Package 'survdnn'

July 22, 2025

Title Deep Neural Networks for Survival Analysis Using 'torch'

Version 0.6.0

Description Provides deep learning models for right-censored survival data using the 'torch' backend. Supports multiple loss functions, including Cox partial likelihood, L2-penalized Cox, time-dependent Cox,
and accelerated failure time (AFT) loss. Offers a formula-based interface, built-in support for cross-validation,
hyperparameter tuning, survival curve plotting, and evaluation metrics such as the C-index, Brier score,
and integrated Brier score. For methodological details, see Kvamme et al. (2019) <https://www.jmlr.org/papers/v20/18-424.html>.

License MIT + file LICENSE

Encoding UTF-8

Depends R (>= 4.1.0)

Imports torch, survival, stats, utils, tibble, dplyr, purrr, tidyr, ggplot2, methods, rsample, cli, glue

Suggests testthat (>= 3.0.0), knitr, rmarkdown

RoxygenNote 7.3.2

Config/testthat/edition 3

BugReports https://github.com/ielbadisy/survdnn/issues

URL https://github.com/ielbadisy/survdnn

NeedsCompilation no

Author Imad EL BADISY [aut, cre]

Maintainer Imad EL BADISY <elbadisyimad@gmail.com>

Repository CRAN

Date/Publication 2025-07-22 11:11:46 UTC

Contents

`brier` *Brier Score for Right-Censored Survival Data at a Fixed Time*

Description

Computes the Brier score at a fixed time point using inverse probability of censoring weights (IPCW).

Usage

```
brier(object, pre_sp, t_star)
```

Arguments

`object`	A 'Surv' object with observed time and status.
`pre_sp`	A numeric vector of predicted survival probabilities at 't_star'.
`t_star`	The evaluation time point.

Value

A single numeric value representing the Brier score.

Examples

```
library(survival)
data(veteran, package = "survival")
mod <- survdnn(Surv(time, status) ~
age + karno + celltype, data = veteran, epochs = 50, verbose = FALSE)
pred <- predict(mod, newdata = veteran, type = "survival", times = c(30, 90, 180))
y <- model.response(model.frame(mod$formula, veteran))
survdnn::brier(y, pre_sp = pred[["t=90"]], t_star = 90)
```

`cindex_survmat` *Concordance Index from a Survival Probability Matrix*

Description

Computes the time-dependent concordance index (C-index) from a predicted survival matrix at a fixed time point. The risk is computed as ‘1 - S(t_star)‘.

Usage

```
cindex_survmat(object, predicted, t_star = NULL)
```

Arguments

`object`	A ‘Surv‘ object representing the observed survival data.
`predicted`	A data frame or matrix of predicted survival probabilities. Each column corresponds to a time point (e.g., ‘t=90‘, ‘t=180‘).
`t_star`	A numeric time point corresponding to one of the columns in ‘predicted‘. If ‘NULL‘, the last column is used.

Value

A single numeric value representing the C-index.

Examples

```
library(survival)
data(veteran, package = "survival")
mod <- survdnn(Surv(time, status) ~
age + karno + celltype, data = veteran, epochs = 50, verbose = FALSE)
pred <- predict(mod, newdata = veteran, type = "survival", times = c(30, 90, 180))
y <- model.response(model.frame(mod$formula, veteran))
cindex_survmat(y, pred, t_star = 180)
```

cv_survdnn *K-Fold Cross-Validation for survdnn Models*

Description

Performs cross-validation for a ‘survdnn‘ model using the specified evaluation metrics.

Usage

```
cv_survdnn(
  formula,
  data,
  times,
  metrics = c("cindex", "ibs"),
  folds = 5,
  .seed = NULL,
  ...
)
```

Arguments

formula	A survival formula, e.g., ‘Surv(time, status) ~ x1 + x2‘.
data	A data frame.
times	A numeric vector of evaluation time points.
metrics	A character vector: any of ‘"cindex"‘, ‘"brier"‘, ‘"ibs"‘.
folds	Integer. Number of folds to use.
.seed	Optional. Set random seed for reproducibility.
...	Additional arguments passed to [survdnn()].

Value

A tibble containing metric values per fold and (optionally) per time point.

Examples

```
library(survival)
data(veteran)
cv_survdnn(
  Surv(time, status) ~ age + karno + celltype,
  data = veteran,
  times = c(30, 90, 180),
  metrics = "ibs",
  folds = 3,
  .seed = 42,
  hidden = c(16, 8),
  epochs = 5
)
```

`evaluate_survdnn` *Evaluate a survdnn Model Using Survival Metrics*

Description

Computes evaluation metrics for a fitted ‘survdnn‘ model at one or more time points. Supported metrics include the concordance index (‘"cindex"‘), Brier score (‘"brier"‘), and integrated Brier score (‘"ibs"‘).

Usage

```
evaluate_survdnn(
  model,
  metrics = c("cindex", "brier", "ibs"),
  times,
  newdata = NULL
)
```

Arguments

`model`	A fitted ‘survdnn‘ model object.
`metrics`	A character vector of metric names: ‘"cindex"‘, ‘"brier"‘, ‘"ibs"‘.
`times`	A numeric vector of evaluation time points.
`newdata`	Optional. A data frame on which to evaluate the model. Defaults to training data.

Value

A tibble with evaluation results, containing at least ‘metric‘, ‘value‘, and possibly ‘time‘.

Examples

```
library(survival)
data(veteran)

mod <- survdnn(Surv(time, status) ~ age + karno + celltype,
               data = veteran, epochs = 5, verbose = FALSE)
evaluate_survdnn(mod, metrics = c("cindex", "ibs"), times = c(30, 90, 180))
evaluate_survdnn(mod, metrics = "brier", times = c(30, 90, 180))
```

`gridsearch_survdnn` *Grid Search for survdnn Hyperparameters*

Description

Performs grid search over user-specified hyperparameters and evaluates performance on a validation set.

Usage

```
gridsearch_survdnn(
  formula,
  train,
  valid,
  times,
  metrics = c("cindex", "ibs"),
  param_grid,
  .seed = 42
)
```

Arguments

`formula`	A survival formula (e.g., 'Surv(time, status) ~ .')
`train`	Training dataset
`valid`	Validation dataset
`times`	Evaluation time points (numeric vector)
`metrics`	Evaluation metrics (character vector): any of "cindex", "ibs", "brier"
`param_grid`	A named list of hyperparameters ('hidden', 'lr', 'activation', 'epochs', 'loss')
`.seed`	Optional random seed for reproducibility

Value

A tibble with configurations and their validation metrics

Examples

```
library(survdnn)
library(survival)
set.seed(123)

# Simulate small dataset
n <- 300
x1 <- rnorm(n); x2 <- rbinom(n, 1, 0.5)
time <- rexp(n, rate = 0.1)
status <- rbinom(n, 1, 0.7)
df <- data.frame(time, status, x1, x2)
```

```
# Split into training and validation
idx <- sample(seq_len(n), 0.7 * n)
train <- df[idx, ]
valid <- df[-idx, ]

# Define formula and param grid
formula <- Surv(time, status) ~ x1 + x2
param_grid <- list(
  hidden     = list(c(16, 8), c(32, 16)),
  lr         = c(1e-3),
  activation = c("relu"),
  epochs     = c(100),
  loss       = c("cox", "coxtime")
)

# Run grid search
results <- gridsearch_survdnn(
  formula = formula,
  train   = train,
  valid   = valid,
  times   = c(10, 20, 30),
  metrics = c("cindex", "ibs"),
  param_grid = param_grid
)

# View summary
dplyr::group_by(results, hidden, lr, activation, epochs, loss, metric) |>
  dplyr::summarise(mean = mean(value, na.rm = TRUE), .groups = "drop")
```

`ibs_survmat` *Integrated Brier Score (IBS) from a Survival Probability Matrix*

Description

Computes the Integrated Brier Score (IBS) over a set of evaluation time points, using trapezoidal integration and IPCW adjustment for right-censoring.

Usage

```
ibs_survmat(object, sp_matrix, times)
```

Arguments

`object`	A ‘Surv‘ object with observed time and status.
`sp_matrix`	A data frame or matrix of predicted survival probabilities. Each column corresponds to a time point in ‘times‘.
`times`	A numeric vector of time points. Must match the columns of ‘sp_matrix‘.

Value

A single numeric value representing the integrated Brier score.

Examples

```
set.seed(123)
library(survival)
data(veteran, package = "survival")
idx <- sample(nrow(veteran), 0.7 * nrow(veteran))
train <- veteran[idx, ]; test <- veteran[-idx, ]
mod <- survdnn(Surv(time, status) ~
age + karno + celltype, data = train, epochs = 50, verbose = FALSE)
pred <- predict(mod, newdata = test, times = c(30, 90, 180), type = "survival")
y_test <- model.response(model.frame(mod$formula, test))
ibs_survmat(y_test, sp_matrix = pred, times = c(30, 90, 180))
```

---

plot.survdnn *Plot survdnn Survival Curves using ggplot2*

Description

Visualizes survival curves predicted by a fitted ‘survdnn‘ model. Curves can be grouped by a categorical variable in ‘newdata‘ and optionally display only the group-wise means or overlay them.

Usage

```
## S3 method for class 'survdnn'
plot(
  x,
  newdata = NULL,
  times = 1:365,
  group_by = NULL,
  plot_mean_only = FALSE,
  add_mean = TRUE,
  alpha = 0.3,
  mean_lwd = 1.3,
  mean_lty = 1,
  ...
)
```

Arguments

x	A fitted ‘survdnn‘ model object.
newdata	Optional data frame for prediction (defaults to training data).
times	A numeric vector of time points at which to compute survival probabilities.
group_by	Optional name of a column in ‘newdata‘ used to color and group curves.

plot_mean_only	Logical; if 'TRUE', plots only the mean survival curve per group.
add_mean	Logical; if 'TRUE', adds mean curves to the individual lines.
alpha	Alpha transparency for individual curves (ignored if 'plot_mean_only = TRUE').
mean_lwd	Line width for mean survival curves.
mean_lty	Line type for mean survival curves.
...	Reserved for future use.

Value

A 'ggplot' object.

Examples

```
library(survival)
data(veteran)
set.seed(42)

mod <- survdnn(Surv(time, status) ~ age + karno + celltype, data = veteran,
               hidden = c(16, 8), epochs = 100, verbose = FALSE)
plot(mod, group_by = "celltype", times = 1:300)
```

---

predict.survdnn *Predict from a survdnn Model*

Description

Generate predictions from a fitted 'survdnn' model for new data. Supports linear predictors, survival probabilities at specified time points, or cumulative risk estimates.

Usage

```
## S3 method for class 'survdnn'
predict(object, newdata, times = NULL, type = c("survival", "lp", "risk"), ...)
```

Arguments

object	An object of class '"survdnn"' returned by [survdnn()].
newdata	A data frame of new observations to predict on.
times	Numeric vector of time points at which to compute survival or risk probabilities. Required if 'type = "survival"' or 'type = "risk"'.
type	Character string specifying the type of prediction to return: **"lp"** Linear predictor (log-risk score; higher implies worse prognosis). **"survival"** Predicted survival probabilities at each value of 'times'. **"risk"** Cumulative risk (1 - survival) at a single time point.
...	Currently ignored (for future extensions).

Value

A numeric vector (if ‘type = "lp"‘ or ‘"risk"‘), or a data frame (if ‘type = "survival"‘) with one row per observation and one column per ‘times‘.

Examples

```
library(survival)
data(veteran, package = "survival")

# Fit survdnn with Cox loss
mod <- survdnn(Surv(time, status) ~ age + karno + celltype, data = veteran,
               loss = "cox", epochs = 50, verbose = FALSE)

# Linear predictor (log-risk)
predict(mod, newdata = veteran, type = "lp")[1:5]

# Survival probabilities at selected times
predict(mod, newdata = veteran, type = "survival", times = c(30, 90, 180))[1:5, ]

# Cumulative risk at 180 days
predict(mod, newdata = veteran, type = "risk", times = 180)[1:5]
```

---

print.survdnn *Print a survdnn Model*

Description

Pretty prints a fitted ‘survdnn‘ model. Displays the formula, network architecture, training configuration, and final training loss.

Usage

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

Arguments

x	An object of class ‘"survdnn"‘, returned by [survdnn()].
...	Ignored (for future compatibility).

Value

The model object, invisibly.

Examples

```
library(survival)
data(veteran, package = "survival")
mod <- survdnn(Surv(time, status) ~
age + karno + celltype, data = veteran, epochs = 20, verbose = FALSE)
print(mod)
```

`summarize_cv_survdnn` *Summarize Cross-Validation Results from survdnn*

Description

Computes mean, standard deviation, and confidence intervals for metrics from cross-validation.

Usage

```
summarize_cv_survdnn(cv_results, by_time = TRUE, conf_level = 0.95)
```

Arguments

`cv_results`	A tibble returned by [cv_survdnn()].
`by_time`	Logical. Whether to stratify results by 'time' (if present).
`conf_level`	Confidence level for the intervals (default: 0.95).

Value

A tibble summarizing mean, sd, and confidence bounds per metric (and per time if applicable).

Examples

```
library(survival)
data(veteran)
res <- cv_survdnn(
  Surv(time, status) ~ age + karno + celltype,
  data = veteran,
  times = c(30, 90, 180, 270),
  metrics = c("cindex", "ibs"),
  folds = 3,
  .seed = 42,
  hidden = c(16, 8),
  epochs = 5
)
summarize_cv_survdnn(res)
```

summarize_tune_survdnn *Summarize survdnn Tuning Results*

Description

Aggregates cross-validation results from ‘tune_survdnn(return = "all")‘ by configuration, metric, and optionally by time point.

Usage

```
summarize_tune_survdnn(tuning_results, by_time = TRUE)
```

Arguments

tuning_results	The full tibble returned by ‘tune_survdnn(..., return = "all")‘.
by_time	Logical; whether to group and summarize separately by time points.

Value

A summarized tibble with mean and standard deviation of performance metrics.

summary.survdnn *Summarize a Deep Survival Neural Network Model*

Description

Provides a structured summary of a fitted ‘survdnn‘ model, including the network architecture, training configuration, and data characteristics. The summary is printed automatically with a styled header and sectioned output using {cli} and base formatting. The object is returned invisibly.

Usage

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

Arguments

object	An object of class ‘"survdnn"‘ returned by the [survdnn()] function.
...	Currently ignored (for future compatibility).

Value

Invisibly returns an object of class ‘"summary.survdnn"‘.

Examples

```
set.seed(42)
sim_data <- data.frame(
  age = rnorm(100, 60, 10),
  sex = factor(sample(c("male", "female"), 100, TRUE)),
  trt = factor(sample(c("A", "B"), 100, TRUE)),
  time = rexp(100, 0.05),
  status = rbinom(100, 1, 0.7)
)
mod <- survdnn(Surv(time, status) ~ age + sex + trt, data = sim_data, epochs = 50, verbose = FALSE)
summary(mod)
```

---

survdnn *Fit a Deep Neural Network for Survival Analysis*

Description

Trains a deep neural network (DNN) to model right-censored survival data using one of the predefined loss functions: Cox, AFT, or Coxtime.

Usage

```
survdnn(
  formula,
  data,
  hidden = c(32L, 16L),
  activation = "relu",
  lr = 1e-04,
  epochs = 300L,
  loss = c("cox", "cox_l2", "aft", "coxtime"),
  verbose = TRUE
)
```

Arguments

formula	A survival formula of the form ‘Surv(time, status) ~ predictors‘.
data	A data frame containing the variables in the model.
hidden	Integer vector. Sizes of the hidden layers (default: c(32, 16)).
activation	Character string specifying the activation function to use in each layer. Supported options: ‘"relu"‘, ‘"leaky_relu"‘, ‘"tanh"‘, ‘"sigmoid"‘, ‘"gelu"‘, ‘"elu"‘, ‘"softplus"‘.
lr	Learning rate for the Adam optimizer (default: ‘1e-4‘).
epochs	Number of training epochs (default: 300).
loss	Character name of the loss function to use. One of ‘"cox"‘, ‘"cox_l2"‘, ‘"aft"‘, or ‘"coxtime"‘.
verbose	Logical; whether to print loss progress every 50 epochs (default: TRUE).

Value

An object of class ‘"survdnn"‘ containing:

model Trained ‘nn_module‘ object.

formula Original survival formula.

data Training data used for fitting.

xnames Predictor variable names.

x_center Column means of predictors.

x_scale Column standard deviations of predictors.

loss_history Vector of loss values per epoch.

final_loss Final training loss.

loss Loss function name used ("cox", "aft", etc.).

activation Activation function used.

hidden Hidden layer sizes.

lr Learning rate.

epochs Number of training epochs.

Examples

```
set.seed(123)
df <- data.frame(
  time = rexp(100, rate = 0.1),
  status = rbinom(100, 1, 0.7),
  x1 = rnorm(100),
  x2 = rbinom(100, 1, 0.5)
)
mod <- survdnn(Surv(time, status) ~ x1 + x2, data = df, epochs = 5

, loss = "cox", verbose = FALSE)
mod$final_loss
```

---

tune_survdnn *Tune Hyperparameters for a survdnn Model via Cross-Validation*

Description

Performs k-fold cross-validation over a user-defined hyperparameter grid and selects the best configuration according to the specified evaluation metric.

Usage

```
tune_survdnn(
  formula,
  data,
  times,
  metrics = "cindex",
  param_grid,
  folds = 3,
  .seed = 42,
  refit = FALSE,
  return = c("all", "summary", "best_model")
)
```

Arguments

formula	A survival formula, e.g., 'Surv(time, status) ~ x1 + x2'.
data	A data frame.
times	A numeric vector of evaluation time points.
metrics	A character vector of evaluation metrics: "cindex", "brier", or "ibs". Only the first metric is used for model selection.
param_grid	A named list defining hyperparameter combinations to evaluate. Required names: 'hidden', 'lr', 'activation', 'epochs', 'loss'.
folds	Number of cross-validation folds (default: 3).
.seed	Optional seed for reproducibility (default: 42).
refit	Logical. If TRUE, refits the best model on the full dataset.
return	One of "all", "summary", or "best_model": **"all"** Returns the full cross-validation result across all combinations. **"summary"** Returns averaged results per configuration. **"best_model"** Returns the refitted model or best hyperparameters.

Value

A tibble or model object depending on the 'return' value.

Index