Package 'SensIAT'

January 20, 2025

Title Sensitivity Analysis for Irregular Assessment Times

Version 0.1.1

Description Sensitivity analysis for trials with irregular and informative assessment times, based on a new influence function-based, augmented inverse intensity-weighted estimator.

Language en-US

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.2

Imports assertthat, dplyr, glue, KernSmooth, MASS, methods, orthogonalsplinebasis, pracma, purrr, Rcpp (>= 1.0.12), rlang, splines, stats, survival, tibble, tidyr, utils

Suggests dfoptim, inline, progress, spelling, testthat (>= 3.0.0)

Config/testthat/edition 3

Depends R (>= 4.4.0)

LazyData true

LinkingTo Rcpp

SystemRequirements C++17

URL `https://github.com/UofUEpiBio/SensIAT`

BugReports `https://github.com/UofUEpiBio/SensIAT/issues`

NeedsCompilation yes

Author Andrew Redd [aut, cre] (<`https://orcid.org/0000-0002-6149-2438`>),
Yujing Gao [aut],
Shu Yang [aut],
Bonnie Smith [aut],
Agatha Mallett [ctb, ctr],
Daniel Scharfstein [pdr, aut] (<`https://orcid.org/0000-0001-7482-9653`>),
University of Utah [cph]

Maintainer Andrew Redd <`andrew.redd@hsc.utah.edu`>

Repository CRAN

Date/Publication 2024-11-17 16:30:02 UTC

Contents

`fit_SensIAT_fulldata_model` *Produce fitted model for group (treatment or control)*

Description

Produces a fitted model that may be used to produce estimates of mean and variance for the given group.

Usage

```
fit_SensIAT_fulldata_model(data, trt, ...)

fit_SensIAT_within_group_model(
  group.data,
  outcome_modeler,
  knots,
  id.var,
  outcome.var,
  time.var,
  alpha = 0,
  intensity.covariates = ~.,
  outcome.covariates = ~. - 1,
  End = max({
     {
         time.var
     }
 }, na.rm = TRUE) + 1,
  integration.tolerance = .Machine$double.eps^(1/3),
  intensity.bandwidth = NULL,
  ...,
  influence.args = list()
)
```

Arguments

`data`	the full data set.
`trt`	an expression that determine what is treated as the treatment. Everything not treatment is considered control.
`...`	add parameters as needed or use this to pass forward into the outcome_modeler.
`group.data`	The data for the group that is being analyzed. Preferably passed in as a single `tibble` that internally is subsetted/filtered as needed.
`outcome_modeler`	A separate function that may be swapped out to switch between negative-binomial, single index model, or another we will dream up in the future.
`knots`	knot locations for defining the spline basis.
`id.var`	The variable that identifies the patient.
`outcome.var`	The variable that contains the outcome.
`time.var`	The variable that contains the time.
`alpha`	The sensitivity parameter.
`intensity.covariates`	A formula representing modifications to the intensity model.
`outcome.covariates`	A formula representing modifications to the outcome model. The default removes the intercept term.
`End`	The end time for this data analysis, we need to set the default value as the max value of the time
`integration.tolerance`	The tolerance for the integration.
`intensity.bandwidth`	The bandwidth for the intensity model kernel.
`influence.args`	A list of additional arguments to pass to the influence function.

Details

This function should be agnostic to whether it is being provided a treatment or control group.

Value

a list with class `SensIAT-fulldata-fitted-model` with two components, `control` and `treatment`, each of which is an independently fitted `SensIAT-within-group-fitted-model` fit with the fit_within_group_model function.

Should return everything needed to define the fit of the model. This can then be used for producing the estimates of mean, variance, and in turn treatment effect. For the full data model a list with two models one each for the treatment and control groups.

Functions

- `fit_SensIAT_fulldata_model()`: Fit the sensitivity analysis for both treatment and control groups.

Examples

```
model <-
    fit_SensIAT_within_group_model(
        group.data = SensIAT_example_data,
        outcome_modeler = SensIAT_sim_outcome_modeler,
        alpha = c(-0.6, -0.3, 0, 0.3, 0.6),
        id.var = Subject_ID,
        outcome.var = Outcome,
        time.var = Time,
        End = 830,
        knots = c(60,60,60,60,260,460,460,460,460),
    )
```

`pcoriaccel_estimate_pmf` *Directly estimate the probability mass function of Y.*

Description

Directly estimate the probability mass function of Y.

Usage

```
pcoriaccel_estimate_pmf(Xb, Y, xi, y_seq, h, kernel = "K2_Biweight")
```

Arguments

`Xb`	Numeric vector of individual linear predictors from the data
`Y`	Numeric vector of individual responses from the data
`xi`	value of the individuals linear predictor at the point of estimation
`y_seq`	Numeric vector of unique values of `Y`.
`h`	bandwidth of the kernel
`kernel`	character string specifying the kernel to use, either `"dnorm"`, `"K2_Biweight"`, or `"K4_Biweight"`

`pcoriaccel_evaluate_basis` *Compiled version of* `evaluate_basis()` *function*

Description

Compiled version of `evaluate_basis()` function

Usage

```
pcoriaccel_evaluate_basis(spline_basis, x)
```

Arguments

`spline_basis`	The spline basis, S4 class `orthogonalsplinebasis::SplineBasis`
`x`	The point to evaluate

Value

Vector of the basis functions evaluated at x.

`pcori_conditional_means` *Compute Conditional Means*

Description

Compute Conditional Means

Usage

```
pcori_conditional_means(model, alpha = 0, new.data = model.frame(model), ...)
```

Arguments

`model`	An object of class `SensIAT::outcome-model`
`alpha`	Sensitivity parameter
`new.data`	Data to compute conditional means for, defaults to the model frame for the fitted model.
`...`	passed onto methods.

Details

Compute the conditional expectations needed for predictions in the models. Three additional values/expectations are computed:

- `$E \big[ Y(t) \exp \{  \alpha Y(t) \}   | A(t)=1, \bar{O}(t) \big]$`, returned as `E_y_past`, and
- `$E \big[ \exp \{ \alpha Y(t) \} \  | A(t)=1, \bar{O}(t) \big]$`, returned as `E_exp_alphaY`.

Value

The `new.data` frame with additional columns `E_Y_past`, and `E_exp_alphaY` appended.

`predict.SensIAT_fulldata_model`

Predict mean and variance of the outcome for a `SensIAT` *within-group model*

Description

Predict mean and variance of the outcome for a `SensIAT` within-group model

Usage

```
## S3 method for class 'SensIAT_fulldata_model'
predict(object, time, ...)

## S3 method for class 'SensIAT_within_group_model'
predict(object, time, include.var = TRUE, ..., base = object$base)
```

Arguments

`object`	SensIAT_within_group_model object
`time`	Time points of interest
`...`	Currently ignored.
`include.var`	Logical. If TRUE, the variance of the outcome is also returned
`base`	A `SplineBasis` object used to evaluate the basis functions.

Value

If include.var is TRUE, a `tibble` with columns time, mean, and var is returned. otherwise if include.var is FALSE, only the mean vector is returned.

Functions

- `predict(SensIAT_fulldata_model)`: For each combination of `time` and `alpha` estimate the mean response and variance for each group as well as estimate the mean treatment effect and variance.

Examples

```
model <-
    fit_SensIAT_within_group_model(
        group.data = SensIAT_example_data,
        outcome_modeler = SensIAT_sim_outcome_modeler,
        alpha = c(-0.6, -0.3, 0, 0.3, 0.6),
        id.var = Subject_ID,
        outcome.var = Outcome,
        time.var = Time,
        End = 830,
        knots = c(60,60,60,60,260,460,460,460,460),
    )
predict(model, time = c(90, 180))
```

`SensIAT_example_data` *SensIAT Example Data*

Description

A simulated dataset for use in the SensIAT tutorial, testing and documentation.

Usage

```
SensIAT_example_data
```

Format

A data frame with 779 rows and 4 variables consisting of 200 simulated patients. Each row in the data represents a visit for the patient. The columns are:

Subject_ID A unique identifier for each patient.

Visit The ordinal number of the visit for the patient. Baseline observation is 0.

Time The time of the visit in days, since baseline.

Outcome The outcome of interest.

`SensIAT_jackknife` *Estimate response with jackknife resampling*

Description

Estimate response with jackknife resampling

Usage

```
SensIAT_jackknife(original.object, time, ...)
```

Arguments

`original.object`	A SensIAT_within_group_model object.
`time`	Time points for which to estimate the response.
`...`	currently ignored.

Value

A `tibble` with columns alpha, time, jackknife_mean, and jackknife_var, where jackknife_mean is the mean of the jackknife estimates and jackknife_var is the estimated variances of the response at the given time points for the specified alpha values.

Examples

```
## Not run:
original.object <-
fit_SensIAT_within_group_model(
    group.data = SensIAT_example_data,
    outcome_modeler = SensIAT_sim_outcome_modeler,
    alpha = c(-0.6, -0.3, 0, 0.3, 0.6),
    id.var = Subject_ID,
    outcome.var = Outcome,
    time.var = Time,
    intensity.bandwidth = 30,
    knots = c(60,60,60,60,260,460,460,460,460),
    End = 830
)
jackknife.estimates <- SensIAT_jackknife(original.object, time = c(90, 180, 270, 360, 450))

## End(Not run)
```

`SensIAT_sim_outcome_modeler` *Outcome Modeler for* `SensIAT` *Single Index Model.*

Description

Outcome Modeler for `SensIAT` Single Index Model.

Usage

```
SensIAT_sim_outcome_modeler(
  formula,
  data,
  kernel = "K2_Biweight",
  method = "nmk",
  id = ..id..,
  ...
)
```

Arguments

`formula`	The outcome model formula
`data`	The data to fit the outcome model to. Should only include follow-up data, i.e. time > 0.
`kernel`	The kernel to use for the outcome model.
`method`	The optimization method to use for the outcome model, either `"optim"`, `"nlminb"`, or `"nmk"`.
`id`	The patient identifier variable for the data.
`...`	Currently ignored, included for future compatibility.

Value

Object of class `SensIAT::Single-index-outcome-model` which contains the outcome model portion.

Examples

```
model <-
    fit_SensIAT_within_group_model(
        group.data = SensIAT_example_data,
        outcome_modeler = SensIAT_sim_outcome_modeler,
        alpha = c(-0.6, -0.3, 0, 0.3, 0.6),
        id.var = Subject_ID,
        outcome.var = Outcome,
        time.var = Time,
        End = 830,
        knots = c(60,60,60,60,260,460,460,460,460),
    )
```

Index