Package ‘SLEMI’

October 7, 2019

Title Statistical Learning Based Estimation of Mutual Information

Version 1.0

Date 2019-09-19

Description
The implementation of the algorithm for estimation of mutual information and channel capacity from experimental data by classification procedures (logistic regression). Technically, it allows to estimate information-theoretic measures between finite-state input and multivariate, continuous output. Method described in Jetka et al. (2019) <doi:10.1371/journal.pcbi.1007132>.

Depends R (>= 3.6.0)

License LGPL (>= 2)

URL https://github.com/sysbiosig/SLEMI

BugReports https://github.com/sysbiosig/SLEMI/issues

Encoding UTF-8

LazyData true

Imports e1071, ggplot2, ggthemes, gridExtra, nnet, Hmisc, reshape2, stringr, doParallel, caret, corrplot, foreach

Suggests knitr, rmarkdown

VignetteBuilder knitr

RoxygenNote 6.1.1

NeedsCompilation no

Author Tomasz Jetka [aut, cre],
Karol Nienaltowski [ctb],
Michal Komorowski [ctb]

Maintainer Tomasz Jetka <t.jetka@gmail.com>

Repository CRAN

Date/Publication 2019-10-07 14:20:02 UTC

R topics documented:

---

capacity_logreg_algorithm

Main algorithm to calculate channel capacity by SLEMI approach

---

Description

Additional parameters: lr_maxit and maxNWts are the same as in definition of multinom function from nnet package. An alternative model formula (using formula_string arguments) should be provided if data are not suitable for description by logistic regression (recommended only for advanced users). It is recommended to conduct estimation by calling capacity_logreg_main.R.

Usage

```
capacity_logreg_algorithm(data, signal = "signal",
  response = "response", side_variables = NULL,
  formula_string = NULL, model_out = TRUE, cc_maxit = 100,
  lr_maxit = 1000, MaxNWts = 5000)
```

Arguments

data	must be a data.frame object. Cannot contain NA values.
signal	is a character object with names of columns of dataRaw to be treated as channel's input.
response	is a character vector with names of columns of dataRaw to be treated as channel's output
side_variables	(optional) is a character vector that indicates side variables' columns of data, if NULL no side variables are included
formula_string	(optional) is a character object that includes a formula syntax to use in logistic regression model. If NULL, a standard additive model of response variables is assumed. Only for advanced users.
model_out	is the logical indicating if the calculated logisitc regression model should be included in output list
cc_maxit	is the number of iteration of iterative optimisation of the algorithm to esimate channel capacity. Default is 100.

`lr_maxit`	is a maximum number of iteration of fitting algorithm of logistic regression. Default is 1000.
`MaxNWts`	is a maximum acceptable number of weights in logistic regression algorithm. Default is 5000.

Value

a list with three elements:

- output$cc - channel capacity in bits
- output$p_opt - optimal probability distribution
- output$regression - confusion matrix of logistic regression predictions
- output$model - nnet object describing logistic regression model (if model_out=TRUE)

References

[1] Jetka T, Nienaltowski K, Winarski T, Blonski S, Komorowski M, Information-theoretic analysis of multivariate single-cell signaling responses using SLEMI, *PLoS Comput Biol*, 15(7): e1007132, 2019, https://doi.org/10.1371/journal.pcbi.1007132.

Examples

```
tempdata=data_example1
outputCLR1=capacity_logreg_algorithm(data=tempdata, signal="signal", response="response",
formula_string = "signal~response")
```

`capacity_logreg_main` *Estimate channel capacity between discrete input and continuouse output*

Description

The main wrapping function for basic usage of SLEMI package for estimation of channel capacity. Firstly, data is pre-processed (all arguments are checked, observation with NAs are removed, variables are scaled and centered (if scale=TRUE)). Then basic estimation is carried out and (if testing=TRUE) diagnostic tests are computed. If output directory path is given (output_path is not NULL), graphs visualising the data and the analysis are saved there, together with a compressed output object (as .rds file) with full estimation results.

Usage

```
capacity_logreg_main(dataRaw, signal = "input", response = NULL,
  output_path = NULL, side_variables = NULL, formula_string = NULL,
  cc_maxit = 100, lr_maxit = 1000, MaxNWts = 5000, testing = FALSE,
  model_out = TRUE, scale = TRUE, TestingSeed = 1234,
  testing_cores = 1, boot_num = 10, boot_prob = 0.8,
  sidevar_num = 10, traintest_num = 10, partition_trainfrac = 0.6,
  plot_width = 6, plot_height = 4, data_out = FALSE)
```

Arguments

`dataRaw`	must be a data.frame object
`signal`	is a character object with names of columns of dataRaw to be treated as channel's input.
`response`	is a character vector with names of columns of dataRaw to be treated as channel's output
`output_path`	is the directory in which output will be saved
`side_variables`	(optional) is a character vector that indicates side variables' columns of data, if NULL no side variables are included
`formula_string`	(optional) is a character object that includes a formula syntax to use in logistic regression model. If NULL, a standard additive model of response variables is assumed. Only for advanced users.
`cc_maxit`	is the number of iteration of iterative optimisation of the algorithm to esimate channel capacity. Default is 100.
`lr_maxit`	is a maximum number of iteration of fitting algorithm of logistic regression. Default is 1000.
`MaxNWts`	is a maximum acceptable number of weights in logistic regression algorithm. Default is 5000.
`testing`	is the logical indicating if the testing procedures should be executed
`model_out`	is the logical indicating if the calculated logisitc regression model should be included in output list
`scale`	is a logical indicating if the response variables should be scaled and centered before fitting logistic regression
`TestingSeed`	is the seed for random number generator used in testing procedures
`testing_cores`	- number of cores to be used in parallel computing (via doParallel package)
`boot_num`	is the number of bootstrap tests to be performed. Default is 10, but it is recommended to use at least 50 for reliable estimates.
`boot_prob`	is the proportion of initial size of data to be used in bootstrap
`sidevar_num`	is the number of re-shuffling tests of side variables to be performed. Default is 10, but it is recommended to use at least 50 for reliable estimates.
`traintest_num`	is the number of overfitting tests to be performed. Default is 10, but it is recommended to use at least 50 for reliable estimates.
`partition_trainfrac`	is the fraction of data to be used as a training dataset
`plot_width`	- the basic dimnesions (width) of plots, in inches
`plot_height`	- the basic dimnesions (height) of plots, in inches
`data_out`	is the logical indicating if the data should be included in output list

Details

In a typical experiment aimed to quantify information flow a given signaling system, input values $x_1 \leq x_2 \ldots\ldots \leq x_m$, ranging from 0 to saturation are considered. Then, for each input level, x_i, n_i observations are collected, which are represetned as vectors

$$y^i_j \sim P(Y|X = x_i)$$

Within information theory the degree of information transmission is measured as the mutual information

$$MI(X,Y) = \sum_{i=1}^{m} P(x_i) \int_{R^k} P(y|X = x_i) log_2 \frac{P(y|X = x_i)}{P(y)} dy,$$

where $P(y)$ is the marginal distribution of the output. MI is expressed in bits and 2^{MI} can be interpreted as the number of inputs that the system can resolve on average.

The maximization of mutual information with respect to the input distribution, $P(X)$, defines the information capacity, C. Formally,

$$C^* = max_{P(X)} MI(X,Y)$$

Information capacity is expressed in bits and 2^{C^*} can be interpreted as the maximal number of inputs that the system can effectively resolve.

In contrast to existing approaches, instead of estimating, possibly highly dimensional, conditional output distributions P(Y|X =x_i), we propose to estimate the discrete, conditional input distribution, $P(x_i|Y = y)$, which is known to be a simpler problem. Estimation of the MI using estimates of $P(x_i|Y = y)$, denoted here as $\hat{P}(x_i|Y = y)$, is possible as the MI, can be alternatively written as

$$MI(X,Y) = \sum_{i=1}^{m} P(x_i) \int_{R^k} P(y|X = x_i) log_2 \frac{P(x_i|Y = y)}{P(x_i)} dy$$

The expected value (as in above expression) with respect to distribution $P(Y|X = x_i)$ can be approximated by the average with respect to data

$$MI(X,Y) \approx \sum_{i=1}^{m} P(x_i) \frac{1}{n_i} \sum_{j=1}^{n_i} P(y|X = x_i) log_2 \frac{\hat{P}(x_i|Y = y^i_j)}{P(x_i)} dy$$

Here, we propose to use logistic regression as $\hat{P}(x_i|Y = y^i_j)$. Specifically,

$$log \frac{P(x_i|Y = y)}{P(x_m|Y = y)} \approx \alpha_i + \beta_i y$$

Following this approach, channel capacity can be calculated by optimising MI with respect to the input distribution, $P(X)$. However, this, potentially difficult problem, can be divided into two simpler maximization problems, for which explicit solutions exist. Therefore, channel capacity can be obtained from the two explicit solutions in an iterative procedure known as alternate maximization (similarly as in Blahut-Arimoto algorithm) [1].

Additional parameters: lr_maxit and maxNWts are the same as in definition of multinom function from nnet package. An alternative model formula (using formula_string arguments) should be provided if data are not suitable for description by logistic regression (recommended only for advanced users). Preliminary scaling of data (argument scale) should be used similarly as in other data-driven approaches, e.g. if response variables are comparable, scaling (scale=FALSE) can be omitted, while if they represent different phenomenon (varying by units and/or magnitude) scaling is recommended.

Value

a list with several elements:

- output$regression - confusion matrix of logistic regression predictions
- output$cc - channel capacity in bits
- output$p_opt - optimal probability distribution
- output$model - nnet object describing logistic regression model (if model_out=TRUE)
- output$params - parameters used in algorithm
- output$time - computation time of calculations
- output$testing - a 2- or 4-element output list of testing procedures (if testing=TRUE)
- output$testing_pv - one-sided p-values of testing procedures (if testing=TRUE)
- output$data - raw data used in analysis

References

[1] Csiszar I, Tusnady G, Information geometry and alternating minimization procedures, Statistics & Decisions 1 Supplement 1 (1984), 205–237.

[2] Jetka T, Nienaltowski K, Winarski T, Blonski S, Komorowski M, Information-theoretic analysis of multivariate single-cell signaling responses using SLEMI, *PLoS Comput Biol*, 15(7): e1007132, 2019, https://doi.org/10.1371/journal.pcbi.1007132.

Examples

```
tempdata=data_example1
outputCLR1=capacity_logreg_main(dataRaw=tempdata,
signal="signal", response="response",
formula_string = "signal~response")

tempdata=data_example2
outputCLR2=capacity_logreg_main(dataRaw=tempdata,
signal="signal", response=c("X1","X2","X3"),
formula_string = "signal~X1+X2+X3")

#For further details see vignette
```

`data_nfkb` *Data from experiment with NFkB pathway*

Description

In the paper describing methodological aspects of our algorithm we present the analysis of information transmission in NfkB pathway upn the stimulation of TNF-α. Experimental data from this experiment in the form of single-cell time series are attached to the package as a data.frame object and can be accessed using ‘data_nfkb‘ variable. Each row of ‘data_nfkb‘ represents a single observation of a cell. Column ’signal’ indicates the level of TNF-α stimulation for a given cell, while columns 'response_T', gives the normalised ratio of nuclear and cytoplasmic transcription factor as described in Supplementary Methods of the corresponding publication. In the CRAN version of the package we included only a subset of the data (5 time measurements). For the full datasets, please access GitHub pages.

Usage

```
data_nfkb
```

Format

A data frame with 15632 rows and 6 variables:

signal Level of TNFa stimulation

response_0 The concentration of normalised NfkB transcription factor, measured at time 0

response_3 The concentration of normalised NfkB transcription factor, measured at time 3

response_21 The concentration of normalised NfkB transcription factor, measured at time 21

response_90 The concentration of normalised NfkB transcription factor, measured at time 90

response_120 The concentration of normalised NfkB transcription factor, measured at time 120 #'

Details

For each concentration, there are at least 1000 single-cell observation (with the exception of 0.5ng stimulation, where the number of identified cells is almost 900)

Source

in-house experimental data

---

mi_logreg_algorithm *Main algorithm to calculate mutual information by SLEMI approach*

Description

Additional parameters: lr_maxit and maxNWts are the same as in definition of multinom function from nnet package. An alternative model formula (using formula_string arguments) should be provided if data are not suitable for description by logistic regression (recommended only for advanced users). It is recommended to conduct estimation by calling mi_logreg_main.R.

Usage

```
mi_logreg_algorithm(data, signal = "signal", response = "response",
  side_variables = NULL, pinput = NULL, formula_string = NULL,
  lr_maxit = 1000, MaxNWts = 5000, model_out = TRUE)
```

Arguments

`data`	must be a data.frame object. Cannot contain NA values.
`signal`	is a character object with names of columns of dataRaw to be treated as channel's input.
`response`	is a character vector with names of columns of dataRaw to be treated as channel's output
`side_variables`	(optional) is a character vector that indicates side variables' columns of data, if NULL no side variables are included
`pinput`	is a numeric vector with piror probabilities of the input values. Uniform distribution is assumed as default (pinput=NULL).
`formula_string`	(optional) is a character object that includes a formula syntax to use in logistic regression model. If NULL, a standard additive model of response variables is assumed. Only for advanced users.
`lr_maxit`	is a maximum number of iteration of fitting algorithm of logistic regression. Default is 1000.
`MaxNWts`	is a maximum acceptable number of weights in logistic regression algorithm. Default is 5000.
`model_out`	is the logical indicating if the calculated logisitc regression model should be included in output list

Value

a list with three elements:

- output$mi - mutual information in bits
- output$pinput - prior probabilities used in estimation
- output$regression - confusion matrix of logistic regression model
- output$model - nnet object describing logistic regression model (if model_out=TRUE)

References

[1] Jetka T, Nienaltowski K, Winarski T, Blonski S, Komorowski M, Information-theoretic analysis of multivariate single-cell signaling responses using SLEMI, *PLoS Comput Biol*, 15(7): e1007132, 2019, https://doi.org/10.1371/journal.pcbi.1007132.

Examples

```
## Estimate mutual information directly
temp_data=data_example1
output=mi_logreg_algorithm(data=data_example1,
                    signal = "signal",
                    response = "response")
```

mi_logreg_main *Estimate mutual information between discrete input and continuouse output*

Description

The main wrapping function for basic usage of SLEMI package for estimation of mutual information. Firstly, data is pre-processed (all arguments are checked, observation with NAs are removed, variables are scaled and centered (if scale=TRUE)). Then basic estimation is carried out and (if testing=TRUE) diagnostic tests are computed. If output directory path is given (output_path is not NULL), graphs visualising the data and the analysis are saved there, together with a compressed output object (as .rds file) with full estimation results.

Usage

```
mi_logreg_main(dataRaw, signal = "input", response = NULL,
  output_path = NULL, side_variables = NULL, pinput = NULL,
  formula_string = NULL, lr_maxit = 1000, MaxNWts = 5000,
  testing = FALSE, model_out = TRUE, scale = TRUE,
  TestingSeed = 1234, testing_cores = 1, boot_num = 10,
  boot_prob = 0.8, sidevar_num = 10, traintest_num = 10,
  partition_trainfrac = 0.6, plot_width = 6, plot_height = 4,
  data_out = FALSE)
```

Arguments

`dataRaw`	must be a data.frame object
`signal`	is a character object with names of columns of dataRaw to be treated as channel's input.
`response`	is a character vector with names of columns of dataRaw to be treated as channel's output
`output_path`	is the directory in which output will be saved
`side_variables`	(optional) is a character vector that indicates side variables' columns of data, if NULL no side variables are included
`pinput`	is a numeric vector with piror probabilities of the input values. Uniform distribution is assumed as default (pinput=NULL).
`formula_string`	(optional) is a character object that includes a formula syntax to use in logistic regression model. If NULL, a standard additive model of response variables is assumed. Only for advanced users.
`lr_maxit`	is a maximum number of iteration of fitting algorithm of logistic regression. Default is 1000.
`MaxNWts`	is a maximum acceptable number of weights in logistic regression algorithm. Default is 5000.
`testing`	is the logical indicating if the testing procedures should be executed

`model_out`	is the logical indicating if the calculated logisitc regression model should be included in output list
`scale`	is a logical indicating if the response variables should be scaled and centered before fitting logistic regression
`TestingSeed`	is the seed for random number generator used in testing procedures
`testing_cores`	- number of cores to be used in parallel computing (via doParallel package)
`boot_num`	is the number of bootstrap tests to be performed. Default is 10, but it is recommended to use at least 50 for reliable estimates.
`boot_prob`	is the proportion of initial size of data to be used in bootstrap
`sidevar_num`	is the number of re-shuffling tests of side variables to be performed. Default is 10, but it is recommended to use at least 50 for reliable estimates.
`traintest_num`	is the number of overfitting tests to be performed. Default is 10, but it is recommended to use at least 50 for reliable estimates.
`partition_trainfrac`	is the fraction of data to be used as a training dataset
`plot_width`	- the basic dimnesions (width) of plots, in inches
`plot_height`	- the basic dimnesions (height) of plots, in inches
`data_out`	is the logical indicating if the data should be included in output list

Details

In a typical experiment aimed to quantify information flow a given signaling system, input values $x_1 \leq x_2 \ldots ... \leq x_m$, ranging from 0 to saturation are considered. Then, for each input level, x_i, n_i observations are collected, which are represetned as vectors

$$y^i_j \sim P(Y|X = x_i)$$

Within information theory the degree of information transmission is measured as the mutual information

$$MI(X,Y) = \sum_{i=1}^{m} P(x_i) \int_{R^k} P(y|X = x_i) log_2 \frac{P(y|X = x_i)}{P(y)} dy,$$

where $P(y)$ is the marginal distribution of the output. MI is expressed in bits and 2^{MI} can be interpreted as the number of inputs that the system can resolve on average.

In contrast to existing approaches, instead of estimating, possibly highly dimensional, conditional output distributions $P(Y|X = x_i)$, we propose to estimate the discrete, conditional input distribution, $P(x_i|Y = y)$, which is known to be a simpler problem. Estimation of the MI using estimates of $P(x_i|Y = y)$, denoted here as $\hat{P}(x_i|Y = y)$, is possible as the MI, can be alternatively written as

$$MI(X,Y) = \sum_{i=1}^{m} P(x_i) \int_{R^k} P(y|X = x_i) log_2 \frac{P(x_i|Y = y)}{P(x_i)} dy$$

The expected value (as in above expression) with respect to distribution $P(Y|X = x_i)$ can be approximated by the average with respect to data

$$MI(X,Y) \approx \sum_{i=1}^{m} P(x_i) \frac{1}{n_i} \sum_{j=1}^{n_i} P(y|X = x_i) log_2 \frac{\hat{P}(x_i|Y = y^i_j)}{P(x_i)} dy$$

Here, we propose to use logistic regression as $\hat{P}(x_i|Y = y_j^i)$. Specifically,

$$log\frac{P(x_i|Y = y)}{P(x_m|Y = y)} \approx \alpha_i + \beta_i y$$

Additional parameters: lr_maxit and maxNWts are the same as in definition of multinom function from nnet package. An alternative model formula (using formula_string arguments) should be provided if data are not suitable for description by logistic regression (recommended only for advanced users). Preliminary scaling of data (argument scale) should be used similarly as in other data-driven approaches, e.g. if response variables are comparable, scaling (scale=FALSE) can be omitted, while if they represent different phenomenon (varying by units and/or magnitude) scaling is recommended.

Value

a list with several elements:

- output$regression - confusion matrix of logistic regression predictions
- output$mi - mutual information in bits
- output$model - nnet object describing logistic regression model (if model_out=TRUE)
- output$params - parameters used in algorithm
- output$time - computation time of calculations
- output$testing - a 2- or 4-element output list of testing procedures (if testing=TRUE)
- output$testing_pv - one-sided p-values of testing procedures (if testing=TRUE)
- output$data - raw data used in analysis

References

[1] Jetka T, Nienaltowski K, Winarski T, Blonski S, Komorowski M, Information-theoretic analysis of multivariate single-cell signaling responses using SLEMI, *PLoS Comput Biol*, 15(7): e1007132, 2019, https://doi.org/10.1371/journal.pcbi.1007132.

Examples

```
tempdata=data_example1
outputCLR1=mi_logreg_main(dataRaw=tempdata, signal="signal", response="response")

tempdata=data_example2
outputCLR2=mi_logreg_main(dataRaw=tempdata, signal="signal", response=c("X1","X2","X3"))

#For further details see vignette
```

`prob_discr_pairwise` *Calculates Probability of pairwise discrimination*

Description

Estimates probabilities of correct discrimination (PCDs) between each pair of input/signal values using a logistic regression model.

Usage

```
prob_discr_pairwise(dataRaw, signal = "input", response = NULL,
  side_variables = NULL, formula_string = NULL, output_path = NULL,
  scale = TRUE, lr_maxit = 1000, MaxNWts = 5000,
  diagnostics = TRUE)
```

Arguments

`dataRaw`	must be a data.frame object
`signal`	is a character object with names of columns of dataRaw to be treated as channel's input.
`response`	is a character vector with names of columns of dataRaw to be treated as channel's output
`side_variables`	(optional) is a character vector that indicates side variables' columns of data, if NULL no side variables are included
`formula_string`	(optional) is a character object that includes a formula syntax to use in logistic regression model. If NULL, a standard additive model of response variables is assumed. Only for advanced users.
`output_path`	is a directory where a pie chart with calculated probabilities will be saved. If NULL, the graph will not be created.
`scale`	is a logical indicating if the response variables should be scaled and centered before fitting logistic regression
`lr_maxit`	is a maximum number of iteration of fitting algorithm of logistic regression. Default is 1000.
`MaxNWts`	is a maximum acceptable number of weights in logistic regression algorithm. Default is 5000.
`diagnostics`	is a logical indicating if details of logistic regression fitting should be included in output list

Details

In order to estimate PCDs, for a given pair of input values x_i and x_j, we propose to fit a logistic regression model using response data corresponding to the two considered inputs, i.e. y^l_u, for $l \in \{i, j\}$ and u ranging from 1 to n_l. To ensure that both inputs have equal contribution to the calculated discriminability, equal probabilities should be assigned, $P(X) = (P(x_i), P(x_j)) =$

$(1/2, 1/2)$. Once the regression model is fitted, probability of assigning a given cellular response, y, to the correct input value is estimated as

$$\max\{\hat{P}_{lr}(x_i|Y=y;P(X)), \hat{P}_{lr}(x_j|Y=y;P(X))\}.$$

Note that $P(x_j|Y=y) = 1 - P(x_i|Y=y)$ as well as $\hat{P}_{lr}(x_j|Y=y;P(X)) = 1 - \hat{P}_{lr}(x_i|Y=y;P(X))$ The average of the above probabilities over all observations y_l^i yields PCDs

$$PCD_{x_i,x_j} = \frac{1}{2}\frac{1}{n_i}\sum_{l=1}^{n_i}\max\{\hat{P}_{lr}(x_i|Y=y_i^l;P(X)), \hat{P}_{lr}(x_i^l|Y=y;P(X))\}+$$

$$\frac{1}{2}\frac{1}{n_j}\sum_{l=1}^{n_j}\max\{\hat{P}_{lr}(x_i|Y=y_j^l;P(X)), \hat{P}_{lr}(x_j|Y=y_j^l;P(X))\}.$$

Additional parameters: lr_maxit and maxNWts are the same as in definition of multinom function from nnet package. An alternative model formula (using formula_string arguments) should be provided if data are not suitable for description by logistic regression (recommended only for advanced users). Preliminary scaling of data (argument scale) should be used similarly as in other data-driven approaches, e.g. if response variables are comparable, scaling (scale=FALSE) can be omitted, while if they represent different phenomenon (varying by units and/or magnitude) scaling is recommended.

Value

a list with two elements:

- output$prob_matr - a $n \times n$ matrix, where n is the number of inputs, with probabilities of correct discirimination between pairs of input values.
- output$diagnostics - (if diagnostics=TRUE) a list correspondning to logistic regression models fitted for each pair of input values. Each element consists of three sub-elements: 1) nnet_model - nnet object summarising logistic regression model; 2) prob_lr - probabilities of assignment obtained from logistic regression model; 3) confusion_matrix - confusion matrix of classificator.

References

[1] Jetka T, Nienaltowski K, Winarski T, Blonski S, Komorowski M, Information-theoretic analysis of multivariate single-cell signaling responses using SLEMI, *PLoS Comput Biol*, 15(7): e1007132, 2019, https://doi.org/10.1371/journal.pcbi.1007132.

Examples

```
## Calculate probabilities of discrimination for toy dataset
temp_data=data_example1
output=prob_discr_pairwise(dataRaw=data_example1,
                   signal = "signal",
                   response = "response")
## Calculate probabilities of discrimination for nfkb dataset
 it=21 # choose from 0, 3, 6, ..., 120 for measurements at other time points
 output=prob_discr_pairwise(dataRaw=data_nfkb,
```

```
                 signal = "signal",
                response = paste0("response_",it))
```

`SLEMI` *Statistical Learning based Estimation of Mutual Information (SLEMI). A package for estimating mutual information and channel capacity from experimental data.*

Description

The package SLEMI is designed to estimate channel capacity between finite state input and multidimensional continuouse output from experimental data. For efficient computations, it uses an iterative algorithm based on logistic regression. In addition, functions to estimate mutual information and calculate probabilities of correct discrimination between a pair of input values are implemented.

Details

Package is deposited at GitHub: https://www.github.com/sysbiosig/SLEMI/

References

[1] Jetka T, Nienaltowski K, Winarski T, Blonski S, Komorowski M, Information-theoretic analysis of multivariate single-cell signaling responses using SLEMI, *PLoS Comput Biol*, 15(7): e1007132, 2019, https://doi.org/10.1371/journal.pcbi.1007132.

Index