Package ‘TBFmultinomial’

October 12, 2022

Type Package
Title TBF Methodology Extension for Multinomial Outcomes
Version 0.1.3
VignetteBuilder knitr
Description Extends the test-based Bayes factor (TBF) methodology to multinomial regression models and discrete time-to-event models with competing risks. The TBF methodology has been well developed and implemented for the generalised linear model [Held et al. (2015) <doi:10.1214/14-STS510>] and for the Cox model [Held et al. (2016) <doi:10.1002/sim.7089)].
Depends VGAM, nnet, parallel, stats, stringr, plotrix, methods
Suggests knitr, splines
License GPL (>= 2)
Encoding UTF-8
LazyData true
RoxygenNote 6.0.1
Author Rachel Heyard [aut, cre]
Maintainer Rachel Heyard <rachel.heyard@uzh.ch>
NeedsCompilation no
Repository CRAN
Date/Publication 2018-10-12 13:30:06 UTC

R topics documented:

  TBFmultinomial-package .......................................................... 2
  AIC_BIC_based_marginalLikelihood ......................................... 2
  all_formulas ............................................................................ 3
  as.data.frame.PMP ................................................................. 4
  CSVS ...................................................................................... 5
  model_priors ........................................................................... 6
  PIPs_by_landmarking ............................................................... 7
  plot_CSVS ............................................................................... 8
TBFmultinomial-package

Objective Bayesian variable selection for multinomial regression and
discrete time-to-event models with competing risks

Description

Extension of the TBF methodology introduced by Held et al. (2015) <doi:10.1214/14-STS510> to
discrete time-to-event models with competing risks (also applicable to the multinomial regression
model)

Author(s)

Rachel Heyard <rachel.heyard@uzh.ch>

AIC_BIC_based_marginalLikelihood

Marginal likelihoods based on AIC or BIC

Description

This function computes the marginal likelihoods based on the AIC or on the BIC, that will later be
used to calculate the TBF.

Usage

AIC_BIC_based_marginalLikelihood(fullModel = NULL, candidateModels = NULL,
data, discreteSurv = TRUE, AIC = TRUE, package = "nnet", maxit = 150,
numberOfCores = 1)
Arguments

fullModel    formula of the model including all potential variables
candidateModels    Instead of defining the full model we can also specify the candidate models
                    whose deviance statistic and d.o.f should be computed
data
discreteSurv    Boolean variable telling us whether a ‘simple’ multinomial regression is looked
                for or if the goal is a discrete survival-time model for multiple modes of failure
                is needed.
AIC    if TRUE, AIC will be used, else we use BIC
package    Which package should be used to fit the models; by default the nnet package is
            used; we could also specify to use the package 'VGAM'
maxit
numberCores    How many cores should be used in parallel?

Value

a vector with the marginal likelihoods of all candidate models

Author(s)

Rachel Heyard

Examples

# data extraction:
data("VAP_data")

# the definition of the full model with three potential predictors:
FULL <- outcome ~ ns(day, df = 4) + gender + type + SOFA
# here the define time as a spline with 3 knots

# now we can compute the marginal likelihoods based on the AIC f.ex:
mL_AIC <-
AIC_BIC_based_marginalLikelihood(fullModel = FULL,
data = VAP_data,
discreteSurv = TRUE,
AIC = TRUE)

all_formulas    Formulas of all the candidate models

Description

This function retrieves the formulas of all the candidate models if the reference model is the null / baseline model.
Usage

all_formulas(fullModel, discreteSurv = TRUE)

Arguments

fullModel  formula of the model including all potential variables
discreteSurv  Boolean variable telling us whether a ‘simple’ multinomial regression is looked for or if the goal is a discrete survival-time model for multiple modes of failure is needed.

Value

character vector with all the formulas; the first one will be the reference model; the last element will be the full model.

Author(s)

Rachel Heyard

Examples

data("VAP_data")
FULL <- outcome ~ ns(day, df = 4) + male + type + SOFA
models <- TBFmultinomial:::all_formulas(fullModel = FULL,
discreteSurv = TRUE)
# models

as.data.frame.PMP

Convert a PMP object into a data frame

Description

This function takes a PMP object an returns a data.frame summarising the information.

Usage

## S3 method for class 'PMP'
as.data.frame(x, ...)

Arguments

x  valid PMP object
...

Value

a data.frame with the posterior and prior probabilities as well as the definition of the models
CSVS

Cause-specific variable selection (CSVS)

Description

This function performs CSVS given a model fitted using the multinom() function of the nnet package or the vglm() function of the VGAM package.

Usage

CSVS(g, model, discreteSurv = TRUE, nbIntercepts = NULL, package = "nnet")

Arguments

g the estimated g, must be fixed to one value
model the model fitted using either nnet or VGAM
discreteSurv Boolean variable telling us whether a 'simple' multinomial regression is looked for or if the goal is a discrete survival-time model for multiple modes of failure is needed.
nbIntercepts how many cause-specific intercepts are there? they
package Which package has been used to fit the model, nnet or VGAM?

Author(s)

Rachel Heyard

Examples

# data extraction:
data("VAP_data")

# the definition of the full model with three potential predictors:
FULL <- outcome ~ ns(day, df = 4) + gender + type + SOFA
# here the define time as a spline with 3 knots

# we first need to fit the multinomial model:
model_full <- multinom(formula = FULL, data = VAP_data,
maxit = 150, trace = FALSE)

G <- 9 # let's suppose g equals to nine

# then we proceed to CSVS
CSVS_nnet <- CSVS(g = G, model = model_full,
    discreteSurv = TRUE, package = "nnet")
model_priors

Description

This function computes the prior model probabilities of the candidate models

Usage

`model_priors(fullModel, discreteSurv = TRUE, modelPrior = "flat")`

Arguments

- `fullModel`: formula of the model including all potential variables
- `discreteSurv`: Boolean var telling us whether a 'simple' multinomial regression is looked for or if the goal is a discrete survival-time model for multiple modes of failure is needed.
- `modelPrior`: what prior should be used on the model space? `modelPrior` should be included in `{'flat','dependent'}` where 'flat' means a uniform prior and 'dependent' sets a multiplicity-corrected model prior on the model space.

Value

a numerical vector with the prior model probabilities

Author(s)

Rachel Heyard

Examples

# the definition of the full model with three potential predictors:
FULL <- outcome ~ ns(day, df = 4) + gender + type + SOFA
# here we define time as a spline with 3 knots
priors <- model_priors(fullModel = FULL, discreteSurv = TRUE,  
modelPrior = 'dependent')
Description

This function gives us the PIPs for each landmark.

Usage

PIPs_by_landmarking(fullModel, data, discreteSurv = TRUE, numberCores = 1, package = "nnet", maxit = 150, prior = "flat", method = "LEB", landmarkLength = 1, lastlandmark, timeVariableName)

Arguments

fullModel formula of the model including all potential variables
data the data frame with all the information
discreteSurv Boolean variable telling us whether a 'simple' multinomial regression is looked for or if the goal is a discrete survival-time model for multiple modes of failure is needed.
numberCores How many cores should be used in parallel?
package Which package should be used to fit the models; by default the nnet package is used; we could also specify to use the package 'VGAM'
maxit Only needs to be specified with package nnet: maximal number of iterations
prior Prior on the model space
method Method for the g definition
landmarkLength Length of the landmark, by default we use each day
lastlandmark Where will be the last landmark?
timeVariableName What is the name of the variable indicating time?

Value

a list with the PIPs for each landmark

Author(s)

Rachel Heyard
Examples

```r
# extract the data:
data("VAP_data")

# the definition of the full model with three potential predictors:
FULL <- outcome ~ ns(day, df = 4) + gender + type + SOFA
# here we define time as a spline with 3 knots

PIPs_landmark <- PIPs_by_landmarking(fullModel = FULL, data = VAP_data,
discreteSurv = TRUE, numberCores = 1,
package = "nnet", maxit = 150,
prior = "flat", method = "LEB",
landmarkLength = 7, lastlandmark = 21,
timeVariableName = "day")
```

---

**plot_CSVS**  
*Plot a CSVS object*

Description

Plot a CSVS object

Usage

```r
plot_CSVS(CSVSobject, namesVar = NULL, shrunken = FALSE,
standardized = FALSE, numberIntercepts, ...)
```

Arguments

- **CSVSobject**: valid CSVS object
- **namesVar**: names of the variables
- **shrunken**: should the coefficients be shrunken?
- **standardized**: should the coefficients be standardized?
- **numberIntercepts**: how many cause-specific intercepts are in the model for each outcome
- **...**: parameters for plot

Author(s)

Rachel Heyard
**Posterior model probability**

**Description**

This function computes the posterior probability of all candidate models.

**Usage**

```r
PMP(fullModel = NULL, candidateModels = NULL, data = NULL,
   discreteSurv = TRUE, modelPrior = NULL, method = "LEB",
   prior = "flat", package = "nnet", maxit = 150, numberCores = 1)
```

**Arguments**

- `fullModel` formula of the model including all potential variables
- `candidateModels` Instead of defining the full model we can also specify the candidate models whose deviance statistic and d.o.f should be computed
- `data` the data frame with all the information
- `discreteSurv` Boolean variable telling us whether a 'simple' multinomial regression is looked for or if the goal is a discrete survival-time model for multiple modes of failure is needed.
- `modelPrior` optionally the model priors can be computed before if candidateModels is different from NULL.
- `method` tells us which method for the definition of g should be used. Possibilities are: LEB, GEB, g=n, hyperG, ZS, ZSadapted and hyperGN
- `prior` should a dependent or a flat prior be used on the model space? Only needed if method = 'GEB'.
- `package` Which package should be used to fit the models; by default the nnet package is used; we could also specify to use the package 'VGAM'
- `maxit` Only needs to be specified with package nnet: maximal number of iterations
- `numberCores` How many cores should be used in parallel?

**Value**

an object of class `TBF.ingredients`

**Author(s)**

Rachel Heyard
Examples

# extract the data:
data("VAP_data")

# the definition of the full model with three potential predictors:
FULL <- outcome ~ ns(day, df = 4) + gender + type + SOFA
# here we define time as a spline with 3 knots

# computation of the posterior model probabilities:
test <- PMP(fullModel = FULL, data = VAP_data,
            discreteSurv = TRUE, maxit = 150)
class(test)

---

**PMP-class**  
*Class for PMP objects*

**Description**

Class for PMP objects

---

**postInclusionProb**  
*Posterior inclusion probability (PIP)*

**Description**

This function computes the PIPs of all potential predictors

**Usage**

postInclusionProb(object)

**Arguments**

- **object**: An object of class PMP

**Value**

an named vector with all PIPs

**Author(s)**

Rachel Heyard
Examples

```r
# extract the data:
data("VAP_data")

# the definition of the full model with three potential predictors:
FULL <- outcome ~ ns(day, df = 4) + gender + type + SOFA
# here we define time as a spline with 3 knots

# computation of the posterior model probabilities:
test <- PMP(fullModel = FULL, data = VAP_data, 
            discreteSurv = TRUE, maxit = 150)
class(test)

# computation of the posterior inclusion probabilities:
postInclusionProb(test)
```

---

**sample_multinomial**  
*Samples from a PMP object*

**Description**

This function samples from a specific model inside a PMP object.

**Usage**

```r
sample_multinomial(PMP_object, shrink = TRUE, data, which = "MPM", 
                   discreteSurv = TRUE)
```

**Arguments**

- `PMP_object` formula of the model including all potential variables
- `shrink` should the coefficients be shrunken towards their prior mean?
- `data` the (training) data frame with all the information
- `which` which model should be sampled from? either an integer, 'MPM' or 'MAP'
- `discreteSurv` Boolean variable telling us whether a 'simple' multinomial regression is looked for or if the goal is a discrete survival-time model for multiple modes of failure is needed.

**Value**

returns an object with the model coefficients and supplementary information

**Author(s)**

Rachel Heyard
Description

This function computes the TBF as well as g

Usage

TBF(ingredients = NULL, fullModel = NULL, method = "LEB", data = NULL, discreteSurv = TRUE, prior = NULL, package = "nnet", maxit = 150)

Arguments

<table>
<thead>
<tr>
<th>Ingredient</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ingredients</td>
<td>TBF_ingredients_object ingredients for the TBF (and g) calculation.</td>
</tr>
<tr>
<td>fullModel</td>
<td>if ingredients is NULL, formula of the model including all potential variables</td>
</tr>
<tr>
<td>method</td>
<td>tells us which method for the definition of g should be used. Possibilities are: LEB, GEB, g=n, hyperG, ZS, ZSadapted and hyperGN</td>
</tr>
<tr>
<td>data</td>
<td>the data frame with all the information. Only needed if ingredients is NULL</td>
</tr>
<tr>
<td>discreteSurv</td>
<td>Boolean variable telling us whether a 'simple' multinomial regression is looked for or if the goal is a discrete survival-time model for multiple modes of failure is needed.</td>
</tr>
<tr>
<td>prior</td>
<td>should a dependent or a flat prior be used on the model space? Only needed if method = 'GEB'.</td>
</tr>
<tr>
<td>package</td>
<td>Which package should be used to fit the models; by default the nnet package is used; we could also specify to use the package 'VGAM'</td>
</tr>
<tr>
<td>maxit</td>
<td>Only needs to be specified with package nnet: maximal number of iterations</td>
</tr>
</tbody>
</table>

Value

A list with the TBF and the g (if it is fixed) for all the candidate models.

Author(s)

Rachel Heyard
**TBF_ingredients**  
*Ingredients to calculate the TBF*

---

### Description

This function calculates the ingredients needed to compute the TBFs: like the deviances with their degrees of freedom of the relevant candidate models.

### Usage

```r
TBF_ingredients(fullModel = NULL, data, discreteSurv = FALSE,  
numberCores = 1, candidateModels = NULL, package = "nnet",  
maxit = 150)
```

### Arguments

- **fullModel**: formula of the model including all potential variables
- **data**: the data frame with all the information
- **discreteSurv**: Boolean variable telling us whether a 'simple' multinomial regression is looked for or if the goal is a discrete survival-time model for multiple modes of failure is needed.
- **numberCores**: How many cores should be used in parallel?
- **candidateModels**: Instead of defining the full model we can also specify the candidate models whose deviance statistic and d.o.f should be computed
- **package**: Which package should be used to fit the models; by default the *nnet* package is used; we could also specify to use the package 'VGAM'
- **maxit**: Only needs to be specified with package *nnet*: maximal number of iterations

### Value

an object of class *TBF.ingredients*

### Author(s)

Rachel Heyard
VAP_data

Data on VAP acquisition in one ICU

Description

It is a tiny subset of the OUTCOMEREA database whose only perhaps will be to test an illustrate the functions of this package.

Usage

data(VAP_data)

Format

A data frame with 1640 rows and 7 variables on 90 distinct patients:

- **ID**  distinct ID for each patient
- **day**  day of ventilation, day = 1 is the first day of ventilation
- **type** is it a medical or a surgical patient
- **gender**  gender of the patient, 1 = male, 0 = female
- **SAPSadmission**  the SAPS 2 score at admission to the ICU
- **SOFA**  the daily SOFA score
- **outcome**  final outcome after the first observation period
Index

* package
  TBFmultinomial-package, 2

AIC_BIC_based_marginalLikelihood, 2
all_formulas, 3
as.data.frame.PMP, 4

CSVs, 5, 8

model_priors, 6

PIPs_by_landmarking, 7
plot_CSVS, 8
PMP, 4, 9
PMP-class, 10
postInclusionProb, 10

sample_multinomial, 11

TBF, 12
TBF_ingredients, 13
TBFmultinomial
  (TBFmultinomial-package), 2
TBFmultinomial-package, 2

VAP_data, 14