Package 'SLOS'

November 18, 2026

Type Package

Title ICU Length of Stay Prediction and Efficiency Evaluation

Version 2.0.0

Author Igor Peres [aut],
Joana da Matta [cre]

Maintainer Joana da Matta <joana.damatta02@gmail.com>

Description Provides tools for predicting intensive care unit (ICU) length of stay and assessing ICU efficiency with the standardized length of stay ('SLOS') ratio, following the methodology of Peres et al. (2022) <doi:10.1016/j.accpm.2022.101142> and Peres et al. (2023) <doi:10.1007/s00134-023-07255-w>. The package supports a pretrained model for Brazilian intensive care units, as well as a complete pipeline for building a new model from local data: feature screening and multiple imputation, recursive feature elimination, training of a stacked ensemble (linear model and random forest), and unit-level evaluation with funnel plots. More information: <https://github.com/igor-peres/ICU-Length-of-Stay-Prediction>.

License MIT + file LICENSE

URL https://github.com/igor-peres/ICU-Length-of-Stay-Prediction

BugReports https://github.com/igor-peres/ICU-Length-of-Stay-Prediction/issues

Depends R (>= 3.5.0)

Imports caret, caretEnsemble, curl, DescTools, dplyr, ems, ggplot2, graphics, httr, magrittr, mice, MLmetrics, ranger, rlang, stats, utils

Suggests e1071, ipred, knitr, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

Encoding UTF-8

LazyData true

RoxygenNote 7.3.1

NeedsCompilation no

Repository CRAN

Date/Publication 2026-11-18 10:00:33 UTC

Contents

sampled_data 2
SLOS 3
slos_apply_feature_selection 4
slos_feature_select_rfe 5
slos_load_pretrained_model 7
slos_predict_pretrained 8
slos_pre_processing 9
slos_train_new_model 11

Index **14**

`sampled_data` *Sampled ICU admissions data*

Description

An anonymized sample of 1000 ICU admissions used to illustrate and test the package. It contains the predictors expected by the pre-trained SLOS model, the ICU identifier (`UnitCode`) and the observed, truncated ICU length of stay (`UnitLengthStay_trunc`, in days).

Usage

```
data(SampledData)
```

Format

A data frame with 1000 rows (one per admission). Among its columns:

UnitCode ICU identifier.

UnitLengthStay_trunc Observed ICU length of stay, in days, truncated (outcome).

... Admission-level predictors used by the model.

Source

Anonymized sample prepared by the package authors.

SLOS *Standardized length of stay (SLOS) per ICU*

Description

Core function of the package. It predicts the length of stay of every patient, aggregates observed and expected length of stay by ICU, computes each unit's SLOS (observed / expected), draws a funnel plot and builds a plot comparing the sum of observed and predicted length of stay per unit.

Usage

```
SLOS(data, icu_column, outcome, model = NULL, show_funnel = TRUE)
```

Arguments

`data`	A data frame containing the test data: the model predictors, the outcome and the ICU identifier.
`icu_column`	Name of the column that identifies the ICU, as a string.
`outcome`	Name of the outcome column (observed ICU length of stay), as a string.
`model`	Optional. A model returned by `slos_train_new_model()`. If `NULL` (default), the pre-trained model is downloaded with `slos_load_pretrained_model()` (requires an internet connection and a download of about 1.1 GB).
`show_funnel`	Logical. If `TRUE` (default), the funnel plot is drawn on the active graphics device. It can always be drawn again with `plot(result$funnel_plot)`.

Value

A list with:

df_unit_slos Data frame with one row per ICU: number of admissions (`admissoes`), sum of observed length of stay (`soma_los_obs`), sum of expected length of stay (`soma_los_esp`) and `SLOS`.

plot_SLOS_obs_prev A `ggplot` object comparing the sum of observed and predicted length of stay per unit.

funnel_plot The funnel plot object created by `ems::funnel()`.

Examples

```
if (requireNamespace("e1071", quietly = TRUE)) {
  # Train a small model so that the example runs quickly
  dat <- sampled_data[, setdiff(names(sampled_data),
                                c("UnitLengthStay_days", "admissionid"))]
  idx <- caret::createDataPartition(dat$UnitLengthStay_trunc,
                                    p = 0.8, list = FALSE)
  fit <- slos_train_new_model(
    dat[idx, ], outcome = "UnitLengthStay_trunc", icu_column = "UnitCode",
    cv_folds = 2, verbose_iter = FALSE,
```

```
    base_ranger_grid = data.frame(.mtry = 5, .splitrule = "variance",
                                  .min.node.size = 5),
    meta_grid = expand.grid(mtry = 2, min.node.size = 5,
                            splitrule = "variance")
  )

  result <- SLOS(dat[-idx, ], icu_column = "UnitCode",
                 outcome = "UnitLengthStay_trunc", model = fit,
                 show_funnel = FALSE)
  head(result$df_unit_slos)
  result$plot_SLOS_obs_prev
  plot(result$funnel_plot)
}


## Not run:
# With the pre-trained model (downloads about 1.1 GB on first use)
result <- SLOS(sampled_data, icu_column = "UnitCode",
               outcome = "UnitLengthStay_trunc")

## End(Not run)
```

`slos_apply_feature_selection` *Apply SLOS Feature Selection to New Data*

Description

Given a fitted `slos_rfe_result`, apply the same pre-processing (if any) and reduce `newdata` to the selected predictors in the correct order.

Usage

```
slos_apply_feature_selection(
  newdata,
  fs_result,
  strict = TRUE,
  fill_missing = c("none", "zero", "na"),
  verbose = TRUE
)
```

Arguments

`newdata`	A `data.frame`/`tibble` with at least the selected columns.
`fs_result`	An object returned by `slos_feature_select_rfe()`.
`strict`	Logical; if `TRUE`, error when selected columns are missing in `newdata` (unless `fill_missing` is not `"none"`). If `FALSE`, missing columns are filled according to `fill_missing` or, when that is `"none"`, dropped with a warning. Default `TRUE`.

`fill_missing`	How to fill selected columns that are absent from `newdata`: `"none"` (default), `"zero"` or `"na"`.
`verbose`	Logical; if `TRUE` (default), warn about missing columns and about unseen factor levels converted to `NA`.

Value

A data.frame with (optionally normalized) predictors restricted to the selected variables in the same order the RFE produced.

See Also

`slos_feature_select_rfe()` for a complete example.

`slos_feature_select_rfe` *Recursive Feature Elimination (RFE) for ICU Length of Stay (LOS) Prediction*

Description

A wrapper around `caret::rfe()` to perform feature selection for ICU Length of Stay modeling. Supports Treebag (`treebag`), Random Forest (`ranger` or `rf`), optional normalization (via `caret::preProcess`), and k-fold cross-validation. Returns selected variables and artifacts so you can consistently transform new datasets.

Usage

```
slos_feature_select_rfe(
  data = NULL,
  outcome = NULL,
  x = NULL,
  y = NULL,
  method = c("ranger", "treebag", "rf"),
  sizes = NULL,
  normalize = FALSE,
  preproc_methods = c("range"),
  cv_folds = 5,
  metric = "RMSE",
  seed = NULL,
  verbose = TRUE,
  tune_length = 5,
  num_trees = 500,
  importance = "permutation",
  respect_unordered_factors = "order",
  min_node_size = NULL,
  splitrule = NULL,
  ...
)
```

Arguments

data	Optional data.frame/tibble containing predictors **and** the outcome.
outcome	Optional single string naming the outcome column in data.
x	Optional predictor matrix/data.frame. If provided, y must be provided too.
y	Optional numeric vector outcome. Required when x is provided.
method	Character scalar: "ranger" (default), "treebag" (requires the **ipred** package) or "rf" (requires **randomForest**).
sizes	Integer vector of subset sizes to evaluate during RFE. If NULL (default), a grid is built from the number of predictors p: 1:min(30, p) plus 25%, 50%, 75% and 100% of p.
normalize	Logical; if TRUE, fit a preProcess object on predictors and use it inside RFE. Default FALSE for trees.
preproc_methods	Character vector passed to caret::preProcess. Default c("range").
cv_folds	Integer; number of CV folds for RFE (rfeControl(method="cv")). Default 5.
metric	Optimization metric for RFE, usually "RMSE" for regression. Default "RMSE".
seed	Integer for reproducibility (passed to set.seed()). If NULL (default), 2025 is used for "ranger", 420 for "treebag" and 100 for "rf".
verbose	Logical; print progress. Default TRUE.
tune_length	Integer; passed to caret::train() when method = "ranger" to control the number of mtry values tried. Default 5.
num_trees	Integer; number of trees for ranger (passed as num.trees). Default 500.
importance	Character; variable-importance type for ranger: "none", "impurity", or "permutation" (default).
respect_unordered_factors	Character; how ranger handles unordered factors: e.g., "order" (default). See ranger::ranger().
min_node_size	Optional integer; ranger minimal node size (passed as min.node.size). If NULL, caret's default is used.
splitrule	Optional character; ranger split rule (e.g., "variance" for regression). If NULL, caret's default is used.
...	Additional arguments forwarded to caret::train() (only relevant when method = "ranger", e.g., tuneGrid, maximize, metric, etc.).

Value

An object of class slos_rfe_result, a list with:

selected_vars Character vector of selected predictor names.

rfe The caret::rfe result object.

preproc The caret::preProcess object if normalization was used; else NULL.

meta List of metadata (method, sizes, cv_folds, metric, normalize, timestamp).

Examples

```
if (requireNamespace("ipred", quietly = TRUE)) {
  pp <- slos_pre_processing(
    sampled_data, outcome = "UnitLengthStay_trunc",
    do_impute = TRUE, m = 1, maxit = 1
  )
  train <- pp$imputation$train_completed
  train <- train[, setdiff(names(train),
                           c("UnitLengthStay_days", "admissionid", "icuid",
                             "UnitCode"))]

  res <- slos_feature_select_rfe(
    data = train, outcome = "UnitLengthStay_trunc",
    method = "treebag", sizes = c(5, 10), cv_folds = 2, verbose = FALSE
  )
  res$selected_vars

  # Keep only the selected columns in the test set
  x_test <- slos_apply_feature_selection(pp$imputation$test_completed, res)
}
```

`slos_load_pretrained_model` *Load the pre-trained SLOS model*

Description

Downloads the pre-trained SLOS model from the project's GitHub release and returns it. **The file is large (about 1.1 GB)**, so the first call can take a long time. The file is stored in a temporary location and the loaded model is cached for the rest of the R session, so repeated calls do not download it again. Nothing is written outside of `tempdir()`.

Usage

```
slos_load_pretrained_model(cache = TRUE, timeout = 3600)
```

Arguments

`cache`	Logical. If `TRUE` (default), reuse the model already downloaded in this session, if any.
`timeout`	Download time limit, in seconds. Default `3600`.

Details

An internet connection is required. If the model cannot be downloaded, an informative error is raised.

Value

The pre-trained model (a `caretStack` object).

Examples

```
## Not run:
# Downloads about 1.1 GB
model <- slos_load_pretrained_model()
class(model)

## End(Not run)
```

`slos_predict_pretrained` *Predict ICU length of stay with the pre-trained SLOS model*

Description

Makes patient-level predictions and evaluates RMSE, MAE and R2. If `model` is not provided, the pre-trained model is downloaded with `slos_load_pretrained_model()`.

Usage

```
slos_predict_pretrained(data, outcome = "UnitLengthStay_trunc", model = NULL)
```

Arguments

`data`	A data frame with the predictors used by the model and the outcome column.
`outcome`	The outcome's column name, as a string. Default `"UnitLengthStay_trunc"`.
`model`	Optional. A fitted `caretStack` model compatible with `data`. If `NULL`, the pre-trained model is used.

Value

A list with:

predictions data.frame with one column, `predictions`.

comparison data.frame with columns `Observations` and `pred`.

RMSE, MAE, R2 Numeric performance metrics.

Examples

```
## Not run:
# Downloads the pre-trained model (about 1.1 GB) on first use
res <- slos_predict_pretrained(sampled_data,
                               outcome = "UnitLengthStay_trunc")
res$RMSE

## End(Not run)
```

`slos_pre_processing` *End-to-end data pre-processing for ICU LOS modeling (screening + optional imputation)*

Description

Performs feature screening and optional multiple imputation in one step. Screening includes:

- Near-zero variance detection via `caret::nearZeroVar` (records *freqRatio* and *percentUnique*).
- High pairwise correlation among numeric predictors (records the max partner correlation for removed vars).
- High pairwise association among categorical predictors using Cramér's V (records max partner association).

Optionally splits the data into train/test with `caret::createDataPartition` and performs multiple imputation using **mice**, returning completed datasets plus diagnostics.

Usage

```
slos_pre_processing(
  data,
  outcome,
  remove_nzv = TRUE,
  nzv_freq_cut = 50,
  remove_corr_num = TRUE,
  corr_cutoff_num = 0.9,
  remove_corr_cat = TRUE,
  corr_cutoff_cat = 0.9,
  do_impute = FALSE,
  p_train = 0.8,
  include_outcome = FALSE,
  mincor = 0.1,
  m = 5,
  maxit = 5,
  method = "pmm",
  seed = 420,
  return_mids = FALSE,
  return_matrices = FALSE
)
```

Arguments

`data`	A `data.frame`/tibble with predictors and the outcome.
`outcome`	Single string with the outcome column name.
`remove_nzv`	Logical. Remove near-zero variance predictors? Default `TRUE`.
`nzv_freq_cut`	Numeric `freqCut` passed to `caret::nearZeroVar`. Default `50`.

`remove_corr_num`
: Logical. Remove highly correlated numeric predictors? Default `TRUE`.

`corr_cutoff_num`
: Numeric absolute correlation cutoff for numeric predictors. Default `0.9`.

`remove_corr_cat`
: Logical. Remove highly associated categorical predictors (Cramér's V)? Default `TRUE`.

`corr_cutoff_cat`
: Numeric cutoff for Cramér's V among categorical predictors. Default `0.9`.

`do_impute`
: Logical. If `TRUE`, perform **mice** imputation on train and test partitions. Default `FALSE`.

`p_train`
: Proportion for training set when splitting with `createDataPartition`. Default `0.8`.

`include_outcome`
: Logical. If `TRUE`, allow the outcome to be used as predictor during imputation (included in `quickpred`). Default `FALSE`.

`mincor`
: Minimum absolute correlation/association threshold used by `mice::quickpred` to include predictors. Default `0.1`.

`m`
: Number of multiple imputations (`mice::mice`). Default `5`.

`maxit`
: Maximum iterations per imputation. Default `5`.

`method`
: Named vector or single string for **mice** methods (e.g., `"pmm"`, `"norm"`, `"rf"`). Default `"pmm"`.

`seed`
: Integer RNG seed for reproducibility. Default `420`.

`return_mids`
: Logical. If `TRUE`, returns the `mids` objects for train/test. Default `FALSE`.

`return_matrices`
: Logical. If `TRUE`, returns full correlation/Cramér's V matrices (can be large). Default `FALSE`.

Value

A list with:

data_screened The screened dataset (with outcome).

removed List with vectors `$nzv`, `$corr_num`, `$corr_cat`.

screening_metrics
- `$nzv_table`: data.frame with variables removed by NZV and their *freqRatio*, *percentUnique*.
- `$num_removed_table`: data.frame with *variable*, *max_abs_cor_partner*, *max_abs_cor*.
- `$cat_removed_table`: data.frame with *variable*, *max_cramer_partner*, *max_cramerV*.
- `$num_cor_mat` and/or `$cat_cramer_mat` if `return_matrices = TRUE`.

split List with `$train_idx` and `$test_idx`.

imputation Present only when `do_impute=TRUE`. Contains:
- `$train_completed`, `$test_completed` completed data.frames.
- `$predictor_matrix`: the **mice** predictor matrix (importance matrix).
- `$methods`: named vector of imputation methods actually used by **mice**.

- $missingness_before: per-variable missing-rate table (train/test).
- $pm_density: proportion of non-zeros in the predictor matrix.
- $args: m, maxit, mincor, include_outcome, seed.
- $mids (optional): train_mids, test_mids if return_mids=TRUE.

Examples

```
data(SampledData)
pp_imp <- slos_pre_processing(
  data = sampled_data, outcome = "UnitLengthStay_trunc",
  do_impute = TRUE, include_outcome = FALSE,
  m = 1, maxit = 1, method = "pmm", seed = 42
)

head(pp_imp$imputation$missingness_before$train)

dim(pp_imp$imputation$train_completed)
dim(pp_imp$imputation$test_completed)

colSums(is.na(pp_imp$imputation$train_completed))
colSums(is.na(pp_imp$imputation$test_completed))

pp_imp$imputation$predictor_matrix[1:6, 1:6]
pp_imp$imputation$pm_density
```

---

slos_train_new_model *Train and stack caret models for ICU LOS (LM + Ranger -> Ranger stack)*

Description

Fits a base ensemble (linear model and random forest) with caretEnsemble::caretStack() (default: ranger). Optionally evaluates on a testing set and saves the final stacked model to disk.

Usage

```
slos_train_new_model(
  train,
  outcome = "UnitLengthStay_trunc",
  icu_column = "icuid",
  test = NULL,
  metric = "RMSE",
  cv_folds = 5,
  cv_method = "cv",
  verbose_iter = TRUE,
  seed = 998,
  base_ranger_grid = NULL,
```

```
  meta_method = "ranger",
  meta_grid = NULL,
  save_path = NULL,
  drop_single_level = TRUE,
  encode_categoricals = TRUE,
  continue_on_fail = TRUE
)
```

Arguments

train	A data.frame with predictors and the outcome column.
outcome	Character scalar. Name of the outcome column used for training. Default "UnitLengthStay_trunc".
icu_column	Name of the column containing the ICU identification. Default "icuid"
test	Optional data.frame to evaluate on. Must contain the same predictors as train (at least the columns used by the final model) and the outcome column for metric computation.
metric	Metric optimized during training (passed to caret). Default "RMSE".
cv_folds	Number of CV folds. Default 5.
cv_method	Resampling method for trainControl(). Default "cv".
verbose_iter	Logical; print training progress. Default TRUE.
seed	Integer seed for reproducibility. Default 998.
base_ranger_grid	Optional data.frame for the base ranger model's grid (columns .mtry, .splitrule, .min.node.size). If NULL, uses: .mtry = 5:10, .splitrule = "variance", .min.node.size = 5.
meta_method	Stacking meta-learner method. Default "ranger".
meta_grid	Optional data.frame of hyperparameters for the meta-learner. If NULL, uses: expand.grid(mtry = 2, min.node.size = c(5,10,15,20), splitrule = c("variance","extratree
save_path	Optional file path to save the stacked model (.RData will be appended if missing). Only the caretStack object (saved under the name stacked_model) is written; to reuse the complete fit with SLOS(), save the returned list yourself with save(). If NULL, nothing is saved. Default NULL.
drop_single_level	Logical; drop factor predictors with <2 levels (computed on the full training data) before encoding. Default TRUE.
encode_categoricals	Logical; if TRUE, use dummyVars(fullRank=TRUE) to one-hot encode non-numeric predictors. Default TRUE.
continue_on_fail	Logical; pass through to caretEnsemble::caretList() so that failed base models don't abort the run. Default TRUE. (All other parameters are unchanged.)

Value

A list with:

stacked_model The fitted `caretStack` object.

base_models The fitted `caretList` of base learners.

train_control The `trainControl` used.

dummyVars The `caret::dummyVars` encoder fitted on the training data.

feature_names Names of the encoded predictors used by the model.

metrics If `test` provided: a list with `patient` (RMSE, MAE and R2 at patient level) and, when `icu_column` is present in `test`, `icu` (the same metrics on the per-ICU sums of observed and predicted length of stay).

predictions If `test` provided: numeric vector of predictions.

The returned list can be passed to `SLOS()` through its `model` argument.

Examples

```
if (requireNamespace("e1071", quietly = TRUE)) {
  dat <- sampled_data[, setdiff(names(sampled_data),
                                c("UnitLengthStay_days", "admissionid"))]
  idx <- caret::createDataPartition(dat$UnitLengthStay_trunc,
                                    p = 0.8, list = FALSE)

  # small grids so that the example runs quickly
  fit <- slos_train_new_model(
    dat[idx, ], outcome = "UnitLengthStay_trunc", icu_column = "UnitCode",
    test = dat[-idx, ], cv_folds = 2, verbose_iter = FALSE,
    base_ranger_grid = data.frame(.mtry = 5, .splitrule = "variance",
                                  .min.node.size = 5),
    meta_grid = expand.grid(mtry = 2, min.node.size = 5,
                            splitrule = "variance"),
    save_path = file.path(tempdir(), "slos_trained_stack.RData")
  )
  fit$metrics
}
```

Index

∗ **datasets**
sampled_data, 2

sampled_data, 2
SampledData *(*sampled_data*)*, 2
save(), *12*
SLOS, 3
SLOS(), *12*, *13*
slos_apply_feature_selection, 4
slos_feature_select_rfe, 5
slos_feature_select_rfe(), *4*, *5*
slos_load_pretrained_model, 7
slos_load_pretrained_model(), *3*, *8*
slos_pre_processing, 9
slos_predict_pretrained, 8
slos_train_new_model, 11
slos_train_new_model(), *3*

tempdir(), *7*