Package ‘tmle.npvi’

May 22, 2015

Type  Package
Title  Targeted Learning of a NP Importance of a Continuous Exposure
Version  0.10.0
Date  2015-05-13
Author  Antoine Chambaz, Pierre Neuvial
Maintainer  Pierre Neuvial <pierre.neuvial@genopole.cnrs.fr>
Description  Targeted minimum loss estimation (TMLE) of a non-parametric variable importance measure of a continuous exposure 'X' on an outcome 'Y', taking baseline covariates 'W' into account.
License  GPL
LazyLoad  yes
LazyData  yes
Depends  R (>= 2.10), R.utils (>= 1.4.1)
Imports  R.methodsS3, R.oo, MASS, Matrix, geometry
Suggests  SuperLearner (>= 2.0), e1071 (>= 1.5.24), randomForest (>= 4.5-35), polspline (>= 1.1.4), gam (>= 1.03), knitr
VignetteBuilder  knitr
NeedsCompilation  no
Repository  CRAN
Date/Publication  2015-05-22 18:59:02

R topics documented:

as.character.NPVI .................................................. 2
extractW .......................................................... 3
extractXW .......................................................... 4
getHistory.NPVI .................................................. 4
getLightFit ......................................................... 6
getObs.NPVI ....................................................... 7
getPsi.NPVI ....................................................... 8
getPsiSd.NPVI ..................................................... 8
as.character.NPVI

Returns a Description

Description

Returns a short string describing the NPVI object.

Usage

```r
## S3 method for class 'NPVI'
as.character(x, ...)
```

Arguments

- `x` An object of class `TMLE.NPVI`.
- `...` Not used.
**Value**

A character string summarizing the content of the object. The summary contains:

- The sample size of the data set involved in the TMLE procedure.
- The value of the TMLE and its estimated standard error.
- A reminder of the tuning of the stopping criteria, and a report on the convergence of the TMLE procedure (see `tmle.npvi`).
- A confidence interval with default level of 95% (the level can be changed by using `setConfLevel`).
- The $p$-value of the two-sided test of “$\Psi(P_0) = 0$”.
- The $p$-value of the two-sided test of “$\Psi(P_0) = \Phi(P_0)$”, with the estimated value of $\Phi(P_0)$.

**Author(s)**

Antoine Chambaz, Pierre Neuvial

**See Also**

`tmle.npvi`

**Examples**

`FALSE`

---

**extractW**

Extracts $W$ Columns from Matrix of Observations

**Description**

Extracts the $W$ column(s) from a matrix of observations.

**Usage**

`extractW(mat)`

**Arguments**

- **mat**: A matrix of observations, as the `obs` argument of function `tmle.npvi`.

**Details**

Mainly for internal use.

**Value**

The matrix extracted from `mat` by removing the two $X$ and $Y$ columns.

**Author(s)**

Antoine Chambaz, Pierre Neuvial
**extractXW**

*Removes the Y Column from Matrix of Observations*

**Description**

Removes the Y column from a matrix of observations.

**Usage**

```r
extractXW(mat)
```

**Arguments**

- `mat` A matrix of observations, as the obs argument of function `tmle.npvi`.

**Details**

Mainly for internal use.

**Value**

The matrix extracted from `mat` by removing the Y column in such a way that the first column is X.

**Author(s)**

Antoine Chambaz, Pierre Neuvial

---

**getHistory.NPVI**

*Returns History of TMLE Procedure*

**Description**

Returns the 'history' of the TMLE procedure.

**Usage**

```r
## S3 method for class 'NPVI'
getHistory(this, ...)
```

**Arguments**

- `this` An object of class `TMLE.NPVI`.
- `...` Not used.
Value

Returns a numeric matrix which encapsulates a summary of the TMLE procedure. If $k$ successive updates were performed, then the matrix has either $k + 1$ rows (if cleverCovTheta was set to FALSE in the call to tmle.npvi) or $2k+1$ rows (otherwise). The matrix has 14 columns:

- "eps", values of the unique fluctuation parameter (if cleverCovTheta was set to FALSE in the call to tmle.npvi), or values of the parameter involved in the fluctuation of the joint distribution of $(X, W)$ during each update (otherwise).
- "llli", increases in likelihood yielded by each update (if cleverCovTheta was set to FALSE in the call to tmle.npvi), or increases in likelihood yielded by the fluctuation of the joint distribution of $(X, W)$ during each update (otherwise).
- "mic1", empirical means of the first component of the efficient influence curve at each step of the TMLE procedure.
- "epsT", values of the fluctuation parameter involved in the fluctuation of the conditional distribution of $Y$ given $(X, W)$ during each update (if cleverCovTheta was set to TRUE in the call to tmle.npvi), or NA (otherwise).
- "lllT", successive increases in likelihood yielded by the fluctuation of the conditional distribution of $Y$ given $(X, W)$ during each update (if cleverCovTheta was set to TRUE in the call to tmle.npvi), or NA (otherwise).
- "mic2", empirical means of the second component of the efficient influence curve at each step of the TMLE procedure.
- "psi", increasingly targeted estimators $\Psi(P^k_n)$ of the parameter of interest. The last one is the TMLE. Their computation involves simulation of iid copies of $(X, W)$ under $P^k_n$.
- "psi.sd", estimated standard deviations of the increasingly targeted estimators of the parameter of interest. The last one corresponds to the TMLE. The computation involves the same iid copies of $(X, W)$ as above.
- "psipn", same as "psi" except that the *observed* $(X_i, W_i)$ are used instead of simulated copies drawn from $P^k_n$. Of course, "psi" must be favored.
- "psipn.sd", same as "psi.sd" except that the *observed* $(X_i, W_i)$ are used instead of simulated copies drawn from $P^k_n$. Of course, "psi.sd" must be favored.
- "mic", empirical means of the efficient influence curve at each step of the TMLE procedure. This column is the sum of the "mic1" and "mic2" columns.
- "div", total variation distances between each pair of successive distributions constructed in the course of the TMLE procedure.
- "sic", estimated standard deviations of the efficient influence curve at each step of the TMLE procedure.
- "phi", non-parametric substitution estimator of $\phi = \Phi(P)$ where

$$
\Phi(P) = \frac{EP[f(X)Y]}{EP[f(X)^2]},
$$

with $P$ the distribution of the random vector $(W, X, Y)$. The alternative parameter $\phi$ should be interpreted as the counterpart of $\psi$ which neglects $W$.
- "sicAlt", estimated standard deviations of the efficient influence curve of $\Psi - \Phi$ at each step of the TMLE procedure.
getLightFit

**Author(s)**

Antoine Chambaz, Pierre Neuvial

**See Also**

tmle.npvi

**Examples**

```
FALSE
```

---

**getLightFit**

* Makes Lighter Fitted Object

**Description**

Makes a lighter version of a fitted object by removing elements containing data.

**Usage**

```
getLightFit(fit)
```

**Arguments**

- **fit**
  
  A fitted object.

**Details**

Most of the space used by a fitted object is not necessary for prediction. This concerns, for instance, the "residuals", "effects", "fitted.values", and "model" entries of a linear model fitted by `lm`. These entries can thus be removed from the object without affecting the model predictions. This function is currently only implemented for fitted objects that derive from class 'lm' or 'rpart'. It is mainly for internal use.

**Value**

Returns the same object as the input without the entries that contain data.

**Author(s)**

Antoine Chambaz, Pierre Neuvial
getObs.NPVI

__Retrieves the Observations__

**Description**

Retrieves the matrix of observations involved in the TMLE procedure.

**Usage**

```r
## S3 method for class 'NPVI'
getObs(this, tabulate, ...)
```

**Arguments**

- `this` An object of class `TMLE.NPVI`.
- `tabulate` A logical, to specify whether it is the original data set that is retrieved (if `FALSE`) or a tabulated version of it (otherwise), for internal use only. If `tabulate` is missing then the value attached to the input object is used.
- ... Not used.

**Value**

Either the original data set involved in the TMLE procedure or a tabulated version of it.

**Author(s)**

Antoine Chambaz, Pierre Neuvial

**See Also**

- `tmle.npvi`

**Examples**

```r
FALSE
```
getPsi.SD.NPVI

Returns Current Estimator

Description

Returns the current value of the estimator.

Usage

## S3 method for class 'NPVI'
geta Psi(this, ...)

Arguments

- **this**: An object of class `TMLE.NPVI`.
- **...**: Not used.

Value

Retrieves the current value of the estimator $\Psi(P^n_k)$ of the parameter of interest. Its computation involves simulation of a large number of iid copies of $(X, W)$ under $P^n_k$.

Author(s)

Antoine Chambaz, Pierre Neuvial

See Also

tmle.npvi, getHistory, getPsiSd

Examples

FALSE

getPsiSd.NPVI

Returns Current Estimated Standard Deviation of the Estimator

Description

Returns the current value of the estimated standard deviation of the current estimator.

Usage

## S3 method for class 'NPVI'
geta PsiSd(this, ...)

Arguments

- **this**: An object of class `TMLE.NPVI`.
- **...**: Not used.

Value

Retrieves the current value of the estimated standard deviation of the estimator $\Psi(P^n_k)$ of the parameter of interest. Its computation involves simulation of a large number of iid copies of $(X, W)$ under $P^n_k$.
getPValue.matrix

Arguments

this          An object of class TMLE.NPVI.
...           Not used.

Value

Retrieves the estimated standard deviation of the current estimator \( \Psi(P^k_n) \) of the parameter of interest. Its computation involves simulation of a large number of iid copies of \((X, W)\) under \(P^k_n\).

Author(s)

Antoine Chambaz, Pierre Neuvial

See Also

tmle.npvi, getHistory, getPsi

Examples

false

getchpvalueNmatrix
Calculates a p-value from a matrix object of type 'history'

Description

Calculates a p-value from a matrix object of type 'history'

Usage

## S3 method for class 'matrix'
getchpvalueNthisL wrtNphi \] trueL nobsL NNNI

Arguments

this          The history of a TMLE procedure.
wrtNphi        A logical equal to TRUE by default, which means that \(psi_n\) is compared with \(phi_n\). Otherwise, \(psi_n\) is compared with 0.
nobs          An integer, the associated number of observations.
...           Not used.

Value

Returns the p-value of the two-sided test of \(\Psi(0) = \Phi(0)\) of \(\Psi(0) = 0\", according to the value of wrt.phi.
Author(s)
    Antoine Chambaz, Pierre Neuvial

See Also
    tmle.npvi, getHistory.NPVI, as.character.NPVI

Examples
    FALSE
**getSample**

Generates Simulated Data

**Description**

Generates a run of simulated observations of the form \((W, X, Y)\) to investigate the "effect" of \(X\) on \(Y\) taking \(W\) into account.

**Usage**

```r
getSample(n, 0, lambda0, p = rep(1/3, 3), omega = rep(1/2, 3),
Sigma1 = matrix(c(1, 1/sqrt(2), 1/sqrt(2), 1), 2, 2), sigma2 = omega[2]/5,
Sigma3 = Sigma1, f = identity, verbose = FALSE)
```

**Arguments**

- **n**
  - An integer, the number of observations to be generated.

- **0**
  - A 3x3 numeric matrix or data.frame. Rows are 3 baseline observations used as "class centers" for the simulation. Columns are:
    - \(W\), baseline covariate (e.g. DNA methylation), or "confounder" in a causal model.
    - \(X\), continuous exposure variable (e.g. DNA copy number), or "cause" in a causal model, with a reference value \(x_0\) equal to \(0[2, "X"]\).
    - \(Y\), outcome variable (e.g. gene expression level), or "effect" in a causal model.

- **lambda0**
  - A function that encodes the relationship between \(W\) and \(Y\) in observations with levels of \(X\) equal to the reference value \(x_0\).

- **p**
  - A vector of length 3 whose entries sum to 1. Entry \(p[k]\) is the probability that each observation belong to class \(0[k,\] \. Defaults to \(\text{rep}(1/3, 3)\).

- **omega**
  - A vector of length 3 with positive entries. Entry \(omega[k]\) is the standard deviation of \(W\) in class \(k\) (on a logit scale). Defaults to \(\text{rep}(1/2, 3)\).

- **Sigma1**
  - A 2x2 covariance matrix of the random vector \((X, Y)\) in class \(1\), assumed to be bivariate Gaussian with mean \(0[1, c("X","Y")]\). Defaults to \(\text{matrix}(c(1, 1/sqrt(2), 1/sqrt(2))\).

- **sigma2**
  - A positive numeric, the variance of the random variable \(Y\) in class \(2\), assumed to be univariate Gaussian with mean \(0[2,"Y"]\). Defaults to \(omega[2]/5\).

- **Sigma3**
  - A 2x2 covariance matrix of the random vector \((X, Y)\) in class \(3\), assumed to be bivariate Gaussian with mean \(0[3, c("X","Y")\]. Defaults to \(\text{Sigma1}\).

- **f**
  - A function involved in the definition of the parameter of interest \(\psi\), which must satisfy \(f(0) = 0\) (see Details). Defaults to \(\text{identity}\).

- **verbose**
  - Prescribes the amount of information output by the function. Defaults to \(\text{FALSE}\).
Details

The parameter of interest is defined as $\psi = \Psi(P)$ with

$$
\Psi(P) = \frac{E_P[f(X - x_0) \ast (\theta(X, W) - \theta(x_0, W))]}{E_P[f(X - x_0)^2]},
$$

with $P$ the distribution of the random vector $(W, X, Y)$, $\theta(X, W) = E_P[Y|X, W]$, $x_0$ the reference value for $X$, and $f$ a user-supplied function such that $f(0) = 0$ (e.g., $f = \text{identity}$, the default value). The value $\psi$ is obtained using the known $\theta$ and the joint empirical distribution of $(X, W)$ based on the same run of observations as in obs. Seeing $W, X, Y$ as DNA methylation, DNA copy number and gene expression, respectively, the simulation scheme implements the following constraints:

- There are two or three copy number classes: normal regions ($k=2$), and regions of copy number gains and/or losses ($k=1$ and/or $k=3$).
- In normal regions, gene expression levels are negatively correlated with DNA methylation.
- In regions of copy number alteration, copy number and expression are positively correlated.

Value

Returns a list with the following tags:

- **obs**: A matrix of $n$ observations. The $c(W, X, Y)$ columns of obs have the same interpretation as the columns of the input argument $0$.
- **psi**: A numeric, approximated value of the true parameter $\psi$ obtained using the known $\theta$ and the joint empirical distribution of $(X, W)$ based on the same run of observations as in obs. The larger the sample size, the more accurate the approximation.
- **g**: A function, the known positive conditional probability $P(X = x_0|W)$.
- **mu**: A function, the known conditional expectation $E_P(X|W)$.
- **muAux**: A function, the known conditional expectation $E_P(X|X \neq x_0, W)$.
- **theta**: A function, the known conditional expectation $E_P(Y|X, W)$.
- **theta0**: A function, the known conditional expectation $E_P(Y|X = x_0, W)$.
- **sigma2**: A positive numeric, the known expectation $E_P(f(X - x_0)^2)$.
- **effIC**: A function, the known efficient influence curve of the functional $\Psi$ at $P$, assuming that the reference value $x_0 = 0$.
- **varIC**: A positive numeric, approximated value of the variance of the efficient influence curve of the functional $\Psi$ at $P$ and evaluated at $O$, obtained using the same run of observations as in obs. The larger the sample size, the more accurate the approximation.

Author(s)

Antoine Chambaz, Pierre Neuvial
References


Examples

```r
## Parameters for the simulation (case 'f=identity')
0 <- cbind(W=c(0.05218652, 0.01113460), X=c(2.722713, 9.362432), Y=c(-0.4569579, 1.2470822))
0 <- rbind(NA, 0)
lambda0 <- function(W) {-W}
p <- c(0, 1/2, 1/2)
omega <- c(0, 3, 3)
S <- matrix(c(10, 1, 1, 0.5), 2, 2)

## Simulating a data set of 200 i.i.d. observations
sim <- getSample(2e2, 0, lambda0, p, omega=omega, sigma2=1, Sigma3=S)
str(sim)

obs <- sim$obs
head(obs)
pairs(obs)

## Adding (dummy) baseline covariates
V <- matrix(runif(3*nrow(obs)), ncol=3)
colnames(V) <- paste("v", 1:3, sep="")
obsv <- cbind(V, obs)
head(obsv)

## True psi and confidence intervals (case 'f=identity')
sim001 <- getSample(1e4, 0, lambda0, p=p, omega=omega, sigma2=1, Sigma3=S)
truePsi1 <- sim001$psi

confInt01 <- truePsi1+c(-1, 1)*qnorm(.975)*sqrt((sim001$varIC/nrow(sim001$obs))
confInt1 <- truePsi1+c(-1, 1)*qnorm(.975)*sqrt((sim001$varIC/nrow(obs))

msg <- "\nCase f=identity:\n"
msg <- c(msg, "\ntrue psi is: " , paste(signif(truePsi1, 3)), "\n")
msg <- c(msg, "\n95%-confidence interval for the approximation is: ",
paste(signif(confInt01, 3)), "\n")
msg <- c(msg, "\nOptimal 95%-confidence interval is: ",
paste(signif(confInt1, 3)), "\n")
cat(msg)

## True psi and confidence intervals (case 'f=atan')
f2 <- function(x) (1.atan(x/1))
sim02 <- getSample(1e4, 0, lambda0, p=p, omega=omega, sigma2=1, Sigma3=S, f=f2);
truePsi2 <- sim02$psi;

confInt02 <- truePsi2+c(-1, 1)*qnorm(.975)*sqrt((sim02$varIC/nrow(sim02$obs))
confInt2 <- truePsi2+c(-1, 1)*qnorm(.975)*sqrt((sim02$varIC/nrow(obs))
```

learnCondExpX2givenW  Estimation of Cond. Expect. of X^2 Given W

Description

Function for the estimation of the conditional expectation of $X^2$ given $W$ when flavor is set to "learning".

Arguments

- **obs**: The matrix of observations, see for instance the obs argument of the function tmle.npvi.
- **light**: A logical, kept for compatibility, which should be set to TRUE (its default value). This requires that the result of each fit be reduced in size (for a faster execution). Currently implemented only for flavor learning.

Value

Returns the fitted object.

Author(s)

Antoine Chambaz, Pierre Neuvial

See Also

learnG, learnMuAux, learnTheta, learnCondExpXYgivenW, learnDevG, learnDevMu, learnDevTheta
learnCondExpXYgivenW  Estimation of Cond. Expect. of XY Given W

Description

Function for the estimation of the conditional expectation of $XY$ given $W$ when flavor is set to "learning".

Arguments

- **obs**: The matrix of observations, see for instance the obs argument of the function `tmle.npvi`.
- **light**: A logical, kept for compatibility, which should be set to TRUE (its default value). This requires that the result of each fit be reduced in size (for a faster execution). Currently implemented only for flavor learning.

Value

Returns the fitted object.

Author(s)

Antoine Chambaz, Pierre Neuvial

See Also

`learnG`, `learnMuAux`, `learnTheta`, `learnCondExpX2givenW`, `learnDevG`, `learnDevMu`, `learnDevTheta`

learnDevG  Estimation of Cond. Expect. of $((X=0)-g_W)*effIC1$ Given W

Description

Function for the estimation of the conditional expectation of $((X=0)-g_W)*effIC1$ given $W$ when flavor is set to "learning".

Arguments

- **obs**: The matrix of observations, see for instance the obs argument of the function `tmle.npvi`.
- **effIC1**: The vector `effIC1` of the first component of the efficient influence curve, as currently estimated, evaluated at our observations.
- **gW**: The vector `gW` of the conditional probability that $X = 0$ given $W$, as currently estimated, evaluated at our observations.
LearnDevMu

light A logical, kept for compatibility, which should be set to TRUE (its default value). This requires that the result of each fit be reduced in size (for a faster execution). Currently implemented only for flavor learning.

verbose Prescribes the amount of information output by the function. Defaults to FALSE.

Additional arguments possibly needed.

Value

Returns the fitted object.

Author(s)

Antoine Chambaz, Pierre Neuvial

See Also

learnG, learnMuAux, learnTheta, learnCondExpX2givenW, learnCondExpXYgivenW, learnDevMu, learnDevTheta

learnDevMu  Estimation of Cond. Expect. of (X-muw)*effIC1 Given W

Description

Function for the estimation of the conditional expectation of (X-muw)*effIC1 given W when flavor is set to "learning".

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>obs</td>
<td>The matrix of observations, see for instance the obs argument of the function tmle.npvi.</td>
</tr>
<tr>
<td>effic1</td>
<td>The vector effic1 of the first component of the efficient influence curve, as currently estimated, evaluated at our observations.</td>
</tr>
<tr>
<td>muw</td>
<td>The vector muw of the conditional expectation of X given W, as currently estimated, evaluated at our observations.</td>
</tr>
<tr>
<td>light</td>
<td>A logical, kept for compatibility, which should be set to TRUE (its default value). This requires that the result of each fit be reduced in size (for a faster execution). Currently implemented only for flavor learning.</td>
</tr>
<tr>
<td>verbose</td>
<td>Prescribes the amount of information output by the function. Defaults to FALSE.</td>
</tr>
<tr>
<td>...</td>
<td>Additional arguments possibly needed.</td>
</tr>
</tbody>
</table>

Value

Returns the fitted object.
learnDevTheta

Author(s)
Antoine Chambaz, Pierre Neuvial

See Also
learnG, learnMuAux, learnTheta, learnCondExpX2givenW, learnCondExpXYgivenW, learnDevG, learnDevTheta

learnDevTheta

Estimation of Cond. Expect. of (Y-thetaXW)^2 Given (X,W)

Description
Function for the estimation of the conditional expectation of (Y-thetaXW)^2 given (X,W) when flavor is set to "learning".

Arguments
obs
The matrix of observations, see for instance the obs argument of the function tmle.npvi.

thetaxW
The vector thetaxW of the conditional expectation of Y given (X, W), as currently estimated, evaluated at our observations.

light
A logical, kept for compatibility, which should be set to TRUE (its default value). This requires that the result of each fit be reduced in size (for a faster execution). Currently implemented only for flavor learning.

verbose
Prescribes the amount of information output by the function. Defaults to FALSE.

... Additional arguments possibly needed.

Value
Returns the fitted object.

Author(s)
Antoine Chambaz, Pierre Neuvial

See Also
learnG, learnMuAux, learnTheta, learnCondExpX2givenW, learnCondExpXYgivenW, learnDevG, learnDevMu
learnG  

Estimation of Cond. Prob. of \( X=x_0 \) Given \( W \)

Description

Function for the estimation of the conditional probability that \( X = x_0 \) (the reference value for \( X \)) given \( W \), version based on `glm`.

Arguments

- **obs**: The matrix of observations, see for instance the obs argument of the function `tmle.npvi`.
- **theX0**: The reference value for \( X \).
- **light**: A logical, kept for compatibility, which should be set to TRUE (its default value). This requires that the result of each fit be reduced in size (for a faster execution). Currently implemented only for flavor `learning`.
- **...**: Additional arguments possibly needed.

Value

Returns the fitted object.

Author(s)

Antoine Chambaz, Pierre Neuvial

See Also

`learnMuAux`, `learnTheta`, `learnCondExpX2givenW`, `learnCondExpXYgivenW`, `learnDevG`, `learnDevMu`, `learnDevTheta`

learningLib  

Description

List of default learning algorithms to use in `tmle.npvi` when `flavor` is set to "learning".
**learnMuAux**

*Estimation of Cond. Expect. of X Given (X!=x_0, W)*

**Description**

Function for the estimation of the conditional expectation of \( X \) given \( (X \neq x_0, W) \), version based on 'glm'.

**Arguments**

- **obs**
  - The matrix of observations, see for instance the obs argument of the function tmle.npvi, where only observations with \( X \neq 0 \) are kept.

- **light**
  - A logical, kept for compatibility, which should be set to TRUE (its default value). This requires that the result of each fit be reduced in size (for a faster execution). Currently implemented only for flavor learning.

- ... Additional arguments possibly needed.

**Value**

Returns the fitted object.

**Author(s)**

Antoine Chambaz, Pierre Neuvial

**See Also**

learnG, learnTheta, learnCondExpX2givenW, learnCondExpXYgivenW, learnDevG, learnDevMu, learnDevTheta

---

**learnTheta**

*Estimation of Cond. Expect. of Y given (X,W)*

**Description**

Function for the estimation of the conditional expectation of \( Y \) given \( (X,W) \), version based on 'glm'.

**Arguments**

- **obs**
  - The matrix of observations, see for instance the obs argument of the function tmle.npvi, where only observations with \( X \neq 0 \) are kept.

- **light**
  - A logical, kept for compatibility, which should be set to TRUE (its default value). This requires that the result of each fit be reduced in size (for a faster execution). Currently implemented only for flavor learning.

- ... Additional arguments possibly needed.
Value

Returns the fitted object.

Author(s)

Antoine Chambaz, Pierre Neuvial

See Also

learnG, learnMuAux, learnCondExpX2givenW, learnCondExpXYgivenW, learnDevG, learnDevMu, learnDevTheta

---

**predict.SL.glm.condExpX2givenW**

*SL Wrapper for Estimation of Cond. Expect. of \( X^2 \) Given \( W \)*

**Description**

Prediction algorithm wrapper for SuperLearner, for the estimation of the conditional expectation of \( X^2 \) given \( W \).

**Arguments**

- `object`
- `newdata`
- ...

**Value**

Returns a prediction.

**Author(s)**

Antoine Chambaz, Pierre Neuvial

**See Also**

SL.glm.condExpX2givenW
predict.SL.glm.condExpXYgivenW

SL Wrapper for Estimation of Cond. Expect. of XY Given W

Description
Prediction algorithm wrapper for SuperLearner.

Arguments
object
newdata
...

Value
Returns a prediction.

Author(s)
Antoine Chambaz, Pierre Neuvial

See Also
SL.glm.condExpXYgivenW

predict.SL.glm.g

SL Wrapper for Estimation of Cond. Prob. of X=0 Given W

Description
Prediction algorithm wrapper for SuperLearner.

Arguments
object
newdata
...

Value
Returns a prediction.
predict.SL.glm.theta  SL Wrapper for Estimation of Cond. Expect. of Y Given (X,W)

Description
Prediction algorithm wrapper for SuperLearner.

Arguments
- object
- newdata
- ...

Value
Returns a prediction.

Author(s)
Antoine Chambaz, Pierre Neuvial

See Also
SL.glm.g

setConfLevel.NPVI  Sets Confidence Level

Description
Sets the confidence level of a TMLE.NPVI object.

Usage
## S3 method for class 'NPVI'
setConfLevel(this, confLevel, ...)

**Arguments**

- `this`  
  An object of class `TMLE.NPVI`.
- `confLevel`  
  A numeric, confidence interval level.
- `...`  
  Not used.

**Author(s)**

Antoine Chambaz, Pierre Neuvial

**See Also**

- `as.character.NPVI`

**Examples**

```r
FALSE
```

---

**Description**

Prediction algorithm wrapper for SuperLearner, for the estimation of the conditional expectation of $X^2$ given $W$.

**Arguments**

- `Y`
- `X`
- `newX`
- `family`
- `obsWeights`
- `...`

**Value**

Returns a fitted object.

**Author(s)**

Antoine Chambaz, Pierre Neuvial

**See Also**

- `learnCondExpX2givenW`
- `predict.SL.glm.condExpX2givenW`
Description
Prediction algorithm wrapper for SuperLearner, for the estimation of the conditional expectation of \(XY\) given \(W\).

Arguments
- \(Y\)
- \(X\)
- \(\text{newX}\)
- \(\text{family}\)
- \(\text{obsWeights}\)
- ...

Value
Returns a fitted object.

Author(s)
Antoine Chambaz, Pierre Neuvial

See Also
learnCondExpXYgivenW, predict.SL.glm.condExpXYgivenW

Description
Prediction algorithm wrapper for SuperLearner, for the estimation of the conditional probability of \(X = 0\) given \(W\).

Arguments
- \(Y\)
- \(X\)
- \(\text{newX}\)
- \(\text{family}\)
- \(\text{obsWeights}\)
- ...

SL.glm.g

\(SL.Wrapper\ for\ Estimation\ of\ Cond.\ Prob.\ of\ X=0\ Given\ W\)
**SL.glm.theta**

<table>
<thead>
<tr>
<th>Value</th>
<th>Returns a fitted object.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Author(s)</strong></td>
<td>Antoine Chambaz, Pierre Neuvial</td>
</tr>
<tr>
<td><strong>See Also</strong></td>
<td>learnG, predict.SL.glm.g</td>
</tr>
</tbody>
</table>

---

**Description**

Prediction algorithm wrapper for SuperLearner, for the estimation of the conditional expectation of \( Y \) given \((X, W)\).

| Arguments | \( Y \) \( X \) \( newX \) \( family \) \( obsWeights \) ... |
|-----------|-------------------|-----------------|-----------------|-------------------|

<table>
<thead>
<tr>
<th>Value</th>
<th>Returns a fitted object.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Author(s)</strong></td>
<td>Antoine Chambaz, Pierre Neuvial</td>
</tr>
<tr>
<td><strong>See Also</strong></td>
<td>learnTheta, predict.SL.glm.theta</td>
</tr>
</tbody>
</table>

---

**superLearningLib**

| Description | List of default libraries of algorithms to use in `tmle.npvi` when `flavor` is set to "superLearning". |

---
Sample breast cancer data from TCGA

**Description**

Expression, DNA copy number, and DNA methylation data of 125 genes of chromosome 21 for 463 breast cancer samples from TCGA.

**Usage**

tcga2012brca

**Format**

A list of matrices. Each matrix corresponds to a particular gene and has the following columns:

- **Y**: gene expression level (outcome variable)
- **X**: DNA copy number (continuous exposure variable)
- **W[1]...W[k]**: DNA methylation (k baseline covariates)

**Details**

These data were obtained using the scripts located in testScripts/tcga2012brca. See the tmle-npvi.pdf vignette for more details on the preparation of the data set.

Gene names and genomic coordinates are stored in the names of the list.

**Author(s)**

Antoine Chambaz, Pierre Neuvial

**Source**

Raw data may be retrieved from https://tcga-data.nci.nih.gov/docs/publications/brca_2012/

- **methylation** http://tcga-data.nci.nih.gov/docs/publications/brca_2012/BRCA.methylation.27k.450k.466.zip
- **copy number data** http://tcga-data.nci.nih.gov/docs/publications/brca_2012/BRCA.GISTIC2.tar.gz
- **expression** http://tcga-data.nci.nih.gov/docs/publications/brca_2012/BRCA.exp.466.med.txt
- **annotation data for methylation probes** http://supportres.illumina.com/documents/myillumina/b78d361a-def5-4adb-ab38-e8990625f053/humanmethylation450_15017482_v1-2.csv
References


Examples

data(tcgA2012brca)
nms <- names(tcgA2012brca)
ii <- grep("TP53", nms)
obs <- tcga2012brca[ii]
pairs(obs, main=names[|i|])

thr <- 0.02
whichSmall <- which(abs(obs[, "X"] <= thr)
obs[whichSmall, "X"] <- 0

## the code below takes ~20s to run
## Not run: res <- tmle.npvi(obs)

---

**tmle.npvi**

**Targeted Minimum Loss Estimation of NPVI**

**Description**

Carries out the targeted minimum loss estimation (TMLE) of a non-parametric variable importance measure of a continuous exposure.

**Usage**

```r
tMLE.npvi(obs, f = identity, nMax = 30L, flavor = c("learning", "superLearning"), lib = list(), nodes = 1L, cvControl = NULL,
  family = c("parsimonious", "gaussian"), cleverCovTheta = FALSE,
  bound = 1, B = 1e+05, trueGMu = NULL, iter = 5L, stoppingCriteria = list(mic = 0.01,
  div = 0.01, psi = 0.1), gmin = 0.05, gmax = 0.95,
  mumin = quantile(f(obs[obs[, "X"] != 0, "X"]), type = 1, probs = 0.01),
  mumax = quantile(f(obs[obs[, "X"] != 0, "X"]), type = 1, probs = 0.99),
  verbose = FALSE, tabulate = TRUE, exact = TRUE, light = TRUE)
```

**Arguments**

- **obs**
  - A n x p matrix or data frame of observations, with p ≥ 3.
  - Column "X" corresponds to the continuous exposure variable (e.g. DNA copy number), or "cause" in a causal model, with a reference value x0 equal to 0.
• Column "y" corresponds to the outcome variable (e.g. gene expression level), or "effect" in a causal model.
• All other columns are interpreted as baseline covariates W taken into account in the definition of the "effect" of X on Y.

\[ f \]
A function involved in the definition of the parameter of interest \( \psi \), which must satisfy \( f(0) = 0 \) (see Details). Defaults to identity.

\[ \text{nm} \]
An integer (defaults to 30L; 10L is the smallest authorized value and we recommend a value less than 50L for reasonable computational time) indicating the maximum number of observed values of \( X \neq 0 \) which are used to create the supports of the conditional distributions of \( X \) given \( W \) and \( X \neq 0 \) involved in the simulation under \( P_n^k \) when family is set to "parsimonious".

\[ \text{flavor} \]
Indicates whether the construction of the relevant features of \( P_n^0 \) and \( P_n^k \), the (non-targeted yet) initial and (targeted) successive updated estimators of the true distribution of \( (W, X, Y) \) relies on the Super Learning methodology (option "superLearning") or not (option "learning", default value). In the former case, the SuperLearner package is loaded.

\[ \text{lib} \]
A list providing the function `tmle.npvi` with the necessary algorithms involved in the estimation of the features of interest. If empty (default) then the default algorithms are used. See Details.

\[ \text{nodes} \]
An integer, which indicates how many nodes should be involved in the computation of the TMLE when it relies on the "superLearning" flavor. Defaults to 1. If larger than 1, then the parallel package is loaded and a cluster with `nodes` nodes is created and exploited.

\[ \text{cvControl} \]
'NULL' (default value) or an integer indicating how many folds are involved in the Super Learning procedure. If flavor and cvControl are simultaneously set to "superLearning" and NULL then the Super Learning procedure relies on 10-fold cross-validation.

\[ \text{family} \]
Indicates whether the simulation of the conditional distribution of \( X \) given \( W \) under \( P_n^k \) (the initial estimator if \( k = 0 \) or its \( k \)th update if \( k \geq 1 \)) should be based on a weighted version of the empirical measure (case "parsimonious", default value and faster execution) or on a Gaussian model (case "gaussian").

\[ \text{cleverCovTheta} \]
A logical, indicating whether the one-step (if FALSE, default value) or the two-step (if TRUE) updating procedure should be carried out.

\[ \text{bound} \]
A positive numeric (defaults to 1), upper-bound on the absolute value of the fluctuation parameter.

\[ B \]
An integer (defaults to 1e5) indicating the sample size of the data set simulated under each \( P_n^k \) to compute an approximated value of \( \Psi(P_n^k) \), the parameter of interest at \( P_n^k \). The larger \( B \), the more accurate the approximation.

\[ \text{trueGMu} \]
Either NULL (default value) if the true conditional probability \( g(W) = P(X = 0 | W) \) and conditional expectation \( \mu_{\text{Aux}}(W) = E_P(X | X \neq 0, W) \) are not both known beforehand. Otherwise, a list with tags g and muAux and respective values the two respective functions (see the related outputs of getSfample).

\[ \text{iter} \]
An integer (defaults to 5) indicating a maximal number of updates.

\[ \text{stoppingCriteria} \]
A list providing tuning parameters for the stopping criteria. Defaults to list(mic=0.01, div=0.01, see Details.
gmin
A positive numeric, lower-bound on the range of values of the estimated probabilities \( P_n^k(X = 0|W) \) that \( X \) be equal to its reference value 0 given \( W \). Defaults to 5e-2, and must be smaller than gmax.

gmax
A positive numeric smaller than 1, upper-bound on the range of values of the estimated probabilities \( P_n^k(X = 0|W) \) that \( X \) be equal to its reference value 0 given \( W \). Defaults to 95e-2, and must be larger than gmin.
mumin
A numeric, lower-bound on the range of values of the estimated conditional expectation \( P_n^k(X|X \neq 0, W) \) of \( X \) given \( X \neq 0 \) and \( W \). Defaults to the first percentile of \( X \), and must be smaller than mumax.
mumax
A numeric, upper-bound on the range of values of the estimated conditional expectation \( P_n^k(X|X \neq 0, W) \) of \( X \) given \( X \neq 0 \) and \( W \). Defaults to the 99th percentile of \( X \), and must be larger than mumin.
verbose
Prescribes the amount of information output by the function. Defaults to FALSE.
tabulate
A logical, kept for compatibility, which should be set to TRUE (its default value). This requires that the joint distribution of \((W, X, Y)\) under \( P_n^k \) be a weighted version of the empirical measure (for a faster execution).
exact
A logical, kept for compatibility, which should be set to TRUE (its default value). This requires that the successive updates of the estimators of the relevant features of the true distribution of \((W, X, Y)\) be derived from the current estimators based on exact formulas rather than on their first-order approximations.
light
A logical, kept for compatibility, which should be set to TRUE (its default value). This requires that the result of each fit be reduced in size (for a faster execution). Currently implemented only for flavor learning.

Details

The parameter of interest is defined as \( \psi = \Psi(P) \) with

\[
\Psi(P) = \frac{E_P[f(X) \ast (\theta(X, W) - \theta(0, W))]}{E_P[f(X)^2]},
\]

with \( P \) the distribution of the random vector \((W, X, Y)\), \( \theta(X, W) = E_P[Y|X, W] \), 0 the reference value for \( X \), and \( f \) a user-supplied function such that \( f(0) = 0 \) (e.g., \( f = \text{identity} \), the default value).

The TMLE procedure stops when the maximal number of iterations, iter, is reached or when at least one of the following criteria is met:

- The empirical mean \( P_n^{\text{effIC}}(P_n^{k+1}) \) of the efficient influence curve at \( P_n^{k+1} \) scaled by the estimated standard deviation of the efficient influence curve at \( P_n^{k+1} \) is smaller, in absolute value, than mic.
- The total variation (TV) distance between \( P_n^{k} \) and \( P_n^{k+1} \) is smaller than div.
- The change between the successive values \( P_{si}(P_n^{k}) \) and \( P_{si}(P_n^{k+1}) \) is smaller than psi.

If lib is an empty list (list(), default value) then the default algorithms for the chosen flavor are loaded (learninglib when flavor is set to "learning" or superlearninglib when flavor is
set to "superLearning"). A valid lib argument must mimic the structure of either learningLib or superLearningLib, depending on flavor.

The "superLearning" flavor requires the SuperLearner package and, by default, the e1071, gam, glmnet, polspline and randomForest packages.

If family is set to "parsimonious" (recommended) then the packages sgeostat and geometry are required.

Value

Returns an object of class "NPVI" summarizing the different steps of the TMLE procedure. The method getHistory outputs the "history" of the procedure (see getHistory). The object notably includes the following information:

- obs: The matrix of observations used to carry out the TMLE procedure. Use the method getObs to retrieve it.
- psi: The TMLE of the parameter of interest. Use the method getPsi to retrieve it.
- psi.sd: The estimated standard deviation of the TMLE of the parameter of interest. Use the method getPsiSD to retrieve it.

Author(s)

Antoine Chambaz, Pierre Neuvial

References


See Also

getSample, getHistory

Examples

```r
set.seed(12345)

## Simulating a data set and computing the true value of the parameter

## Parameters for the simulation (case 'f=identity')
0 <- cbind(W=c(0.05218652, 0.01113460),
           X=c(2.722713, 9.362432),
           Y=c(-0.4569579, 1.2470822))
0 <- rbind(NA, 0)
lambda0 <- function(W) {-W}
p <- c(0, 1/2, 1/2)
```
sim <- getSample(2e2, 0, lambda0, p=p, omega=omega, sigma2=1, Sigma3=S)
ob <- sim

## Adding (dummy) baseline covariates
V <- matrix(runif(3*nrow(obs)), ncol=3)
colnames(V) <- paste("V", 1:3, sep="")
ob <- cbind(V, obs)

## Caution! MAKING '0' THE REFERENCE VALUE FOR 'X'
X0 <- 0[2,2]
obsC <- obs
obsC[, "X"] <- obsC[, "X"] - X0
obs <- obsC

truePsi <- sim

confInt0 <- truePsi + c(-1, 1)*qnorm(.975)*sqrt(sim$varIC/nrow(sim$obs))
confInt <- truePsi + c(-1, 1)*qnorm(.975)*sqrt(sim$varIC/nrow(obs))
cat("\nCase f=identity:\n")
msg <- paste("true psi is: ", signif(truePsi, 3), "\n", sep="")
msg <- paste(msg, "t95%-confidence interval for the approximation is: ", signif(confInt0, 3), "\n", sep="")
msg <- paste(msg, "toptimal 95%-confidence interval is: ", signif(confInt, 3), "\n", sep="")
cat(msg)

## TMLE procedure

## Running the TMLE procedure
npvi <- tmle.npvi(obs, f=identity, flavor="learning", B=5e4, nMax=10)

## Summarizing its results
npvi
setConfLevel(npvi, 0.9)
npvi

history <- getHistory(npvi)
print(round(history, 4))

hp <- history[, "psi"]
hs <- history[, "sic"]
hs[1] <- NA
ics <- c(-1, 1) %*% t(qnorm(0.975)*hs/sqrt(nrow(getObs(npvi)))))
pch <- 20
ylim <- range(c(confInt, hp, ics+hp), na.rm=TRUE)

xs <- (1:length(hs))-1
plot(xs, hp, ylim=ylim, pch=pch, xlab="Iteration", ylab=expression(\psi[n]),
   xaxp=c(0, length(hs)-1, length(hs)-1))
dummy <- sapply(seq(along=xs), function(x) lines(c(xs[x],xs[x]), hp[x]+ics[, x]))

abline(h=confInt, col=4)
abline(h=confInt0, col=2)
# Index

*Topic **datasets**
  - `tcga2012brca`, 26

  - `as.character (as.character.NPVI)`, 2
  - `as.character.NPVI`, 2

  - `extractW`, 3
  - `extractXW`, 4

  - `getHistory (getHistory.NPVI)`, 4
  - `getHistory.NPVI`, 4
  - `getLightFit`, 6
  - `getObs (getObs.NPVI)`, 7
  - `getObs.NPVI`, 7
  - `getPsi (getPsi.NPVI)`, 8
  - `getPsi.NPVI`, 8
  - `getPsiSd (getPsiSd.NPVI)`, 8
  - `getPsiSd.NPVI`, 8
  - `getPValue (getPValue.matrix)`, 9
  - `getPValue.matrix`, 9
  - `getPValue.NPVI`, 10
  - `getSample`, 11

  - `learnCondExpX2givenW`, 14
  - `learnCondExpXYgivenW`, 15
  - `learnDevG`, 15
  - `learnDevMu`, 16
  - `learnDevTheta`, 17
  - `learnG`, 18
  - `learningLib`, 18
  - `learnMuAux`, 19
  - `learnTheta`, 19

  - `predict.SL.glm.condExpX2givenW`, 20
  - `predict.SL.glm.condExpXYgivenW`, 21
  - `predict.SL.glm.g`, 21
  - `predict.SL.glm.theta`, 22

  - `setConfLevel`, 3
  - `setConfLevel (setConfLevel.NPVI)`, 22
  - `setConfLevel.NPVI`, 22

- `SL.glm.condExpX2givenW`, 23
- `SL.glm.condExpXYgivenW`, 24
- `SL.glm.g`, 24
- `SL.glm.theta`, 25
- `superLearningLib`, 25

- `tcga2012brca`, 26
- `tmle.npvi`, 3, 4, 27