibration slope, intercept, integrated calibration index (ICI), calibration-in-the-large, and a smoothed calibration curve (`loess`). Bootstrap percentile CIs (1,000 reps default) on every point estimate.

Definitions:

- **Calibration slope**: logistic regression of `y ~ logit(prob)`, slope coefficient. Ideal = 1; <1 indicates over-fitting.
- **Calibration intercept**: logistic regression of `y ~ offset(logit(prob))`, intercept. Ideal = 0; >0 indicates systematic under-prediction.
- **Calibration-in-the-large**: `log(prev_obs / prev_pred)`. Ideal = 0.
- **ICI**: mean absolute difference between smoothed observed and predicted probabilities.

Usage

```
calibration_suite(prob, y, n_boot = 1000L, seed = 20260508L)
```

Arguments

`prob`	Numeric vector of predicted probabilities.
`y`	Outcome (0/1 integer).
`n_boot`	Bootstrap replicates; default 1000.
`seed`	Integer seed; default 20260508.

Value

A list with elements `slope`, `intercept`, `ici`, `cit_large` (each named `c(estimate, lower, upper)`), `curve` (a `data.frame` for plotting), and `n`, `n_events`.

Examples

```
set.seed(20260517L)
n <- 200L
prob <- stats::plogis(stats::rnorm(n, mean = -2, sd = 1))
y    <- stats::rbinom(n, 1L, prob)
cal  <- calibration_suite(prob, y, n_boot = 50L)
cal$slope
cal$ici
head(cal$curve)
```

cohort_attrition *Cohort attrition table (for the manuscript Methods figure)*

Description

[Experimental]

Returns the exclusion cascade documenting how many ICU stays were dropped at each cohort-filter step. Surfaces the same data carried as the `"attrition"` attribute on `build_cohort()`'s output.

Usage

```
cohort_attrition(paths, min_los_hours = 24L)
```

Arguments

`paths`	Output of `pic_paths()`.
`min_los_hours`	See `build_cohort()`.

Value

A `data.table` of (step, n, excluded, reason).

Examples

```
## Not run:
paths <- pic_paths()
attrition <- cohort_attrition(paths, min_los_hours = 24L)
print(attrition)

## End(Not run)
```

compute_pim3 *Reconstruct PIM3 from PIC fields*

Description

[Experimental]

Computes the Pediatric Index of Mortality 3 (Straney et al. 2013) from PIC `CHARTEVENTS` and the `cohort` table. Components that cannot be recovered from PIC default to 0 (Straney convention) and are listed in `proxy_flags`.

Demonstrating the linear-predictor calculation without PIC source files (this is what `compute_pim3()` produces internally per patient). This is the ANZICS PIM2 & PIM3 booklet's worked example (Jan 2019, p. 11): a 6 y-old girl, leukaemia post-1st-induction, intubated, SBP 70 mmHg, PaO2 65 mmHg, FiO2 0.7, base excess −12 mmol/L, reactive pupils, non-elective. The booklet gives PIM3val = −0.11114 and risk of death = 47.22%.

```
beta <- picMort:::pim3_coefficients()
logit <-
  beta$intercept +                       # -1.7928
  beta$pupils_fixed          * 0 +        # reactive pupils
  beta$elective              * 0 +        # non-elective
  beta$mech_vent             * 1 +        # intubated
  beta$base_excess_abs       * 12 +       # |-12|
  beta$sbp_linear            * 70 +       # SBP 70 mmHg
  beta$sbp_squared_over_1000 * (70 * 70 / 1000) +
  beta$fio2_pao2             * (100 * 0.7 / 65) +  # 100*FiO2/PaO2
  beta$recov_card_byp        * 0 +
  beta$recov_card_nonbyp     * 0 +
  beta$recov_noncard         * 0 +
  beta$very_high_risk_dx     * 1          # leukaemia after 1st induction
stats::plogis(logit)  #   0.4722
```

Usage

```
compute_pim3(cohort, paths, window_hours = 1L)
```

Arguments

`cohort`	A cohort `data.table` from `build_cohort()`.
`paths`	Output of `pic_paths()`.
`window_hours`	Time window in hours for first-hour PIM3 components. Straney uses 1 hour; we follow.

Value

A `data.table` keyed on `icustay_id` with columns: `pim3_logit` (numeric), `pim3` (probability), `risk_group` (factor: low / default / high / very_high), `sbp_used` (numeric), `proxy_flags` (list-column; vector of components that defaulted).

References

Straney L et al. (2013). PIM3: an updated Pediatric Index of Mortality. Pediatr Crit Care Med 14(7):673-681.

Examples

```
## Not run:
paths    <- pic_paths()
cohort   <- build_cohort(paths, min_los_hours = 24L, verbose = FALSE)
pim3_tbl <- compute_pim3(cohort, paths)
summary(pim3_tbl$pim3)
table(pim3_tbl$risk_group)

## End(Not run)
```

`decision_curve` *Decision-curve analysis at clinically meaningful thresholds*

Description

[Experimental]

Computes net benefit per threshold per model alongside the "treat-all" / "treat-none" references. Net benefit is computed analytically (no external package dependency); equivalent to `dcurves::dca()`.

Usage

```
decision_curve(
  probs,
  y,
  thresholds = c(0.05, 0.1, 0.2),
  plot_grid = TRUE,
  n_boot = 0L,
  seed = 20260508L
)
```

Arguments

`probs`	Named list of predicted-probability vectors.
`y`	Outcome (0/1 integer).
`thresholds`	Numeric vector of threshold probabilities; default `c(0.05, 0.10, 0.20)` plus a fine grid for plotting.
`plot_grid`	If `TRUE` (default), additionally returns a fine grid of thresholds (every 1 %) for plot rendering.
`n_boot`	Bootstrap replicates for paired CIs on net benefit at the prespecified `thresholds` and on the Brier skill score. Default 0 (no CIs); typical production value 1000.
`seed`	Integer seed for the bootstrap (used only when `n_boot > 0`). Default 20260508.

Value

A `data.table` of (model, threshold, net_benefit, type) when `n_boot = 0`, where `type` is `"model"`, `"all"`, or `"none"`. When `n_boot > 0`, additional rows with `(model, metric, estimate, lower, upper)` columns carry paired bootstrap 95 % CIs on net benefit at each prespecified threshold and on the Brier skill score.

Examples

```
set.seed(20260517L)
n <- 200L
p_a <- stats::plogis(stats::rnorm(n, -2, 1))
p_b <- stats::plogis(stats::rnorm(n, -2, 1.4))
y   <- stats::rbinom(n, 1L, (p_a + p_b) / 2)
dca <- decision_curve(list(model_a = p_a, model_b = p_b), y,
                      thresholds = c(0.05, 0.10, 0.20),
                      plot_grid = FALSE)
dca
```

`default_recipe` *Recipe for preprocessing*

Description

[Experimental]

Returns a `recipes::recipe` encoding canonical preprocessing: median imputation for continuous predictors, mode imputation for categorical predictors, near-zero-variance drop, dummy-encoding, centring + scaling. Fit on training fold; applied to test fold via `prep()` / `bake()`.

Usage

```
default_recipe(x, y)
```

Arguments

`x`	Feature `data.table` from `build_features()`.
`y`	Outcome integer vector aligned to `x$icustay_id`.

Value

A `recipes::recipe` object.

Examples

```
if (requireNamespace("recipes", quietly = TRUE)) {
  set.seed(20260517L)
  x <- data.table::data.table(
    icustay_id = 1:20,
    age_years  = runif(20, 0, 18),
    hr_mean    = rnorm(20, 100, 15),
    sex_male   = rbinom(20, 1L, 0.5)
  )
  y <- rbinom(20, 1L, 0.1)
  rec <- default_recipe(x, y)
  class(rec)
}
```

`discrimination_metrics` *Discrimination metrics with bootstrap CIs (supporting role)*

Description

[Experimental]

AUROC, AUPRC, Brier score, and Brier skill score relative to a reference model.

Usage

```
discrimination_metrics(
  probs,
  y,
  reference = NULL,
  n_boot = 1000L,
  seed = 20260508L
)
```

Arguments

`probs`	Named list of predicted-probability vectors aligned to `y`.
`y`	Outcome (0/1 integer).
`reference`	Name in `names(probs)` to use for the Brier-skill- score reference. Default `"pim3"` if present, else first model.
`n_boot`	Bootstrap replicates; default 1000.
`seed`	Integer seed; default 20260508.

Value

A `data.table` of (model, metric, estimate, lower, upper).

Examples

```
set.seed(20260517L)
n <- 200L
p_a <- stats::plogis(stats::rnorm(n, -2, 1))
p_b <- stats::plogis(stats::rnorm(n, -2, 1.4))
y   <- stats::rbinom(n, 1L, (p_a + p_b) / 2)
discrimination_metrics(list(model_a = p_a, model_b = p_b), y,
                       reference = "model_a", n_boot = 50L)
```

`fit_bayes_horseshoe` *Fit Bayesian logistic regression with regularized horseshoe prior*

Description

[Experimental]

Uses `brms::brm()` with a Bernoulli likelihood and a regularized horseshoe prior on the coefficients (Carvalho 2010 / Piironen & Vehtari 2017). The prior shrinks irrelevant coefficients aggressively while leaving informative ones effectively unregularized; the posterior gives every patient a credible interval on predicted mortality probability – the package's headline functional output.

Usage

```
fit_bayes_horseshoe(
  features,
  train_idx,
  chains = 2L,
  iter = 1000L,
  par_ratio = 0.1,
  adapt_delta = 0.95,
  cores = chains,
  seed = 20260508L
)
```

Arguments

`features`	Output of `build_features()`.
`train_idx`	Integer vector of training-row indices.
`chains`	MCMC chains. Default 2.
`iter`	Total iterations per chain (warmup + sampling). Default 1000.
`par_ratio`	Prior on the proportion of non-zero coefficients (regularized-horseshoe `par_ratio`). Default 0.1 (~10 % expected to be non-zero from ~100 candidate predictors).
`adapt_delta`	NUTS adaptation. Default 0.95.
`cores`	Number of cores. Default = `chains`.
`seed`	Integer seed; default 20260508.

Value

A list with `model` (a `brmsfit`), `prep`, `predictors`, `chains`, `iter`, `seed`, and `type = "bayes_horseshoe"`.

Examples

```
# `fit_bayes_horseshoe()` compiles a Stan model via `brms`. Compilation
# alone runs 30 s - 3 min and a minimal 1-chain / 200-iter sample
# another 1-5 min, so the example is wrapped in `\dontrun{}` rather than
# `\donttest{}` -- `R CMD check --run-donttest` would otherwise need a
# fully provisioned rstan toolchain on every CI worker. The brms smoke
# test in `tests/testthat/test-fits.R` runs only when the environment
# variable `PICMORT_TEST_BAYES=true` is set.
## Not run:
paths    <- pic_paths()
cohort   <- build_cohort(paths, min_los_hours = 24L, verbose = FALSE)
features <- build_features(cohort, paths, window_hours = 24L)
split    <- make_train_test_split(features)
fit      <- fit_bayes_horseshoe(features, split$train_idx,
                                chains = 1L, iter = 200L)
fit$type

## End(Not run)
```

`fit_elastic_net` *Fit penalized logistic regression (elastic net)*

Description

[Experimental]

Inner CV over the (alpha, lambda) grid; class-weighted to handle the ~7-9 % pediatric ICU mortality rate. Selects best alpha by minimum CV deviance and uses `lambda.1se` for parsimony.

Usage

```
fit_elastic_net(
  features,
  train_idx,
  alpha_grid = seq(0, 1, by = 0.2),
  nfolds = 5L,
  seed = 20260508L
)
```

Arguments

`features`	Output of `build_features()`.
`train_idx`	Integer vector of training-row indices.
`alpha_grid`	Numeric vector of alpha values; default `seq(0, 1, 0.2)`.
`nfolds`	Inner CV folds; default 5.
`seed`	Integer seed; default 20260508.

Value

A list with `model`, `prep`, `predictors`, `best_alpha`, `best_lambda`, `cv_log`, and `type = "glmnet"`.

Examples

```
## Not run:
paths    <- pic_paths()
cohort   <- build_cohort(paths, min_los_hours = 24L, verbose = FALSE)
features <- build_features(cohort, paths, window_hours = 24L)
split    <- make_train_test_split(features)
fit      <- fit_elastic_net(features, split$train_idx)
fit$best_alpha
fit$best_lambda

## End(Not run)
```

`fit_xgboost` *Fit XGBoost classifier*

Description

[Experimental]

Inner CV over a small principled grid. Class imbalance handled via `scale_pos_weight = n_neg / n_pos`. Returns the best parameter set and the corresponding fitted model.

Usage

```
fit_xgboost(features, train_idx, grid = NULL, nfolds = 5L, seed = 20260508L)
```

Arguments

`features`	Output of `build_features()`.
`train_idx`	Integer vector of training-row indices.
`grid`	`data.frame` of hyperparameter combinations. If `NULL`, uses the canonical 4-row grid documented in `vignettes/paper1_baseline.Rmd`.
`nfolds`	Inner CV folds; default 5.
`seed`	Integer seed; default 20260508.

Value

A list with `model`, `prep`, `predictors`, `best_params`, `best_nrounds`, `cv_log`, `scale_pos_weight`, and `type = "xgboost"`.

Examples

```
## Not run:
paths    <- pic_paths()
cohort   <- build_cohort(paths, min_los_hours = 24L, verbose = FALSE)
features <- build_features(cohort, paths, window_hours = 24L)
split    <- make_train_test_split(features)
fit      <- fit_xgboost(features, split$train_idx)
fit$best_nrounds

## End(Not run)
```

---

icd10_to_chapter *Map an ICD-10 code to its WHO chapter*

Description

[Experimental]

Vectorised. Accepts either ICD-10 codes with a decimal point (e.g. `"K52.901"`, `"J18.900"`) or without (e.g. `"K52901"`). The chapter is determined by the leading letter and the two leading digits per the ICD-10 chapter ranges.

Usage

```
icd10_to_chapter(code)
```

Arguments

`code`	Character vector of ICD-10 codes. `NA` codes return `NA`.

Value

Character vector of chapter names; one of: `"infectious"`, `"neoplasms"`, `"blood"`, `"endocrine"`, `"mental"`, `"nervous"`, `"eye"`, `"ear"`, `"circulatory"`, `"respiratory"`, `"digestive"`, `"skin"`, `"musculoskeletal"`, `"genitourinary"`, `"pregnancy"`, `"perinatal"`, `"congenital"`, `"symptoms"`, `"injury"`, `"external"`, `"factors"`, `"special"`, `"unknown"`.

Examples

```
icd10_to_chapter(c("J18.900", "K52901", "C91.0", "I46", NA_character_))
# Round-trips with or without the decimal separator
identical(icd10_to_chapter("J18.900"), icd10_to_chapter("J18900"))
```

make_train_test_split *Stratified train / test split*

Description

[Experimental]

Outcome-stratified single split into training and test folds via `rsample::initial_split()`. Returns 1-based integer indices into `features$y` (and rows of `features$x`).

Usage

```
make_train_test_split(features, prop = 0.7, seed = 20260508L)
```

Arguments

`features`	Output of `build_features()`.
`prop`	Proportion in the training fold. Default 0.7.
`seed`	Integer seed; default 20260508.

Value

A list with `train_idx`, `test_idx` (integer vectors).

Examples

```
if (requireNamespace("rsample", quietly = TRUE)) {
  set.seed(20260517L)
  features <- list(y = c(rep(0L, 90), rep(1L, 10)))
  split <- make_train_test_split(features, prop = 0.7, seed = 1L)
  length(split$train_idx) + length(split$test_idx) # 100
  mean(features$y[split$train_idx]) # ~ 0.10
}
```

pic_paths *Resolve PIC v1.1.0 source paths*

Description

[Experimental]

Returns a named list of canonical paths to PIC CSVs. If the `PICMORT_DATA_DIR` environment variable is set, it is treated as the directory containing the PIC v1.1.0 CSVs. Otherwise paths are resolved through the project-local `data_links/pic_v110/` symlink.

Usage

```
pic_paths(root = NULL, check = TRUE)
```

Arguments

root	Path to the directory containing data_links/. Defaults to PICMORT_DATA_DIR when that environment variable is set, or to the project root containing data_links/ otherwise.
check	Logical. If TRUE (default), verifies that every referenced CSV exists and is readable.

Value

Named list with elements admissions, patients, icustays, chartevents, labevents, diagnoses_icd, d_items, d_icd_diagnoses, d_labitems, prescriptions, inputevents, outputevents, microbiologyevents, surgery_vital_signs, or_exam_reports, emr_symptoms.

Examples

```
# Requires the registered PIC v1.1.0 source, either via
# `PICMORT_DATA_DIR` or `<project root>/data_links/pic_v110/`.
# Run `check = FALSE` to inspect the expected file names without
# touching disk.
## Not run:
paths <- pic_paths()
names(paths)

## End(Not run)

# File-name discovery without verifying existence
tmp <- tempfile(); dir.create(file.path(tmp, "data_links", "pic_v110"),
                              recursive = TRUE)
paths <- pic_paths(root = tmp, check = FALSE)
names(paths)
unlink(tmp, recursive = TRUE)
```

---

pim3_face_validity *PIM3 face-validity check*

Description

[Experimental]

Reports the distribution of reconstructed PIM3 probabilities, the observed-vs-expected (O/E) ratio against actual cohort mortality, and a summary of which components defaulted to proxy values. The result is informational, not a hard gate – imperfect PIM3 reconstruction is normal when all components are not recoverable from the source database; the manuscript Limitations records which components were proxied.

Usage

```
pim3_face_validity(pim3_tbl, cohort)
```

Arguments

`pim3_tbl`	Output of `compute_pim3()`.
`cohort`	Cohort `data.table` with `hospital_expire_flag`.

Value

A list with elements `summary` (named numeric of pim3 distribution stats), `oe_ratio`, `oe_ci` (Wilson 95% CI), `proxy_freq` (table of which proxies fired and how often), `risk_group_counts` (table), `notes` (character).

Examples

```
toy <- readRDS(system.file("extdata", "toy_cohort.rds",
                           package = "picMort"))
# Build a minimal synthetic pim3 table matching compute_pim3()'s schema.
set.seed(20260517L)
pim3_tbl <- data.table::data.table(
  icustay_id  = toy$icustay_id,
  pim3_logit  = stats::rnorm(nrow(toy), mean = -2.5, sd = 0.5),
  pim3        = stats::plogis(stats::rnorm(nrow(toy), -2.5, 0.5)),
  risk_group  = factor(sample(c("default", "low", "high", "very_high"),
                              nrow(toy), replace = TRUE,
                              prob = c(0.7, 0.2, 0.08, 0.02)),
                       levels = c("default", "low", "high", "very_high")),
  sbp_used    = stats::rnorm(nrow(toy), 110, 20),
  proxy_flags = replicate(nrow(toy),
                          c("fio2_pao2", "base_excess"), simplify = FALSE)
)
fv <- pim3_face_validity(pim3_tbl, toy)
fv$summary
fv$oe_ratio
```

`pim3_risk_group` *PIM3 risk-group ICD-10 mapping (pediatric, simplified)*

Description

[Experimental]

Maps an ICD-10 code to one of `"low"`, `"high"`, `"very_high"`, or `NA_character_` (no risk-group assignment). Covers the most frequent pediatric admissions per Straney 2013 Tables 2-4. Codes not in the lookup return `NA`, treated as default-risk for PIM3.

Usage

```
pim3_risk_group(code)
```

Arguments

code	Character vector of ICD-10 codes.

Value

Character vector of risk-group labels (or NA).

Examples

```
pim3_risk_group(c("J45.0",   # asthma -> low
                  "I42.1",   # cardiomyopathy -> high
                  "I46",     # post-cardiac-arrest -> very_high
                  "K52.901", # unmapped -> NA
                  NA_character_))
```

---

plot_calibration *Render the headline calibration plot*

Description

[Experimental]

Smoothed loess calibration curves per model with the ideal diagonal. Returns a ggplot object; save via ggplot2::ggsave().

Usage

```
plot_calibration(calib_list)
```

Arguments

calib_list	Named list; each element is the output of calibration_suite() for one model.

Value

A ggplot object.

Examples

```
if (requireNamespace("ggplot2", quietly = TRUE)) {
  set.seed(20260517L)
  n <- 200L
  prob <- stats::plogis(stats::rnorm(n, -2, 1))
  y    <- stats::rbinom(n, 1L, prob)
  cal  <- calibration_suite(prob, y, n_boot = 25L)
  p    <- plot_calibration(list(model_a = cal))
  class(p)
}
```

plot_decision_curve *Render the headline decision-curve plot*

Description

[Experimental]

Net benefit vs threshold across models, with treat-all and treat-none references.

Usage

```
plot_decision_curve(dca)
```

Arguments

dca	A `data.table` from `decision_curve()` (with `plot_grid = TRUE`).

Value

A `ggplot` object.

Examples

```
if (requireNamespace("ggplot2", quietly = TRUE)) {
  set.seed(20260517L)
  n <- 200L
  p_a <- stats::plogis(stats::rnorm(n, -2, 1))
  y   <- stats::rbinom(n, 1L, p_a)
  dca <- decision_curve(list(model_a = p_a), y, plot_grid = TRUE)
  p   <- plot_decision_curve(dca)
  class(p)
}
```

predict_mortality *Predict mortality probabilities (unified interface)*

Description

[Experimental]

Routes to the appropriate predict implementation based on `model$type`. Returns a `data.table` with columns:

- `prob_raw` – model output (probability of in-hospital mortality)
- `prob_calibrated` – post-hoc calibrated (only XGBoost; same as raw otherwise)
- `prob_lower`, `prob_upper` – 95 % credible interval (Bayesian only)

Usage

```
predict_mortality(model, new_data)
```

Arguments

`model`	Fit object from one of the `fit_*()` functions, OR a PIM3 `data.table` from `compute_pim3()`.
`new_data`	New `features$x`-style `data.table` (with `icustay_id` and the same columns as the training features).

Value

A `data.table` aligned to `nrow(new_data)`.

Examples

```
# PIM3 dispatch branch: predict_mortality() also accepts the PIM3
# data.table returned by compute_pim3(), bypassing model fits.
toy <- readRDS(system.file("extdata", "toy_cohort.rds",
                           package = "picMort"))
pim3_tbl <- data.table::data.table(
  icustay_id = toy$icustay_id,
  pim3       = stats::plogis(stats::rnorm(nrow(toy), mean = -2.5, sd = 0.5))
)
preds <- predict_mortality(pim3_tbl, toy[1:5, ])
preds

## Not run:
# Full model-fit dispatch (requires the registered PIC data)
paths    <- pic_paths()
cohort   <- build_cohort(paths, min_los_hours = 24L, verbose = FALSE)
features <- build_features(cohort, paths, window_hours = 24L)
split    <- make_train_test_split(features)
fit      <- fit_elastic_net(features, split$train_idx)
predict_mortality(fit, features$x[split$test_idx, ])

## End(Not run)
```

`subgroup_performance` *Subgroup performance table*

Description

[Experimental]

Reports calibration + AUROC per pre-registered subgroup: age strata (<1, 1-5, 6-12, 13-18 y), surgical vs medical, and top-3 ICD chapters by frequency. Cells with fewer than `n_min_events` events are suppressed.

Usage

```
subgroup_performance(probs, cohort_test, n_min_events = 5L)
```

Arguments

`probs`	Named list of predicted-probability vectors aligned to `cohort_test`.
`cohort_test`	Cohort test fold (a `data.table` with `age_years`, `is_surgical`, `primary_icd_chapter`, `hospital_expire_flag`).
`n_min_events`	Minimum events per cell. Default 5.

Value

A `data.table` of (model, subgroup_var, level, n, n_events, auroc, ici, cal_slope).

Examples

```
toy <- readRDS(system.file("extdata", "toy_cohort.rds",
                           package = "picMort"))
set.seed(20260517L)
probs <- list(model_a = stats::runif(nrow(toy)))
# n_min_events = 1L because the toy cohort only has 8 deaths.
subgroup_performance(probs, toy, n_min_events = 1L)
```

Index