

# Package ‘survxai’

August 28, 2020

**Title** Visualization of the Local and Global Survival Model  
Explanations

**Version** 0.2.2

**Description** Survival models may have very different structures. This package contains functions for creating a unified representation of a 'survival' models, which can be further processed by various survival explainers. Tools implemented in 'survxai' help to understand how input variables are used in the model and what impact do they have on the final model prediction. Currently, four explanation methods are implemented. We can divide them into two groups: local and global. Explanations of the methods can be found in Grudziak et al.(2018) <doi:10.21105/joss.00961>.

**License** GPL

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 7.1.1

**Imports** breakDown, ggplot2, pec, scales, survival, survminer

**Depends** prodlim

**Suggests** CFC, covr, DALEX, knitr, markdown, randomForestSRC, rmarkdown, rms, testthat, tibble

**VignetteBuilder** knitr

**URL** <https://mi2datalab.github.io/survxai/>

**BugReports** <https://github.com/MI2DataLab/survxai/issues>

**NeedsCompilation** no

**Author** Aleksandra Grudziak [aut, cre],  
Alicja Gosiewska [aut],  
Przemyslaw Biecek [aut, ths]

**Maintainer** Aleksandra Grudziak <aleksandra.grudziak@gmail.com>

**Repository** CRAN

**Date/Publication** 2020-08-28 11:30:03 UTC

## R topics documented:

ceteris_paribus . . . . .	2
explain . . . . .	3
model_performance . . . . .	5
pbCTest . . . . .	6
pbCTrain . . . . .	6
plot.surv_ceteris_paribus_explainer . . . . .	7
plot.surv_explainer . . . . .	8
plot.surv_model_performance_explainer . . . . .	9
plot.surv_prediction_breakdown_explainer . . . . .	10
plot.surv_variable_response_explainer . . . . .	11
prediction_breakdown . . . . .	12
print.surv_ceteris_paribus_explainer . . . . .	13
print.surv_explainer . . . . .	13
print.surv_model_performance_explainer . . . . .	14
print.surv_prediction_breakdown_explainer . . . . .	14
print.surv_variable_response_explainer . . . . .	15
theme_mi2 . . . . .	15
variable_response . . . . .	16

<b>Index</b>	<b>17</b>
--------------	-----------

---

ceteris_paribus	<i>Ceteris Paribus</i>
-----------------	------------------------

---

### Description

The `ceteris_paribus()` function computes the predictions for the neighbor of our chosen observation. The neighbour is defined as the observations with changed value of one of the variable.

### Usage

```
ceteris_paribus(
  explainer,
  observation,
  grid_points = 5,
  selected_variables = NULL
)
```

### Arguments

explainer	a model to be explained, preprocessed by the 'survxai::explain' function
observation	a new observation for which predictions need to be explained
grid_points	grid_points number of points used for response path
selected_variables	if specified, then only these variables will be explained

**Value**

An object of the class `surv_ceteris_paribus_explainer`. It's a data frame with calculated average responses.

**Examples**

```
library(survxai)
library(rms)
data("pbcTrain")
data("pbcTest")
predict_times <- function(model, data, times){
  prob <- rms::survest(model, data, times = times)$surv
  return(prob)
}
cph_model <- cph(Surv(years, status)~ sex + bili + stage,
  data = pbcTrain, surv = TRUE, x = TRUE, y=TRUE)
surve_cph <- explain(model = cph_model, data = pbcTest[,-c(1,5)],
  y = Surv(pbcTest$years, pbcTest$status),
  predict_function = predict_times)
cp_cph <- ceteris_paribus(surve_cph, pbcTest[1,-c(1,5)])
```

---

`explain`*Create Survival Model Explainer*

---

**Description**

Survival models may have very different structures. This function creates a unified representation of a survival model, which can be further processed by various survival explainers (see also [explain](#)).

Please NOTE, that the `model` is actually the only required argument. But some survival explainers may require additional arguments.

**Usage**

```
explain(
  model,
  data = NULL,
  y,
  times = NULL,
  predict_function = yhat,
  link = I,
  label = tail(class(model), 1),
  ...
)

## Default S3 method:
explain(
```

```

model,
data = NULL,
y,
times = NULL,
predict_function = yhat,
link = I,
label = tail(class(model), 1),
...
)

```

### Arguments

model	object - a survival model to be explained
data	data.frame, tibble or matrix - data that will be used by survival explainers. If not provided then will be extracted from the model
y	object of class 'surv', contains event status and times
times	optional argument, the vector of time points on which survival probability will be predicted
predict_function	function that takes three arguments: model, new data, vector with times, and returns numeric vector or matrix with predictions. If not passed, function <a href="#">predictSurvProb</a> is used.
link	function - a transformation/link function that shall be applied to raw model predictions
label	character - the name of the survival model. By default it's extracted from the 'class' attribute of the model.
...	other parameters

### Value

An object of the class 'surv\_explainer'.

It's a list with following fields:

- model the explained model
- data the dataset
- y event statuses and times
- times time points on which survival probability is predicted
- predict\_function function that may be used for model predictions, shall return a single numerical value for each time.
- link function - a transformation/link function that shall be applied to raw model predictions
- class class/classes of a model
- label label, by default it's the last value from the class vector, but may be set to any character.

## Examples

```
library(survxai)
library(rms)
library(randomForestSRC)
data(pbc, package = "randomForestSRC")
pbc <- pbc[complete.cases(pbc),]
predict_times <- function(model, data, times){
  prob <- rms::survest(model, data, times = times)$surv
  return(prob)
}
cph_model <- cph(Surv(days/365, status)~ sex + bili + stage, data=pbc, surv=TRUE, x = TRUE, y=TRUE)
surve_cph <- explain(model = cph_model, data = pbc[,-c(1,2)], y = Surv(pbc$days/365, pbc$status),
  predict_function = predict_times)
```

---

model_performance	<i>Model performance for survival models</i>
-------------------	--

---

## Description

Function model\_performance calculates the prediction error for chosen survival model.

## Usage

```
model_performance(explainer, type = "BS")
```

## Arguments

explainer	a model to be explained, preprocessed by the 'survxai::explain' function
type	character - type of the response to be calculated Currently following options are implemented: 'BS' for Expected Brier Score

## Details

For type = "BS" prediction error is the time dependent estimates of the population average Brier score. At a given time point t, the Brier score for a single observation is the squared difference between observed survival status and a model based prediction of surviving time t.

## References

Ulla B. Mogensen, Hemant Ishwaran, Thomas A. Gerds (2012). Evaluating Random Forests for Survival Analysis Using Prediction Error Curves. *Journal of Statistical Software*, 50(11), 1-23. URL <http://www.jstatsoft.org/v50/i11/>.

**Examples**

```

library(survxai)
library(rms)
data("pbcTrain")
data("pbcTest")
cph_model <- cph(Surv(years, status)~ sex + bili + stage,
  data=pbcTrain, surv=TRUE, x = TRUE, y=TRUE)
surve_cph <- explain(model = cph_model, data = pbcTest[,-c(1,5)],
  y = Surv(pbcTest$years, pbcTest$status))
mp_cph <- model_performance(surve_cph)

```

---

pbcTest	<i>pbcTest</i>
---------	----------------

---

**Description**

PBC test set Data set based on pbc from randomForestSRC package. The data consists of 138 randomly chosen observations The pbcTest contains only complete cases for each observation. It contains 5 variables: 'status', 'sex', 'bili', 'stage', and 'years'.

**Source**

randomForestSRC

**References**

Flemming T.R and Harrington D.P., (1991) Counting Processes and Survival Analysis. New York: Wiley.

**Examples**

```

data("pbcTest", package = "survxai")
head(pbcTest)

```

---

pbcTrain	<i>pbcTrain</i>
----------	-----------------

---

**Description**

PBC train set Data set based on pbc from randomForestSRC package. The data consists of 138 randomly chosen observations The pbcTrain contains only complete cases for each observation. It contains 5 variables: 'status', 'sex', 'bili', 'stage', and 'years'.

**Source**

randomForestSRC

**References**

Flemming T.R and Harrington D.P., (1991) Counting Processes and Survival Analysis. New York: Wiley.

**Examples**

```
data("pbcTrain", package = "survxai")
head(pbcTrain)
```

---

```
plot.surv_ceteris_paribus_explainer
Plot for ceteris_paribus object
```

---

**Description**

Function plot for ceteris\_paribus object visualise estimated survival curve of mean probabilities in chosen time points. Black lines on each plot correspond to survival curve for our new observation specified in the ceteris\_paribus function.

**Usage**

```
## S3 method for class 'surv_ceteris_paribus_explainer'
plot(
  x,
  ...,
  selected_variable = NULL,
  scale_type = "factor",
  scale_col = NULL,
  ncol = 1
)
```

**Arguments**

x	object of class "surv_ceteris_paribus_explainer"
...	arguments to be passed to methods, such as graphical parameters for function <a href="#">geom_step</a> .
selected_variable	name of variable we want to draw ceteris paribus plot
scale_type	type of scale of colors, either "discrete" or "gradient"
scale_col	vector containing values of low and high ends of the gradient, when "gradient" type of scale was chosen
ncol	number of columns for faceting

**Examples**

```

library(survxai)
library(rms)
data("pbcTest")
data("pbcTrain")
predict_times <- function(model, data, times){
  prob <- rms::survest(model, data, times = times)$surv
  return(prob)
}
cph_model <- cph(Surv(years, status)~sex + bili + stage, data=pbcTrain, surv=TRUE, x = TRUE, y=TRUE)
surve_cph <- explain(model = cph_model, data = pbcTest[,-c(1,5)],
  y = Surv(pbcTest$years, pbcTest$status), predict_function = predict_times)
cp_cph <- ceteris_paribus(surve_cph, pbcTest[1,-c(1,5)])
plot(cp_cph)

```

---

plot.surv\_explainer     *Plot for surv\_explainer object*

---

**Description**

Function plot for surv\_explainer object visualise estimated survival curve of mean probabilities in chosen time points.

**Usage**

```

## S3 method for class 'surv_explainer'
plot(x, ...)

```

**Arguments**

x                    object of class "surv\_explainer"  
...                   other arguments for function [ggsurvplot](#)

**Examples**

```

library(survxai)
library(rms)
data("pbcTest")
data("pbcTrain")
predict_times <- function(model, data, times){
  prob <- rms::survest(model, data, times = times)$surv
  return(prob)
}
cph_model <- cph(Surv(years, status)~sex + bili + stage, data=pbcTrain, surv=TRUE, x = TRUE, y=TRUE)
surve_cph <- explain(model = cph_model, data = pbcTest[,-c(1,5)],
  y = Surv(pbcTest$years, pbcTest$status), predict_function = predict_times)

```



```
plot(surve_cph)
```

---

```
plot.surv_model_performance_explainer
```

*Plot for surv\_model\_performance object*

---

## Description

Function plot for surv\_model\_performance object.

## Usage

```
## S3 method for class 'surv_model_performance_explainer'  
plot(x, ...)
```

## Arguments

x	object of class "surv_model_performance"
...	optional, additional objects of class "surv_model_performance_explainer"

## Examples

```
library(survxai)  
library(rms)  
data("pbcTest")  
data("pbcTrain")  
predict_times <- function(model, data, times){  
  prob <- rms::survest(model, data, times = times)$surv  
  return(prob)  
}  
cph_model <- cph(Surv(years, status)~sex + bili + stage, data=pbcTrain, surv=TRUE, x = TRUE, y=TRUE)  
surve_cph <- explain(model = cph_model, data = pbcTest[,-c(1,5)],  
  y = Surv(pbcTest$years, pbcTest$status), predict_function = predict_times)  
mp_cph <- model_performance(surve_cph)  
plot(mp_cph)
```

---

```
plot.surv_prediction_breakdown_explainer
      Plot for surv_breakdown object
```

---

## Description

Function plot for surv\_breakdown object visualise estimated survival curve of mean probabilities in chosen time points.

## Usage

```
## S3 method for class 'surv_prediction_breakdown_explainer'
plot(
  x,
  ...,
  numerate = TRUE,
  lines = TRUE,
  lines_type = 1,
  lines_col = "black",
  scale_col = c("#010059", "#e0f6fb")
)
```

## Arguments

x	an object of class "surv_prediction_breakdown_explainer"
...	optional, additional objects of class "surv_prediction_breakdown_explainer"
numerate	logical; indicating whether we want to number curves
lines	logical; indicating whether we want to add lines on chosen time point or probability
lines_type	a type of line; see <a href="http://sape.inf.usi.ch/quick-reference/ggplot2/linetype">http://sape.inf.usi.ch/quick-reference/ggplot2/linetype</a>
lines_col	a color of line
scale_col	a vector containig two colors for gradient scale in legend

## Examples

```
library(survxai)
library(rms)
data("pbcTest")
data("pbcTrain")
predict_times <- function(model, data, times){
  prob <- rms::survest(model, data, times = times)$surv
  return(prob)
}
cph_model <- cph(Surv(years, status)~sex + bili + stage, data=pbcTrain, surv=TRUE, x = TRUE, y=TRUE)
surve_cph <- explain(model = cph_model, data = pbcTest[, -c(1,5)],
```

```

y = Surv(pbcTest$years, pbcTest$status), predict_function = predict_times)
broken_prediction <- prediction_breakdown(surve_cph, pbcTest[, -c(1,5)])
plot(broken_prediction)

```

---

```
plot.surv_variable_response_explainer
```

*Plot for surv\_variable\_response object*

---

## Description

Function plot for surv\_variable\_response object shows the expected output condition on a selected variable.

## Usage

```

## S3 method for class 'surv_variable_response_explainer'
plot(x, ..., split = "model")

```

## Arguments

x	an object of class "surv_variable_response"
...	optional, additional objects of class "surv_variable_response_explainer"
split	a character, either "model" or "variable"; sets the variable for faceting

## Examples

```

library(survxai)
library(rms)
data("pbcTest")
data("pbcTrain")
predict_times <- function(model, data, times){
  prob <- rms::survest(model, data, times = times)$surv
  return(prob)
}
cph_model <- cph(Surv(years, status)~sex + bili + stage, data=pbcTrain, surv=TRUE, x = TRUE, y=TRUE)
surve_cph <- explain(model = cph_model, data = pbcTest[, -c(1,5)],
  y = Surv(pbcTest$years, pbcTest$status), predict_function = predict_times)
svr_cph <- variable_response(surve_cph, "sex")
plot(svr_cph)

```

---

prediction\_breakdown *BreakDown for survival models*

---

## Description

Function `prediction_breakdown` is an extension of a `broken` function from `breakDown` package. It computes the contribution in prediction for the variables in the model. The contribution is defined as the difference between survival probabilities for model with added specific value of variable and with the random levels of this variable.

## Usage

```
prediction_breakdown(explainer, observation, time = NULL, prob = NULL, ...)
```

## Arguments

<code>explainer</code>	an object of the class 'surv_explainer'
<code>observation</code>	a new observation to explain
<code>time</code>	a time point at which variable contributions are computed. If <code>NULL</code> median time is taken.
<code>prob</code>	a survival probability at which variable contributions are computed
<code>...</code>	other parameters corresponding to arguments from <code>broken</code> function from <code>breakDown</code> package. See <a href="https://github.com/pbiecek/breakDown/blob/master/R/break_agnostic.R">https://github.com/pbiecek/breakDown/blob/master/R/break_agnostic.R</a> for details

## Value

An object of class `surv_prediction_breakdown_explainer`

## Examples

```
library(survxai)
library(rms)
data("pbcTest")
data("pbcTrain")
predict_times <- function(model, data, times){
  prob <- rms::survest(model, data, times = times)$surv
  return(prob)
}
cph_model <- cph(Surv(years, status)~sex + bili + stage, data=pbcTrain, surv=TRUE, x = TRUE, y=TRUE)
surve_cph <- explain(model = cph_model, data = pbcTest[,-c(1,5)],
  y = Surv(pbcTest$years, pbcTest$status), predict_function = predict_times)
broken_prediction <- prediction_breakdown(surve_cph, pbcTest[1,-c(1,5)])
```

---

`print.surv_ceteris_paribus_explainer`  
*Ceteris Paribus Print*

---

### **Description**

Ceteris Paribus Print

### **Usage**

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

### **Arguments**

x                    the model of 'surv\_ceteris\_paribus\_explainer' class  
...                   further arguments passed to or from other methods

### **Value**

a data frame

---

`print.surv_explainer`    *Print Survival Explainer Summary*

---

### **Description**

Print Survival Explainer Summary

### **Usage**

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

### **Arguments**

x                    a model survival explainer created with the 'explain()' function  
...                   further arguments passed to or from other methods

---

```
print.surv_model_performance_explainer
      Print Survival Model Performance
```

---

**Description**

Print Survival Model Performance

**Usage**

```
## S3 method for class 'surv_model_performance_explainer'
print(x, times = NULL, ...)
```

**Arguments**

x	a model to be explained, object of the class 'model_performance_explainer'
times	a vector of integer times on which we want to check the value of prediction error
...	further arguments passed to or from other methods

---

```
print.surv_prediction_breakdown_explainer
      Prediction Breakdown Print
```

---

**Description**

Prediction Breakdown Print

**Usage**

```
## S3 method for class 'surv_prediction_breakdown_explainer'
print(x, ..., digits = 3, rounding_function = round)
```

**Arguments**

x	the model model of 'surv_prediction_breakdown_explainer' class
...	further arguments passed to or from other methods
digits	number of decimal places (round) or significant digits (signif) to be used See the rounding_function argument
rounding_function	function that is to used for rounding numbers. It may be signif() which keeps a specified number of significant digits. Or the default round() to have the same precision for all components

---

```
print.surv_variable_response_explainer  
      Variable Response Print
```

---

### **Description**

Variable Response Print

### **Usage**

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

### **Arguments**

x                    the model of 'surv\_variable\_response\_explainer' class  
...                   further arguments passed to or from other methods

### **Value**

a data frame

---

```
theme_mi2                    MI^2 plot theme
```

---

### **Description**

ggplot theme for charts generated with MI<sup>2</sup> Data Lab packages.

### **Usage**

```
theme_mi2()
```

### **Value**

theme object that can be added to ggplot2 plots

---

variable_response	<i>Variable response for survival models</i>
-------------------	--

---

### Description

Function `variable_response` calculates the expected output condition on a selected variable.

### Usage

```
variable_response(explainer, variable, type = "pdp", link = explainer$link)
```

### Arguments

<code>explainer</code>	an object of the class 'surv_explainer'.
<code>variable</code>	a character with variable name.
<code>type</code>	a character - type of the response to be calculated. Currently following options are implemented: 'pdp' for Partial Dependency.
<code>link</code>	a function - a link function that shall be applied to raw model predictions. This will be inherited from the explainer.

### Examples

```
library(survxai)
library(rms)
data("pbcTest")
data("pbcTrain")
predict_times <- function(model, data, times){
  prob <- rms::survest(model, data, times = times)$surv
  return(prob)
}
cph_model <- cph(Surv(years, status)~sex + bili + stage, data=pbcTrain, surv=TRUE, x = TRUE, y=TRUE)
surve_cph <- explain(model = cph_model, data = pbcTest[,-c(1,5)],
  y = Surv(pbcTest$years, pbcTest$status), predict_function = predict_times)
svr_cph <- variable_response(surve_cph, "sex")
```



# Index

broken, [12](#)

ceteris\_paribus, [2](#)

explain, [3](#), [3](#)

geom\_step, [7](#)

ggsurvplot, [8](#)

model\_performance, [5](#)

pbCTest, [6](#)

pbCTrain, [6](#)

plot.surv\_ceteris\_paribus\_explainer, [7](#)

plot.surv\_explainer, [8](#)

plot.surv\_model\_performance\_explainer,  
[9](#)

plot.surv\_prediction\_breakdown\_explainer,  
[10](#)

plot.surv\_variable\_response\_explainer,  
[11](#)

prediction\_breakdown, [12](#)

predictSurvProb, [4](#)

print.surv\_ceteris\_paribus\_explainer,  
[13](#)

print.surv\_explainer, [13](#)

print.surv\_model\_performance\_explainer,  
[14](#)

print.surv\_prediction\_breakdown\_explainer,  
[14](#)

print.surv\_variable\_response\_explainer,  
[15](#)

theme\_mi2, [15](#)

variable\_response, [16](#)